VALID BUT EXCLUDED: TOKEN-PATH EXCLUSIONS IN CONSTRAINED DECODING FOR LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Constrained decoding can exclude the canonical token path of a valid JSON string even when its character parser accepts the complete text and the omitted token's characters at the reachable prefix. We distinguish these *canonical-path exclusions* from native-stack rejections caused by a narrower compiled language. On a constructed 192-case holdout, four stacks reject different path sets; two reject 48 paths each but disagree on 72. Within LM Format Enforcer, we identify and repair two token-enumeration rules without changing its character parser. The repair restores 16 excluded paths among 160 character-accepted strings while rejecting all 12 invalid controls. A separate BFCL-schema panel extends the result beyond the factorial fixtures: exclusions fall from 5/164 to 0/164 under each of two tokenizer files, on the same five cases. On 16 mask-selected generation cases, exact target copying rises from 0/16 to 16/16 for Qwen and 14/16 for Llama; 61 of 128 selected agent-state pairs also change their final-answer payloads. A larger mask audit finds that all 2,201,656 newly admitted token–prefix memberships pass the live character parser. Full-path CPU replay is substantially slower, even after a cache optimization. These results establish path restoration and selected output sensitivity, not deployment prevalence or end-to-end agent gains.

1 INTRODUCTION

LLM agents often express tool calls and final answers as JSON. Constrained decoding restricts these outputs by masking tokens before each generation step (Willard & Louf, 2023; Park et al., 2024; Dong et al., 2025). The grammar describes characters, whereas the model selects subword tokens. A single token can cross a quote, comma, escape, or property boundary. A valid continuation at the character level can therefore depend on whether the decoder considers the corresponding token in its current parser state.

We study the *canonical token path*: a tokenizer's deterministic encoding of a completed string. A native backend *rejects* this path when it omits one of its tokens. The stronger diagnosis, a *canonical-path exclusion*, requires that the complete string pass both independent schema validation and the backend's character parser, and that the omitted token's characters be accepted at the reachable prefix. This distinction separates a narrower compiled language from a token-enumeration error. Excluding one tokenization does not necessarily make the string itself unreachable.

Prior work establishes tokenizer alignment as a requirement for minimally invasive constrained generation (Beurer-Kellner et al., 2024; Ugare et al., 2025; Cognetta & Okazaki, 2025). We examine how this requirement behaves in native decoding stacks and whether repairing an exclusion changes an agent's output. The experiments address three questions: which canonical paths different stacks admit; whether a rejection comes from the character language or token enumeration; and how an enumeration-only intervention changes generation.

Figure 1 summarizes the cross-backend result. On a fixed 192-case holdout, four native JSON-Schema stacks reject different sets of independently schema-valid paths. LM Format Enforcer (LMFE) and llguidance have equal rejection counts but disagree on 72 cases. The complete per-case partition is identical under two byte-distinct tokenizer files, Qwen3.5-0.8B and Llama-3.1-8B-Instruct, without loading either model's weights. On a separate 384-case bank, LMFE and XGrammar disagree on one third of the paths. Crossing schema-property order with text-property order

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.

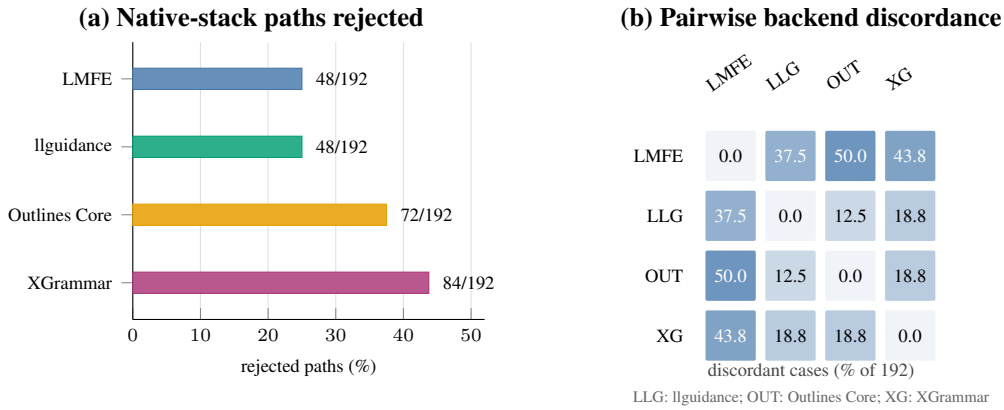


Figure 1: Native JSON-Schema stacks reject different canonical paths on the same 192-case holdout. Panel (a) shows each backend's rejection count, which is identical under the Qwen and Llama tokenizer files. Panel (b) shows pairwise disagreement between rejected sets. Equal counts do not imply equal rejected sets: LMFE and llguidance disagree on 72 cases, and the largest pairwise disagreement is 96/192. These are native path rejections, which do not require full-string character acceptance; the repair study instead conditions on 160 character-accepted paths. Counts describe the constructed holdout, not deployment prevalence. Versions: LMFE 0.11.3, llguidance 1.3.0, Outlines Core 0.2.11, and XGrammar 0.1.27.

isolates a 75.0-point XGrammar rejection effect from its declared-order compilation policy, rather than from the LMFE enumeration defect.

Within LMFE, we trace character-accepted exclusions to two interacting enumeration rules. One shortcut rejects compound fragments when tested as standalone JSON-string contents; later traversal can miss those fragments even when they are valid in context. A second cache treats raw and decoded lengths alike for backslash fragments. We repair these rules while retaining the character parser. On the shared holdout, the repair restores all 16 excluded paths among 160 character-accepted strings and leaves the registered invalid controls unadmitted. In a separate panel using BFCL schemas, it restores five excluded paths under each of the Qwen and Llama tokenizer files. The affected cases are shared across the two tokenizers, rather than ten different strings.

The path change also affects generation. Under two checkpoints, each omitted token is locally rank one in the model's next-token scores on the selected 16-case panel. Repair changes all 32 constrained outputs. In a separate panel of 128 saved checkpoint–state pairs, it restores 61 excluded reference paths and changes exactly those 61 final-answer payloads. The mask is applied after route selection, so the intervention concerns the chosen payload rather than the decision to call a tool.

Contributions.

- We compare fixed canonical paths across four native stacks, distinguish per-case disagreement from aggregate rejection rates, and separate a property-order compiler policy from a token-enumeration defect. An exact-string control admits all 16 selected texts under six backend–template combinations and both tokenizer files.
- We identify and repair two LMFE 0.11.3 enumeration rules without changing the schema or character language. The repair restores the 16 diagnosed holdout exclusions and five BFCL-schema exclusions per tokenizer. A full-mask audit tests newly admitted tokens against the live parser; CPU measurements quantify the cost.
- We connect the repair to paired outputs under two checkpoints and selected agent states. These comparisons establish path restoration and output sensitivity on the selected inputs, not deployment prevalence or end-to-end task benefit.

2 Related work

Grammar-constrained generation. Constrained decoding filters token choices against a formal language (Willard & Louf, 2023); PICARD uses incremental parsing for text-to-SQL (Scholak et al., 2021). DOMINO treats subword alignment as a requirement for minimally invasive generation (Beurer-Kellner et al., 2024), while SynCode constructs grammar-aware masks with lookahead (Ugare et al., 2025). Finite-state transduction provides tokenizer-aware representations of regular constraints (Cognetta & Okazaki, 2025). GreatGramma reports incorrectly masked valid tokens in the tested SynCode implementation (Park et al., 2025). JSONSchemaBench walks tokenized valid and invalid JSON instances through several engines to test over- and under-constraining (Geng et al., 2025).

In a vendor-authored engineering study, Picard & .txt Team (2026) compare token-mask disagreements among llguidance, XGrammar, and dotjson on 5,047 schemas and report model accuracy on BFCL's `live_multiple` subset. XGrammar studies efficient token–grammar matching (Dong et al., 2025). Our comparison uses fixed schema–text–token inputs and a crossed property-order intervention. The within-LMFE experiment then changes two enumeration rules while keeping the character parser fixed and compares outputs from the same checkpoints and saved states. This intervention attributes an output change more narrowly than cross-engine mask disagreement.

Validity and model behavior. A locally valid mask need not sample from the model distribution conditioned on the grammar (Park et al., 2024), and format compliance can rise while task quality falls (Schall & de Melo, 2025; Ray, 2026). IFEval measures verifiable instruction satisfaction rather than token-path admission (Zhou et al., 2023). StructureBench and a small-model study compare constrained decoding on structured tasks, including function calls, while separating structural validity from semantic correctness (Xiong et al., 2026; Chavan, 2026). OrderBench likewise distinguishes schema validity from unsafe or semantically wrong ordering-agent decisions (Li, 2026). Draft-conditioned decoding uses an unconstrained draft to reduce semantic distortion under a valid mask (Reddy et al., 2026). Our intervention instead changes which context-valid tokens LMFE admits; exact target copying measures its effect on selected outputs, not semantic answer accuracy.

Li et al. (2026) report suppression of XML-style tool-call tags when JSON constraints govern the same generation stream in their configuration. Function-calling work also introduces a Decision Token for function-relevance detection (Chen et al., 2025). Here the mask is applied after the route token, and exclusions occur inside the tool or final-answer payload.

Tool-using LLM agents. ReAct, Toolformer, Gorilla, and ToolLLM connect language models to external tools (Yao et al., 2023; Schick et al., 2023; Patil et al., 2024; Qin et al., 2024). BFCL evaluates function calling (Patil et al., 2025), while τ-bench evaluates tool-agent-user interactions and repeated-trial reliability (Yao et al., 2025). These task endpoints differ from the fixed-path and same-state comparisons studied here. Appendix J, Table 21, compares the constraint languages, interventions, and endpoints of these research lines.

3 Measurements

The agent first emits a route token for a tool call or a final answer, followed by a JSON object. Tool objects contain `type`, `name`, and `arguments`; final-answer objects contain `type` and `answer`. A normalizer extracts an action from the text, so the route token alone does not determine the interpreted action. The schema mask restricts next-token choices only while generating the payload.

Table 1 distinguishes acceptance of a completed string from admission of a particular token path. For a string s, let $\tau(s) = (t_1, \ldots, t_K)$ be its canonical encoding and $A(t_{<j})$ the backend's allowed-token set after prefix $t_{<j}$. A first path rejection occurs when

$$t_i \in A(t_{<i}) \text{ for all } i < j, \qquad t_j \notin A(t_{<j}). \tag{1}$$

A canonical-path exclusion additionally requires independent schema validity, complete character acceptance, and acceptance of t_j's decoded characters from that same prefix. These checks rule out an earlier unreachable prefix or a character-language mismatch as the explanation for the omitted token. Appendix B specifies the decoding and end-of-sequence (EOS) checks used with Equation 1.

Table 1: Distinct measurements. The canonical path is the specified tokenizer's deterministic encoding, not necessarily the token IDs actually generated. Rejection of one path does not establish that all tokenizations fail.

Measurement	Unit	Test
Schema validity	completed JSON	Independent schema validator accepts it.
Character acceptance	complete string, backend	Configured parser accepts the text and can end.
Native path rejection	canonical tokens, backend	First token omitted; no character-language premise.
Canonical-path exclusion	reachable prefix, backend	Character-accepted full string; omitted next token's characters accepted in context.
Output or task endpoint	generation or episode	Exact target copy, action correctness, or full task success; scored separately.

A native stack includes its compiler, tokenizer adapter, parser configuration, dependencies, and EOS policy. Comparing stacks therefore need not hold the compiled character language fixed. In contrast, the LMFE repair changes only token enumeration. For generation, *exact target copying* compares the complete output with a registered string; it is distinct from an answer rubric or a completed-task score. AgentGraph-v2 supplies deterministic, synthetic tool-use tasks and saved states (Appendix A). Its historical execution and answer metrics are defined separately in Appendix H.

The canonical encoding is a reproducible test input, not a reconstruction of the token IDs emitted by a model. Decoding consistency is checked before interpreting a mask decision, and EOS is queried only after the complete path is admitted. Passing an initially excluded position alone does not establish completion. Searching for another tokenization answers a different question: whether at least one admitted path expresses the same text. A witness leaves the original token-membership failure unchanged and does not add an observation to the factorial bank.

4 Public tokenizer tests

We first test three hand-constructed final-answer JSON strings using LMFE 0.11.3 and the unmodified public Qwen3.5-0.8B tokenizer (Gat, 2025; Qwen Team, 2026). The tests run on CPU without model weights. The schema requires `type` to equal `final` and `answer` to be a string, with no extra fields. Appendix B.1 records the tokenizer revision, dependency versions, import compatibility alias, and replay procedure.

All three strings pass independent schema validation and the complete character-parser walk, yet their canonical paths are excluded (Table 4). At each first rejection, every earlier token is admitted and the character parser accepts the omitted token's decoded increment. The nonempty mask excludes a regular vocabulary token rather than a special token or an EOS-only error fallback. These examples locate a disagreement between context-sensitive character acceptance and token enumeration, without requiring a model to generate the text.

The omitted fragments cross a JSON-string boundary or contain a partial escape. LMFE's free-text shortcut tests such fragments as self-contained JSON-string contents. All three fail that standalone test despite being legal after the current prefix, and later quote-starting traversal does not restore them. The direct cache-admission checks in Appendix B.1 isolate this behavior; Section 5.3 evaluates the full-backend repair.

Factorial replication. Two disjoint lexical banks each exhaust a $4 \times 4 \times 4 \times 3$ design: JSON boundary, escape condition, surface fragment, and string schema. Each bank has 192 cases fixed before inspecting its backend outcomes. The factor levels and construction procedure appear in Appendix B.2. A positive case must satisfy every canonical-path exclusion check, including complete character acceptance and an exactly decoded, reachable token prefix.

Each bank has 30 exclusions, with identical outcomes in all matched factor cells: 30 positive in both and 162 negative in both. All cases complete without an unknown outcome, and reverse-order cache

controls reproduce the forward membership decisions. The factor margins in Table 5 concentrate the observed exclusions in quote escapes, period fragments, and exact-length schemas. No finite-enum case is positive. Agreement across the banks shows stability over the two lexical realizations of these factors, rather than estimating a rate in agent traffic. Appendix C also documents the separate numeric-guard error and the corrected bank used for these counts.

Alternative token paths. All three initial strings have alternative tokenizations admitted through EOS by the original backend. A bounded breadth-first search finds the period and right-brace witnesses; the partial-escape witness requires an adaptive follow-up after the bounded search times out. Table 6 reports the paths, and Appendix C.1 gives the search limits and replay checks. Thus these examples demonstrate exclusion of a particular tokenization, not unreachability of the complete string. Separate numeric controls find five disagreements between JSON Schema validity and the configured character language (JSON Schema Contributors, n.d.); those are not counted as canonical-path exclusions.

5 Backend dependence and repair

5.1 Matched cross-backend paths

We run the same 384 schemas, texts, and canonical token sequences through the native JSON-Schema stacks of LMFE 0.11.3 and XGrammar 0.1.27 (Dong et al., 2025; MLC AI, 2025). The tokenizer file, case order, and EOS queries are shared. The stacks retain their compatible dependencies and native compilation policies (Appendix D.1); Appendix D collects the full comparison and repair protocols.

XGrammar rejects 168 paths and LMFE rejects 60. The paired outcomes in Table 7 show disagreement in both directions: 118 paths are rejected only by XGrammar, ten only by LMFE, and 50 by both. Neither rejected set contains the other. Each 192-case lexical bank reproduces these counts within its stack. XGrammar admits the canonical paths and EOS for all three initial LMFE examples. Its other rejections are native path outcomes, not necessarily character-accepted exclusions.

Four native stacks. The separately constructed 192-case holdout extends the comparison to LMFE, llguidance, Outlines Core, and XGrammar (Gat, 2025; Moskal, 2025; Outlines Developers, 2025; Dong et al., 2025). Each receives the same schema, text, case order, and token IDs. We repeat the panel with the Qwen3.5-0.8B and Llama-3.1-8B-Instruct tokenizer files (Qwen Team, 2026; Grattafiori et al., 2024). This holdout is disjoint from the two earlier lexical banks but is shared by the subsequent repair and generation tests; those measurements are not independent replications.

As Figure 1 shows, the stacks reject 48, 48, 72, and 84 paths, respectively. LMFE and llguidance have the same marginal count but disagree on 72 cases. Pairwise disagreement ranges from 24 to 96 cases. All 48 paths rejected by llguidance are also rejected by Outlines Core and XGrammar, whereas only 12 are rejected by LMFE. The full outcome partition is identical under the two tokenizer files. Appendix D.3 and Table 9 give the complete counts and replay checks. Aggregate rejection rates alone conceal which paths each stack removes.

Exact-language negative control. To separate the target text from its constraint representation, we select 16 texts without consulting backend outcomes and encode each as a singleton language in regex, GBNF, or Lark. All 16 canonical paths are admitted through EOS under six backend–template combinations and both tokenizer files (Table 8). Appendix D.2 specifies the 192 checks and their reverse-order repeat. These representations accept exactly the selected text; they are not equivalent rewritings of its original JSON Schema. Their success rules out the surface string alone as a sufficient explanation for rejection, while leaving schema compilation and its interaction with tokenization to be distinguished.

5.2 Property-order intervention

The largest XGrammar rejection stratum occurs when the schema lists `guard` before `slot` but the text starts with `slot`. We cross both schema-property orders with both text-property orders for 96 object-comma bases, producing 384 renderings. Values, required properties, and schema semantics

are unchanged. JSON objects are unordered collections of name/value pairs (Bray, 2017); reordering their serialized text changes its canonical token sequence. Both backends receive the same sequence for each rendering, and every object passes independent schema validation.

XGrammar rejects 48/192 paths when schema and text orders match and 192/192 when they differ, a within-base increase of 75.0 percentage points. LMFE rejects 62/192 in both conditions. Table 10 shows all four cells, and Appendix D.4 defines the paired contrast. The pinned XGrammar compiler builds its object rule in declared property order (MLC AI, 2025); LMFE's character parser instead rejects the same 16 bases in every order cell. This contrast therefore measures a compiled-language policy rather than the LMFE enumeration fault. A rejected serialization also does not imply that the JSON object's values cannot be expressed in another serialization.

5.3 A TARGETED LM FORMAT ENFORCER REPAIR

The LMFE repair (v2) changes two interacting rules. First, it retains the free-text shortcut for tokens the cache admits, while placing rejected nonquote fragments in a fallback trie. Those fragments are checked character by character from the live parser state, alongside the existing quote-starting traversal. Second, backslash fragments bypass the raw-length cache because raw and decoded lengths can differ. The schema, character parser, tokenizer, vocabulary, and EOS policy remain fixed. Appendix D.5 details the source changes and controls.

We designed the repair from the first factorial bank and fixed the shared 192-case holdout before evaluating either implementation on it. Table 2 separates the 160 character-accepted paths from the 32 rejected by the character parser. The original enumerator excludes 16 of the accepted paths; the repair admits all 160 through EOS. The 32 character-rejected paths retain their membership decisions, all 12 invalid controls remain unadmitted, and no other canonical-path outcome changes. Forward, reverse, and warmed-cache runs agree. These controls concern the registered paths, not equality of the entire allowed-token sets.

Table 2: Matched production-tokenizer holdout outcomes. The repair changes the 16 intended canonical-path membership projections, while the invalid controls remain unadmitted. Entire allowed-token sets need not be identical.

192-case holdout check	Original	Repair
Character-accepted paths	160	160
Canonical paths excluded	16	0
Character-accepted paths reaching EOS	144	160
Character-rejected paths	32	32
Invalid controls admitted	0/12	0/12
Unexpected canonical-path changes	0	0

Why both components matter. The post-hoc ablation in Appendix D.6, Table 11, tests each change on the same holdout. The fallback alone leaves all 16 exclusions. Cache bypass alone leaves those exclusions and removes 80 previously admitted paths. Only their combination restores the original 16 without changing another canonical-path outcome. The result explains why simply disabling a shortcut is insufficient: the tokens it had supplied must still be considered by contextual traversal.

An independent synthetic-tokenizer diagnostic compares the repaired mask with exhaustive trie traversal (Appendix D.7). They agree on all 1,570 tested repair-reachable, parser-accepted prefix rows, including 660 distinct newly admitted schema–prefix–token combinations. Both masks admit 83 of 84 schema-valid complete paths and reject all 35 invalid controls; the remaining valid path is rejected by the unchanged character parser. This finite check supports the contextual enumeration change without treating it as a repair of the parser's language or a general completeness proof.

BFCL schemas. A separate 192-task BFCL panel (Patil et al., 2025) yields 188 schema-valid targets; 23 LMFE type errors leave 165 evaluable paths per tokenizer. For Qwen and Llama, native rejection falls from 6/165 to 1/165, and character-accepted exclusions from 5/164 to 0/164. The same

five strings change under both tokenizers, with no reverse change. Phi remains at 1/165 native rejection and 0/164 strict exclusion. Both implementations admit 0/165 invalid controls per tokenizer (Appendix L.1, Table 23). All ten affected case–tokenizer combinations have a post-hoc EOS-valid alternative under original LMFE, each one token longer than the canonical encoding. These five strings are distinct from the three initial examples and are not naturally generated outputs.

Full-mask checks. The larger audit in Appendix L.4, Table 29, tests every recorded v2-only membership: all 2,201,656 token–prefix entries are accepted by the live character parser, and none occurs only in the original mask. The later character-set-cached variant, v3, matches v2's complete masks at all 3,318 Qwen, 3,099 Llama, and 3,815 Phi prefixes checked. These tests extend the synthetic-vocabulary check to the recorded tokenizer configurations; they do not establish independent JSON Schema correctness or equivalence at untested prefixes.

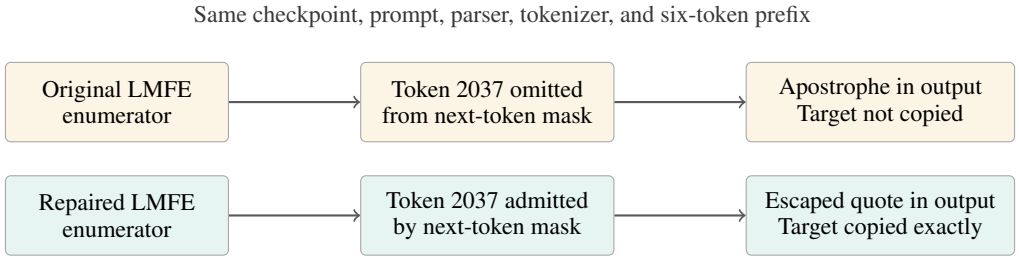


Figure 2: A paired intervention on one frozen holdout case under Qwen3.5-0.8B. At the shared reachable target prefix, token 2037 decodes to an escaped quote and ranks first in the unconstrained model's teacher-forced next-token scores. The character parser accepts its decoded characters in both conditions. Changing only LM Format Enforcer's token enumeration admits the token; greedy decoding then copies the target instead of substituting an apostrophe. Appendix D.8 gives the full schema, token path, and outputs. This trace illustrates a mechanism, not a population effect; Table 3 covers all 16 selected generation cases.

Figure 2 connects one such mask change to an output. After the same six-token prefix, the original mask excludes an escaped-quote token that the character parser accepts; the repair admits it. Under the same Qwen checkpoint and prompt, greedy decoding then copies the target instead of substituting an apostrophe. Appendix D.8 gives the schema, token path, and both complete outputs.

6 Paired Model and Agent Outputs

6.1 Selected Paired Generation

The generation panel contains the 16 character-accepted holdout cases excluded by the original LMFE backend and admitted by the repair under both tokenizer files. Selection uses mask outcomes, not model outputs. We compare unconstrained, original-constrained, and repaired-constrained generation under the same Qwen3.5-0.8B and Llama-3.1-8B-Instruct checkpoints, with fixed prompts and greedy decoding. Appendix E specifies precision, budgets, stopping rules, and repeat order; checkpoint revisions appear in Table 20.

At the first registered exclusion, the omitted canonical token is the unconstrained rank-1 next token in all 32 model–case teacher-forced scores. This is a local comparison at a shared target prefix, not a claim that the entire target is the model's globally most probable sequence. Changing the enumerator changes every constrained token path and raw output in the panel.

Table 3 reports exact target copying alongside output changes. Qwen rises from 0/16 under the original mask to 16/16 under the repair, and Llama rises from 0/16 to 14/16. The repair also matches the unconstrained output whenever that output copies the target: 14/14 Qwen cases and 13/13 Llama cases. Path admission and copying are nevertheless different outcomes. The repaired mask admits every registered target path, but Llama still fails to copy two targets. Conversely, two Qwen and one Llama targets are copied under the repaired constraint but not unconstrained decoding. These

Table 3: Exact target copying on the selected 16-case generation panel. The final column counts original-versus-repair raw output changes. Selection required original path rejection and repaired path admission before either model was run. These finite-panel outcomes are not population estimates.

Model	Unconstr.	Original	Repair	Changed output
Qwen3.5-0.8B	14/16	0/16	16/16	16/16
Llama-3.1-8B	13/16	0/16	14/16	16/16

paired comparisons isolate output sensitivity to enumeration on the mask-selected cases; they do not estimate semantic answer accuracy.

BFCL-prefix intervention. On the five BFCL cases, a separate teacher-forced comparison covers two models and two prompts (20 records). Original and v2 masked argmax choices differ in 6/10 BFCL-prompt records and all 10 exact-copy records. Adding only the omitted canonical token to the original mask reproduces v2's argmax in 20/20 records. This local ranking result is distinct from the earlier 32 rank-one scores and does not imply equality of the full distributions. Appendix L.3 reports the normalization correction that caused the preregistered total-variation replication check to fail.

6.2 Selected agent states

We evaluate 64 saved AgentGraph-v2 states under two checkpoints, giving 128 pairs in each of two comparisons. An archived mask-off/mask-on comparison locates exclusions; an original-versus-repaired comparison then keeps the mask on and changes its enumerator. States were selected by task motif from one checkpoint's mask-off histories before the later paired outcomes. Each comparison fixes the state, checkpoint, route, and decoding conditions. The 64 states come from distinct tasks, but the two checkpoints reuse them; 128 pairs are therefore not 128 independently sampled tasks. Of the 64 states, 39 use the final route and 25 the tool route. Crossing them with the two checkpoints gives 78 final-route and 50 tool-route comparisons, keeping the route-specific denominators separate. Appendix F documents both analyses, and Table 12 gives the panel composition and truncation checks.

The archived off/on comparison changes 61 raw outputs, all on the final-answer route; all 50 tool-route pairs match. Every one of the 61 unconstrained strings passes independent schema validation and the character parser. At its first difference, the original mask excludes a contextually accepted canonical token present in the tokenizer trie. The same boundary-crossing and escape classes found in the constructed examples account for these exclusions. Normalization removes 42 raw differences, leaving 19 interpreted action differences. Excluding the historically truncated states still leaves raw and normalized differences (Appendix F.1).

With the mask kept on, the original condition reproduces all 128 archived outputs. Repair restores all 61 excluded reference paths and changes the payload in exactly those pairs, with no newly invalid complete output. Schema validity stays 64/64 on validation and rises from 60/64 to 63/64 on development. All 50 tool-route payloads remain unchanged; the 61 changes occur among 78 final-route pairs. Every repaired text and token sequence matches its previously frozen unconstrained reference. Appendix F.2 and Table 13 report the paired checks and the separate post-hoc answer-rubric comparison. This fixed-state comparison does not continue full agent episodes.

7 Discussion

Locating a rejection. A schema-valid string can fall outside a stack's compiled character language, lose a token path despite character acceptance, or retain a path that the model does not select. The property-order intervention, LMFE repair, and repaired Llama outputs distinguish these cases. Equal native rejection counts do not identify equal affected paths, and the singleton-language control shows that changing the constraint representation can alter admission without changing the target

string. The BFCL-schema extension adds five excluded strings beyond the factorial design; it does not turn the constructed-bank rates into estimates of natural traffic.

Consequences for agent evaluation. In the selected states, original and repaired masks both produce 64/64 schema-valid validation outputs, yet 30 reference paths and payloads change. Validity alone misses those changes, while normalization removes many raw off/on differences. Alternative tokenizations also do not guarantee output stability: a longer path can express the same text even when the original mask removes a preferred next token. The BFCL-prefix intervention isolates that token's effect on the masked argmax, not on complete generations. These failures occur after route choice, unlike tool-tag suppression under a global JSON constraint (Li et al., 2026).

Runtime and configuration. CPU overhead persists on matched complete paths. A 23,040-observation ABBA study reports median paired v2/original ratios of 10.5300, 52.2600, and 6.2100 for Qwen, Llama, and Phi; a separate v3 study gives 5.8460, 15.0460, and 6.0690. Only pairs completing the same canonical path and EOS contribute. The smaller direct three-condition panel is reported separately (Appendix L.5, Table 30). These are single-host CPU replay costs, not GPU generation latency. On the shared 188-target BFCL panel, XGrammar 0.2.7 and llguidance 1.8.0 reject one and three paths per Qwen/Llama tokenizer. Outlines Core 0.2.14 resolves only three paths, all rejected, with 185 errors; its frozen Phi adapter is unsupported. XGrammar and llguidance reject the Phi paths at position zero (Appendix L.2). These outcomes do not revise the earlier factorial order contrast.

The archived task scores concern different interventions: AgentGraph training and mask use in Appendix F.3, compatible off/on pairs in Appendix H, and trained BFCL policies in Appendix G. They do not estimate the repair's full-task effect. Appendix K separates these observation units from the added schema, prefix, mask, and timing studies in Appendix L.

8 Limitations

The factorial banks cover chosen factors, not deployed traffic. The 192-case holdout is shared by native-stack, repair, ablation, and generation tests; these are related measurements, not independent replications. BFCL-schema paths likewise do not estimate exclusions along naturally generated token sequences. The unscreened archive lacks the emitted token IDs needed for that measurement. The BFCL path and release tests share inputs but have different evaluable subsets. Runtime errors and unsupported adapters are not admitted paths; Phi's position-zero rejections are not character-accepted LMFE exclusions.

Generation uses two checkpoints and selected cases or saved states. The BFCL-prefix tests are teacher-forced, and single-token isolation reproduces an argmax rather than a full distribution. Its normalization correction is post-outcome. One initial alternative path required adaptive search; the five BFCL witnesses were also found post hoc. The v3 cache is a later optimization, validated only on recorded fields and masks, and all CPU timings come from one host. The live-parser audit does not prove independent schema soundness or universal completeness. Exact copying and a post-hoc answer-rubric join are not full-episode outcomes; the included comparisons do not establish an end-to-end agent benefit. Historical M0 records also lack worker exit codes.

9 Conclusion

Character acceptance, canonical-path admission, and generated output are different properties. Four native stacks reject different sets of fixed paths, and a crossed order test identifies a 75.0-point XGrammar policy effect. Holding LMFE's parser fixed, the two-component repair restores all 16 diagnosed holdout exclusions and five additional BFCL-schema exclusions per Qwen/Llama tokenizer, without admitting the tested invalid controls. It changes all 32 selected model outputs, raises exact copying from 0/16 to 16/16 and 14/16, and restores 61 reference paths with corresponding payload changes in saved states. Full-mask checks support the added memberships relative to the live parser, while CPU measurements show that restoration remains costly even with caching. The evidence establishes a token-enumeration mechanism and selected output effects, not a full-task accuracy improvement.

AI USE STATEMENT

In this work, we used generative AI tools to refine hypotheses, design and review research methodology and experiments, implement and test methods and evaluation code, assist with synthetic test-case construction, and interpret results. AI-assisted data generation also included proposing alternative request texts for selected AgentGraph training tasks. The task programs and labels remained fixed, and the string-level request filters do not establish semantic equivalence (Appendix A). Separately, language models produced the experimental outputs reported in the paper; the public-tokenizer path tests use tokenizer files without loading model weights (Section 4).

Additionally, we used generative AI tools for literature search and citation checking, figure preparation, drafting sections of the manuscript, and editing for clarity and readability. Assistance with proving mathematical claims or writing proofs is not applicable, as this empirical study presents no theorem proofs.

We have reviewed all AI-assisted work, checking the final claims and citations against source material and the reported tables and figures against retained experiment records. AI-assisted implementation was tested using the independent schema-validation, character-parser and token-path replay, invalid-control, and cache-order checks described in the paper. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

An anonymous repository for this submission is available at https://anonymous.4open.science/r/VBE-ICLR27-162E. Section 3 and Appendix B specify the measurement definitions and replay checks; Appendix B.1 records package versions, tokenizer revision, and the import alias that changes neither parser nor token-enumeration code. Appendix A documents input construction, data sources, split provenance, and case and state selection. Section 5 describes the matched backend stacks, exact-language and property-order controls, and targeted repair. Appendix Sections D.6 and D.7 detail the post-hoc component ablation and finite synthetic-tokenizer check. Appendix E records the paired model-generation conditions, precision, decoding budgets, stopping rules, and run orders; Table 20 identifies the original pinned checkpoints. Appendix Section F.2 documents the fixed-state repair comparison, while Appendix H describes the separate historical mask-off/mask-on analysis. The path-level tables and figures are regenerated from retained schemas, complete texts, token sequences, run orders, controls, and matched generation settings. Table 22 distinguishes the observation units and interpretation of each result family. Appendix L describes the BFCL-schema, additional-release, prefix, mask, and timing studies, including shared BFCL inputs, tokenizer identities, outcome-specific denominators, completed-path timing eligibility, and the normalization correction.

ETHICS STATEMENT

AgentGraph-v2 uses synthetic entities and deterministic tool effects; its evaluation contains no real-world tool actions and is designed without personal data. External training and evaluation sources are documented separately in Appendix A. The study distinguishes schema validity, canonical-path admission, exact target copying, and full-task outcomes. An excluded canonical path does not establish that the entire string is unreachable, and restored paths or changed payloads do not establish semantic improvement or safer agent behavior. The constructed inputs and selected model/state panels do not estimate deployment prevalence, and the included repair comparisons do not estimate completed-episode success. The finite invalid-control and synthetic-tokenizer checks do not prove soundness or completeness for other schemas, tokenizers, or backend versions. Artifact sharing is subject to the licenses of the underlying models and datasets.

REFERENCES

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. Guiding LLMs the right way: Fast, non-invasive constrained generation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 3658–3673. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/beurer-kellner24a.html.

Tim Bray. The JavaScript Object Notation (JSON) data interchange format. RFC 8259, Internet Engineering Task Force, December 2017. URL https://www.rfc-editor.org/rfc/rfc8259.

Akash Chavan. Constrained decoding eliminates structural failures in small LLMs but reveals a scale-dependent semantic gap, 2026. URL https://arxiv.org/abs/2609.23742.

Yi-Chang Chen, Po-Chun Hsu, Chan-Jan Hsu, and Da-shan Shiu. Enhancing function-calling capabilities in LLMs: Strategies for prompt formats, data integration, and multilingual translation. In Weizhu Chen, Yi Yang, Mohammad Kachuee, and Xue-Yong Fu (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pp. 99–111, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-194-0. doi: 10.18653/v1/2025.naacl-industry.9. URL https://aclanthology.org/2025.naacl-industry.9/.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.557. URL https://aclanthology.org/2024.findings-acl.557/.

Marco Cognetta and Naoaki Okazaki. Tokenization as finite-state transduction. *Computational Linguistics*, 51(4):1119–1149, December 2025. doi: 10.1162/coli.a.23. URL https://aclanthology.org/2025.cl-4.2/.

Yixin Dong, Charlie F. Ruan, Yaxing Cai, Ziyi Xu, Yilong Zhao, Ruihang Lai, and Tianqi Chen. XGrammar: Flexible and efficient structured generation engine for large language models. In *Proceedings of Machine Learning and Systems*, volume 7. MLSys, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/hash/5c20ca4b0b20b0bd2f1d839dc605e70f-Abstract-Conference.html.

Noam Gat. LM Format Enforcer v0.11.3. GitHub repository, 2025. URL https://github.com/noamgat/lm-format-enforcer/tree/v0.11.3. Version 0.11.3, released August 24, 2025.

Saibo Geng, Hudson Cooper, Michał Moskal, Samuel Jenkins, Julian Berman, Nathan Ranchin, Robert West, Eric Horvitz, and Harsha Nori. JSONSchemaBench: A rigorous benchmark of structured outputs for language models. *arXiv preprint arXiv:2501.10868*, 2025. doi: 10.48550/arXiv.2501.10868. URL https://arxiv.org/abs/2501.10868.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi: 10.48550/arXiv.2407.21783. URL https://arxiv.org/abs/2407.21783.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

JSON Schema Contributors. Numeric types. JSON Schema Reference Documentation, n.d. URL https://json-schema.org/understanding-json-schema/reference/numeric. Accessed September 24, 2026.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=i1uGbfHHpH.

Fangzheng Li, Aimin Zhang, and Chen Lv. Constraint tax in open-weight LLMs: An empirical study of tool calling suppression under structured output constraints. *arXiv preprint arXiv:2606.25605*, 2026. doi: 10.48550/arXiv.2606.25605. URL https://arxiv.org/abs/2606.25605.

Yin Li. When JSON is not enough: Semantic reliability of schema-constrained LLM ordering agents. *arXiv preprint arXiv:2607.18261*, 2026. doi: 10.48550/arXiv.2607.18261. URL https://arxiv.org/abs/2607.18261.

Huanzhi Mao, Raymond Tsao, Jingzhuo Zhou, Shishir G. Patil, and Joseph E. Gonzalez. BFCL V4: Web search. Berkeley Function Calling Leaderboard Technical Blog, July 2025. URL https://gorilla.cs.berkeley.edu/blogs/15_bfcl_v4_web_search.html.

MLC AI. XGrammar v0.1.27: JSON Schema converter. Source code, 2025. URL https://github.com/mlc-ai/xgrammar/blob/v0.1.27/cpp/json_schema_converter.cc. Version v0.1.27, released November 4, 2025.

Michal Moskal. Low-level guidance (llguidance) v1.3.0. Software, 2025. URL https://pypi.org/project/llguidance/1.3.0/. Version 1.3.0, released October 20, 2025.

Outlines Developers. Outlines Core v0.2.11. Software, 2025. URL https://pypi.org/project/outlines-core/0.2.11/. Version 0.2.11, released May 19, 2025.

Kanghee Park, Jiayu Wang, Taylor Berg-Kirkpatrick, Nadia Polikarpova, and Loris D'Antoni. Grammar-aligned decoding. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24547–24568. Curran Associates, Inc., 2024. doi: 10.52202/079017-0774. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/2bdc2267c3d7d01523e2e17ac0a754f3-Abstract-Conference.html.

Kanghee Park, Timothy Zhou, and Loris D'Antoni. Flexible and efficient grammar-constrained decoding. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48262–48275. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/park25l.html.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 126544–126565. Curran Associates, Inc., 2024. doi: 10.52202/079017-4020. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e4c61f578ff07830f5c37378dd3ecb0d-Abstract-Conference.html.

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 Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

Robin Picard and .txt Team. Comparing the quality of structured generation engines. .txt Engineering, June 2026. URL https://blog.dottxt.ai/comparing-the-quality-of-structured-generation-engines.html.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Manoj Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Quoc Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. APIGen-MT: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=qk6ORqQ4Cu.

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://proceedings.iclr.cc/paper_files/paper/2024/hash/28e50ee5b72e90b50e7196fde8ea260e-Abstract-Conference.html.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Jaideep Ray. The constraint tax: Measuring validity-correctness tradeoffs in structured outputs for small language models. *arXiv preprint arXiv:2605.26128*, 2026. doi: 10.48550/arXiv.2605.26128. URL https://arxiv.org/abs/2605.26128.

Avinash Reddy, Thayne T. Walker, James S. Ide, and Amrit Singh Bedi. The hidden cost of structured generation in LLMs: Draft-conditioned constrained decoding. *arXiv preprint arXiv:2603.03305*, 2026. doi: 10.48550/arXiv.2603.03305. URL https://arxiv.org/abs/2603.03305.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

Maximilian Schall and Gerard de Melo. The hidden cost of structure: How constrained decoding affects language model performance. In Galia Angelova, Maria Kunilovskaya, Marie Escribe, and Ruslan Mitkov (eds.), *Proceedings of the 15th International Conference on Recent Advances in Natural Language Processing - Natural Language Processing in the Generative AI Era*, pp. 1074–1084, Varna, Bulgaria, September 2025. INCOMA Ltd., Shoumen, Bulgaria. URL https://aclanthology.org/2025.ranlp-1.124/.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, 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, pp. 68539–68551. Curran Associates, Inc., 2023. doi: 10.52202/075280-2997. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 9895–9901, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.779. URL https://aclanthology.org/2021.emnlp-main.779/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. doi: 10.48550/arXiv.1707.06347. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shubham Ugare, Tarun Suresh, Hangoo Kang, Sasa Misailovic, and Gagandeep Singh. SynCode: LLM generation with grammar augmentation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=HiUZtgAPoH.

Brandon T. Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*, 2023. doi: 10.48550/arXiv.2307.09702. URL https://arxiv.org/abs/2307.09702.

Xiaokun Xiong, Zhengjie Xu, Junyi Chen, Shihao Bai, Ruihao Gong, and Xianglong Liu. StructureBench: A unified benchmark suite for multi-scenario structured generation tasks with on-device models. In Diego Calvanese (ed.), *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI-26*, pp. 2400–2408. International Joint Conferences on Artificial Intelligence Organization, 8 2026. doi: 10.24963/ijcai.2026/267. URL https://www.ijcai.org/proceedings/2026/267. Main Track.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. doi: 10.48550/arXiv.2505.09388. URL https://arxiv.org/abs/2505.09388.

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

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b126cc38b8638e07bef37e7b2bb72bf-Abstract-Conference.html.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. doi: 10.48550/arXiv.2311.07911. URL https://arxiv.org/abs/2311.07911.

A INPUT SOURCES AND CASE SELECTION

The study uses three kinds of input. Constructed JSON objects test character and token-path behavior; AgentGraph-v2 supplies synthetic tool-use tasks and saved agent states; BFCL V4 supplies public function-calling cases. These units are not pooled. Loading a public tokenizer file for a path test does not turn a constructed JSON object into an example from that model's training data.

Constructed JSON paths. Three hand-written final-answer objects first expose the boundary failure in Section 4. Two generated lexical banks then exhaust the same $4 \times 4 \times 4 \times 3$ factor cells, once per bank. Their lexical stems are `d0` and `e9`; their complete texts and schema–text signatures do not overlap. Each registered case contains a JSON Schema, a finished valid JSON text, factor labels, and a fixed execution order. The public tokenizer file determines its canonical token path. Independent schema validation and backend character, token-mask, and EOS checks supply the measured labels; the fixtures have no human annotations. The second bank was fixed before its backend outcomes were inspected.

A third, separately generated `h7` bank repeats the 192 factor cells with different texts and schema–text signatures. It was frozen without importing LM Format Enforcer or querying a backend. It contains 192 intended schema-valid paths, 12 invalid-path controls, and 16 reverse-order cache controls. This bank is held out from repair design, but the native-backend comparison, repair test, component ablation, and selected model generations all reuse it. The 16 generation cases are the paths excluded by the original backend and admitted by the repair under both tokenizer files; selection uses mask outcomes, not model outputs. None of the designed banks estimates natural traffic prevalence.

Two further controls reuse the original lexical fixtures. The property-order test crosses two schema-property orders with two text-property orders for each of 96 object-comma base objects, giving 384 renderings. Its design follows an observed XGrammar order effect; the subsequent outcomes did not select the renderings. The exact-language control selects one case from each of 16 boundary–escape cells in the `e9` evaluation bank by its frozen run-order key, without consulting backend outcomes. It converts each selected text to an equivalent singleton regex, GBNF, or Lark language, not to an equivalent JSON Schema. A separate synthetic-tokenizer check uses 57 regular tokens, 84 generated schema-valid paths, and 35 invalid controls (Section D.7). The three initial anchors and nine numeric parser controls are not added to either factorial denominator.

AgentGraph-v2. Generator `agentgraph-v2.1`, with seed 2026, creates 40,000 training, 4,000 validation, and 4,000 development tasks. It defines 12 domains, nine graph motifs, and 120 typed tool schemas in each split's namespace. Training and validation use the same eight domains. Their sequential tasks have oracle tool-chain lengths 2–6. Development has 2,000 finance and 2,000 shopping tasks from held-out domains; its sequential oracle chains have lengths 6–8. Each task contains a request, an oracle action graph, and an expected answer, and is evaluated against a typed tool catalog. Programs generate the oracle and answer; deterministic tool validators execute calls. The frozen dataset manifest records 48,000 unique task IDs and requests. A normalized exact-match audit finds no cross-split request overlaps; this does not rule out structural similarity or model-pretraining exposure. The reserved blind split was not generated or evaluated. Templates and deterministic tools make this a controlled tool-use suite, not a sample of deployed agent traffic.

The inspected supervised runs on Qwen3.5-0.8B, 2B, and 4B draw from 40,000 AgentGraph training tasks. For 12,000 selected tasks, a language-model teacher proposed three alternative requests per task. String-level checks reject protected identifiers and tool-name or answer leakage, then retain one candidate per selected task. Only request text changes; the program and labels remain fixed. These checks do not establish semantic equivalence. The other requests remain unchanged. Revision-pinned APIGen-MT-5k contributes 5,000 executable airline and retail tool trajectories (Prabhakar et al., 2025). Another 5,000 source records come from the Tülu-3 SFT mixture after a streaming shuffle with seed 2026 and a 20,000-example buffer (Lambert et al., 2025). The trainer turns the external source records into per-action candidate examples, then selects one state per source record using the training seed. Each inspected completed SFT run records 50,000 selected examples; this is not 50,000 distinct AgentGraph tasks. The Agent-FLAN baseline (Chen et al., 2024) constructs corrective candidates before selection. The later DAgger and GRPO-KL diagnostics use separate update procedures; none of these training records tests the path-repair effect.

The fixed-state mechanism panel draws 32 validation and 32 development states from the repeat-A, seed-0, mask-off archive. A frozen ranking (seed 270913) selects four states per split from each of eight task motifs: conditional, direct no-tool, distractor, irreversible action avoidance, missing parameter, parallel, sequential, and single-tool. Selection precedes paired mask or repair outcomes. The 64 states come from distinct tasks; 39 are on the final route and 25 on the tool route. Crossing each state with two checkpoint seeds yields 128 comparisons, not 128 independently sampled tasks. A post-hoc answer check uses the task's generated expected answer only for final-route payloads. This fixed-state comparison does not continue full episodes.

BFCL-schema path extension. The additional experiments in Appendix L use BFCL schemas to test fixed JSON strings and their canonical paths, separately from the trained-policy scores below. LMFE has 165 evaluable paths per tokenizer; the Qwen and Llama character-accepted subset has 164. The release-extension panel uses 188-case counts. These denominators follow their respective experiments and are not pooled. The five changed BFCL cases are shared by Qwen and Llama and reused for the prefix and alternative-tokenization analyses. Schema provenance alone does not make these strings a sample of natural model output.

Archived BFCL V4 evaluation. BFCL is a public function-calling benchmark (Patil et al., 2025); V4 extends it with agentic evaluation (Mao et al., 2025). Our archived evaluation set contains 4,441 cases: 1,124 no-call cases in the `irrelevance` and `live_irrelevance` categories, and 3,317 cases requiring a call. The category-specific evaluator scores each archived policy on this fixed case set. We report its recorded aggregate, not a verified full-V4 leaderboard score: the published V4 aggregate combines agentic and function-calling categories with a specified weighting hierarchy (Mao et al., 2025). These archived-set scores compare differently trained checkpoints; they do not test the LM Format Enforcer repair or estimate the prevalence of canonical-path exclusions.

The historical endpoint comparison in Appendix H reuses the AgentGraph-v2 validation and development files, rather than introducing another benchmark. It retains only compatible archived off/on pairs: 3,560 validation and 3,556 development tasks per checkpoint seed, across five seeds. The exporter applies a task-contract filter independent of model outcomes. Here *compatible* means that the producer defines a schema-valid plan or an answer-only solution, not that a model succeeded. The filter excludes the recovery motif because that motif requires a missing-argument call prohibited by the producer schema. The frozen task manifest contains 440 validation and 444 development recovery tasks, giving 4,000−440 and 4,000−444 compatible tasks, respectively. Malformed or unrecognized contracts are classified as unknown rather than silently included. The finite answer rubric and execution-compliance labels are retrospective scores of stored traces; they are not new human annotations or outcomes of the enumerator repair.

B Definitions and evaluation procedure

Let s be a completed JSON string and let $\tau(s) = (t_1, \ldots, t_K)$ be its deterministic encoding under the specified tokenizer file. This path need not equal the token IDs actually generated by a model. For a token prefix $t_{<j}$, the constrained decoder returns an allowed-token set $A(t_{<j})$. We record a canonical-path exclusion at the first position j for which

$$t_i \in A(t_{<i}) \quad \text{for every } i < j, \qquad t_j \notin A(t_{<j}),$$

while the completed s is independently schema-valid, the configured backend character parser accepts s through completion, and that parser accepts the contextual decoded increment of t_j from the same reachable prefix. This definition excludes failures caused by an earlier rejected token. We call the first omitted canonical token a *native path rejection* when the full-string character check has not been imposed; it can instead reflect the compiled character language or serialization policy.

We keep four checks separate. *Schema validity* applies an independent JSON Schema validator to the completed string. *Character acceptance* walks the configured backend parser one character at a time and then tests whether it can end. *Path admissibility* walks the tokenizer's exact encoded sequence through the token mask. *EOS completion* tests whether the EOS token is allowed after the full sequence. A string is counted as a positive canonical-path exclusion only if the independent schema check and complete character walk pass, the tokenizer round trip is exact, all earlier canonical tokens

are allowed, and the next canonical token is absent despite contextual character acceptance. A native path-rejection count omits the complete-character-walk requirement.

Canonical-path exclusion is weaker than string unreachability. An alternative sequence u is a reachability witness when it decodes to the same s, every token in u is allowed from its preceding prefix, and EOS is allowed after u. We search for such witnesses separately and never count them as additional observations in the factorial bank.

B.1 Public-tokenizer environment and initial checks

We run three hand-constructed final-answer JSON strings through LM Format Enforcer 0.11.3, Transformers 5.15.0, tokenizers 0.22.2, and jsonschema 4.26.0, on CPU, with no model weights. Transformers 5.15.0 does not expose `PreTrainedTokenizerBase` at the import location expected by this LMFE release, so the process supplies an import alias to the same class. The alias changes neither parser nor token-enumeration code. The run loads the unmodified `tokenizer.json` from Qwen/Qwen3.5-0.8B (Qwen Team, 2026) at revision `2fc06364715b967f1860aea9cf38778875588b17`. A fast wrapper keeps the vocabulary and the marked special tokens. The schema requires `type` equal to `final` and a string `answer`, with no extra fields.

Table 4: Three final-answer JSON strings under LM Format Enforcer 0.11.3 and the public Qwen3.5-0.8B tokenizer (no weights). The answers are `x.`, a lone }, and a quoted `x` plus newline. Schema and character checks use the finished string. Excluded means that the allowed set omits the canonical encoding's next token at a reachable prefix; it does not mean that all encodings of the string are unreachable.

Decoded increment	ID	Pos.	Schema / chars	Token
period, quote, comma	10152	5	accept / accept	excluded
right brace, quote, comma	9262	4	accept / accept	excluded
backslash, quote, backslash	21555	5	accept / accept	excluded

For each string we validate the parsed object, walk characters through a copy of the configured parser after tokenizer integration, and test its canonical encoded tokens in order. The walk stops at the first rejection. Full-string decoding, append-only prefix decoding, and prefix decoding after a fixed prompt must reproduce the expected characters. Every earlier token is admitted, so the first omission is evaluated at a reachable prefix.

Table 4 reports the three rejected increments. All three complete strings pass the schema and character checks. Each recorded allowed-token set has 246,896 members and omits the indicated regular token; none of the three tokens is special. The initial walk stops at this first exclusion rather than checking alternative encodings or EOS. The searches in Appendix C.1 perform those additional checks.

A direct call to the unmodified free-text cache-admission method rejects all three fragments and accepts ordinary text `ok`. This call isolates the standalone-fragment check but does not run the full backend or apply a repair. Each full-backend rejection records zero `ERROR` messages; the nonempty allowed sets are not the implementation's EOS-only exception fallback. The parser-preserving repair is specified in Appendix D.5.

B.2 Factorial bank construction

Each lexical bank exhausts a $4 \times 4 \times 4 \times 3$ design. The four boundaries are object close, object comma, array close, and array comma; escape conditions are none, quote, backslash, and newline. Surface fragments are an alphanumeric control, a period, a right brace, and a right bracket. String schemas are free text, exact length, and finite enum. Each of the 192 cells is represented once. The second bank uses different lexical stems and input signatures while retaining the same factor cells. We fixed both banks before inspecting the second bank's backend outcomes. Agreement across the paired banks is therefore a transport check over lexical realization, not an increase from 192 to 384 independent samples.

The evaluation order is fixed: parse the completed JSON, apply the independent schema validator, walk the backend character parser, verify exact encode–decode round trip, walk the canonical token path, and finally query EOS. Cache-order controls repeat the same inputs in reverse order. Unknown outcomes, backend exceptions, duplicate case identities, or incomplete factor coverage invalidate a complete-bank summary.

C FACTORIAL BANK RESULTS AND CONTROLS

The discovery and evaluation banks share factor cells but not lexical stems or input signatures. Table 5 gives exact counts for either bank; the matched outcomes are identical.

Quote escapes account for 18/48 positives, compared with 4/48 for each other escape level; exact-length schemas account for 22/64, compared with 8/64 free-string and 0/64 enum cases. These are overlapping margins of the same 30 exclusions, not additional events: each fixture has one level of every factor. The crossed design localizes affected combinations without treating one factor's marginal count as an isolated causal effect.

Table 5: Exact marginal counts in either 192-case bank. The table does not treat the paired banks as independent samples.

Factor	Level	Positive	Total
Boundary	Array close / object close	4 / 4	48 / 48
	Array comma / object comma	11 / 11	48 / 48
Escape	None / quote	4 / 18	48 / 48
	Backslash / newline	4 / 4	48 / 48
Fragment	Alphanumeric / period	4 / 18	48 / 48
	Right brace / right bracket	4 / 4	48 / 48
Schema	Free / exact length / enum	8 / 22 / 0	64 / 64 / 64

The initial design used numeric `const: 0` as a nuisance guard after the target string. LMFE called `startswith` on that integer, a separate backend error. The corrected design changes only the guard to a string constant: 288 cases retain byte-identical inputs, while 96 use the corrected string guard.

The initial runtime emitted 384 rows and 32 cache controls, but 86 rows hit the numeric-constant backend error and were ineligible for a complete-bank claim. A diagnostic rerun made that failure deterministic. The amended run covers 384/384 unique case IDs, has no extra or duplicate case, no backend error or unknown, and reproduces all 32 cache controls. For the 288 byte-identical cases, every canonical-token membership annotation, first rejection, character/decode check, and EOS result matches the diagnostic run.

C.1 ALTERNATIVE-PATH WITNESSES AND PARSER CONTROLS

Table 6 reports exact-text witnesses for the three initial strings used in Section 4. The bounded breadth-first search over regular vocabulary tokens finds 13-token and 11-token alternatives for the period and right-brace examples. For the partial-escape case, it reproduces the initial rejection but reaches its prespecified 240-second bound after 1,203 states without a witness; that bounded-search result remains unknown. A separately frozen, adaptive best-first follow-up finds a 14-token path after visiting 15 states. Fresh replay reproduces each complete target and admits EOS. The adaptive witness is supplemental and does not enter the factorial denominator.

Three final-output controls and nine numeric controls separately test schema and character-parser behavior. The numeric controls probe the configured character language rather than token-path admissibility. Values 481, 501, and 1,000,001 are one above their declared maxima. The independent schema validator rejects them, but the character parser accepts them. Conversely, `480.0` and `500.0` satisfy integer schemas under JSON Schema and are rejected by the configured character parser. Exact upper-bound integers pass both checks. These five disagreements are reported as parser-language results and are excluded from every canonical-path count.

Table 6: Exact-text alternative token paths for the three initial anchors. Fresh replay admits every witness through EOS. Token counts describe the witnesses found; they are not population statistics.

Excluded token	Surface increment	Witness tokens	Search status
10152	period, quote, comma	13	Bounded search
9262	right brace, quote, comma	11	Bounded search
21555	backslash, quote, backslash	14	Adaptive follow-up

D BACKEND COMPARISONS AND REPAIR DETAILS

The controls in this appendix distinguish native-stack rejection, constraint representation, and token enumeration. Sections 5.1, 5.2, and 5.3 compare their principal results; the subsections below provide the matched inputs, complete cells, repair procedure, and diagnostic checks.

D.1 MATCHED BACKEND INPUTS

The backend comparison holds fixed every scientific input that can be shared: schema bytes, JSON text, canonical token IDs, tokenizer file, case order, and EOS query. No language model is loaded. LM Format Enforcer 0.11.3 runs with Transformers 5.15.0, tokenizers 0.22.2, and jsonschema 4.26.0. XGrammar 0.1.27 runs in its existing compatible environment. Because the installed dependency stacks differ, the comparison identifies backend-stack behavior rather than the isolated effect of changing one library file.

Table 7: Paired native-stack canonical-path outcomes on the 384-case bank. The stacks disagree on 128 paths, in both directions. Rejection does not require full-string character acceptance by both stacks.

	XGrammar 0.1.27	
LM Format Enforcer 0.11.3	Admitted	Rejected
Admitted	206	118
Rejected	10	50

The 384 paired native outcomes are 206 admitted by both backends, 50 rejected by both, 10 rejected only by LM Format Enforcer, and 118 rejected only by XGrammar. Thus neither backend's rejected set contains the other. Both 192-case lexical banks reproduce the same counts within each backend: 30 rejections per bank for LM Format Enforcer and 84 per bank for XGrammar. These counts do not impose a common compiled character language.

D.2 EXACT-LANGUAGE NEGATIVE CONTROL

Each case defines a singleton language over one complete UTF-8 string. The regex is an escaped literal; the GBNF and Lark grammars each have one literal production. This construction gives global language equivalence rather than agreement on sampled strings. We keep JSON Schema separate because it constrains parsed values and can admit multiple serializations.

The 16 cases cross four boundary types with four escaping types. Within each cell, we select the lexicographically smallest frozen run-order key without consulting a backend outcome. The six combinations per tokenizer use LM Format Enforcer 0.11.3, XGrammar 0.1.27, Outlines Core 0.2.11, and llguidance 1.3.0 (Gat, 2025; Dong et al., 2025; Outlines Developers, 2025; Moskal, 2025). The run loads tokenizer files but no model weights. Table 8 shows that all six combinations admit every selected path under both tokenizers. These are exact finite-panel counts, not estimates from a population sample.

Table 8: Exact-language negative control on 16 frozen texts, six backend–template combinations, and two tokenizer files (192 checks). Each entry of $0/16$ means no rejected canonical path and 16 paths completed with EOS enabled. The reverse-order replication matches all 192 primary records. Singleton languages are not equivalent to the original JSON Schemas.

Backend	Template	Qwen3.5 tokenizer	Llama-3.1 tokenizer
LM Format Enforcer	Regex	$0/16$	$0/16$
XGrammar	Regex	$0/16$	$0/16$
XGrammar	GBNF	$0/16$	$0/16$
Outlines Core	Regex	$0/16$	$0/16$
llguidance	Regex	$0/16$	$0/16$
llguidance	Lark CFG	$0/16$	$0/16$

D.3 Native JSON-Schema backend panel

Table 9 gives the complete marginal counts for the native panel. Each cell has 192 fixed cases. Every path not rejected reaches EOS. The versions are LMFE 0.11.3, llguidance 1.3.0, Outlines Core 0.2.11, and XGrammar 0.1.27. Forward and reverse execution orders match, and an independent fresh-process replay reproduces all 3,072 primary records. No cell has an unsupported input, compilation error, or runtime error.

Table 9: Native JSON-Schema path outcomes. "Reject" counts the first backend rejection on the canonical path; "EOS" counts complete paths with EOS enabled.

	Qwen3.5 tokenizer		Llama-3.1 tokenizer	
Backend	Reject	EOS	Reject	EOS
LM Format Enforcer	48	144	48	144
llguidance	48	144	48	144
Outlines Core	72	120	72	120
XGrammar	84	108	84	108

The pairwise discordance counts are 72 for LM Format Enforcer versus llguidance, 96 for LM Format Enforcer versus Outlines Core, 84 for LM Format Enforcer versus XGrammar, 24 for llguidance versus Outlines Core, 36 for llguidance versus XGrammar, and 36 for Outlines Core versus XGrammar. The two tokenizer files reproduce this matrix exactly.

D.4 Property-order cells

Table 10 reports every cell of the property-order intervention. The two matched cells and two mismatched cells each contain the same 96 base cases.

Table 10: Complete 2×2 property-order intervention. "Path rej." is native-stack path rejection without a full-string character-acceptance premise; "char. rej." counts full-string rejection by the LM Format Enforcer character parser. XG denotes XGrammar 0.1.27 and LMFE denotes LM Format Enforcer 0.11.3.

Schema order	Text order	Relation	XG path rej.	LMFE char. rej.	LMFE path rej.
guard, slot	guard, slot	match	24/96	16/96	24/96
guard, slot	slot, guard	mismatch	96/96	16/96	38/96
slot, guard	guard, slot	mismatch	96/96	16/96	24/96
slot, guard	slot, guard	match	24/96	16/96	38/96

The intervention changes only serialization order. For every base, we render both schema-property orders and both JSON-text orders. The values, required-property set, additional-property rule, tokenizer, and backend stack stay fixed. Text reordering changes the canonical token sequence; each

backend receives the same sequence for a given rendering. All 384 rendered objects pass the same independent Draft 2020-12 schema checks. The registered contrast is the native path-rejection rate for the two mismatched cells minus that for the two matched cells, computed within the same 96 bases. Seventy-two of the 96 XGrammar base-level contrasts equal $+1$, 24 equal zero, and none is negative. Every retained path reaches EOS in both backends. The LMFE character parser rejects the same 16 bases in every cell. This contrast measures order sensitivity, not an isolated enumeration effect.

D.5 Repair mechanism and holdout

The original free-text shortcut classifies some vocabulary fragments without their surrounding JSON-string context. A fragment rejected by that shortcut can be missing from subsequent trie traversal even when the character parser accepts it at the current prefix. A second cache uses raw fragment length for backslash fragments although decoded length can differ.

The repair changes only the token-enforcer and tokenizer-prefix-tree files. In the JSON free-text shortcut, tokens admitted by the original cache retain the original fast path and length cache. Cache-rejected, nonempty nonquote fragments enter a fallback trie and are checked character by character from the live parser state, alongside the existing quote-starting traversal. Backslash fragments bypass the raw-length cache because raw and decoded lengths can differ. Contextual token decoding, the vocabulary, marked-special-token filtering, the schema and character parser, EOS admission only when the parser can end, and the exception fallback are unchanged. The modification does not address numerical or other character-parser limits, and the finite controls do not establish soundness or completeness for all schemas.

We fixed this mechanism and the two-file repair before evaluating the 192-case lexical holdout shared with the native-stack panel. The production tokenizer encodes and contextually decodes all 192 holdout strings exactly. The holdout is evaluated in six isolated passes: original and repair conditions, each in forward, reverse, and warmed-cache orders. Sixteen separate cache-order controls must agree. Twelve invalid holdout controls must remain unadmitted. Canonical-path membership outside the registered exclusion class is checked for unintended changes; Table 2 reports none. Equality of the complete allowed-token sets is not required.

D.6 Component ablation

We evaluated the fallback trie and the backslash cache bypass in a two-factor diagnostic on the same 192 holdout cases. This ablation was run after the original and full-repair outcomes were known; it is not another independent holdout. Each variant used the same tokenizer, schemas, character parser, case order, and controls. Table 11 gives the four component combinations.

Table 11: Post-hoc component ablation on the same 160 character-accepted holdout paths. Excluded counts refer to canonical token paths, not all valid token continuations.

Fallback	Cache bypass	Excluded / 160	EOS / 160
No	No	16	144
Yes	No	16	144
No	Yes	96	64
Yes	Yes	0	160

The fallback alone leaves all 16 original exclusions. Bypassing the cache alone also leaves those 16 and excludes 80 previously admitted paths. The two components together restore the original 16 without changing any other canonical-path outcome. All variants retain the 160/32 character-accepted/rejected split, reject all 12 invalid controls, pass all 16 cache-order controls, and report no backend errors. Thus the repair depends on the combination, rather than two independent additive improvements. The bypass-only variant introduces additional exclusions rather than repairing the diagnosed class. These finite-case results do not estimate the frequency of exclusions in natural agent output.

D.7 FINITE SYNTHETIC-TOKENIZER CHECK

We also check the repair against exhaustive token-trie traversal on a small, independently constructed vocabulary. The synthetic tokenizer concatenates token strings. Its 57 regular tokens comprise 33 single-character tokens and 24 stress fragments, including escapes and fragments crossing JSON boundaries. Five schema variants supply 84 JSON Schema-valid complete target paths and 35 invalid controls. Complete target paths use single-character tokens. This check uses neither a model nor either real tokenizer file in the preceding studies.

The original and repaired masks each admit 83 of the 84 valid complete paths through EOS. The remaining case, `enum:valid:02`, is valid under its JSON Schema, but the configured character parser rejects its escaped quote; neither mask admits it. Both conditions reject all 35 invalid controls, which the same parser also rejects. Across 1,570 repair-reachable, parser-accepted prefix rows, the repair newly admits 1,746 token memberships. Repeated prefixes reduce these to 660 distinct schema–prefix–token combinations. The configured parser accepts every newly admitted continuation, and the repaired mask agrees with exhaustive trie traversal on those 1,570 prefix rows. This finite check does not prove soundness or completeness for other schemas, tokenizers, Unicode behavior, or generated agent outputs.

To obtain the 660 distinct combinations, we join newly admitted token memberships to their prefix text and schema identity, then group by schema, prefix, and token ID. The input and source snapshots are fixed, and the original and repaired replay records retain the path, character, and exhaustive-traversal checks. These synthetic diagnostics are not additional cases in the 192-case holdout.

D.8 A TRACED HOLDOUT PATH

The worked case combines an object-comma boundary, a quote escape, a right-brace fragment, and an exact-length schema. It was frozen before either holdout outcome. Its full schema is:

```
{"type":"object","required":["slot","guard"],
 "additionalProperties":false,"properties":{
 "guard":{"const":"hold-guard"},
 "slot":{"type":"string","minLength":7,"maxLength":7}}}
```

The complete target is `{"slot":"h7r\"s}}","guard":"hold-guard"}`. Independent schema validation succeeds. The original and repaired character parsers both accept all its characters and can end. The first six Qwen canonical tokens, IDs 4754, 20877, 3147, 71, 22, and 81, are admitted by both conditions and decode to the reachable prefix `{"slot":"h7r`. At position 6, token 2037 decodes to `\"`. The original mask omits it from 120,133 allowed tokens although the live character parser accepts the increment. The repaired mask includes it; all remaining target tokens and EOS are admitted. The target's canonical token-ID digest is identical between conditions.

For the same Qwen3.5-0.8B checkpoint and prompt, original constrained greedy decoding yields `{"slot":"h7r's}}","guard":"hold-guard"}`. The repaired run yields the registered target, and both emit EOS. These observations join one frozen schema, path replay, and generation pair. The reverse-order pass reproduces the same outputs; it is not a second independent case. This example demonstrates path and output sensitivity, not the frequency of this event or answer correctness.

E PAIRED MODEL-GENERATION PROTOCOL

The panel in Section 6.1 is drawn from the fixed 160-case character-accepted holdout. Its 16 cases are exactly those for which the original LM Format Enforcer path is excluded and the repaired path is admitted under both the Qwen and Llama tokenizer files. This selection uses path-level backend outcomes and is therefore outcome-informed. It does not use generations, logits, or target-copy outcomes.

For each of the 16 cases, we load the pinned Qwen3.5-0.8B and Llama-3.1-8B-Instruct checkpoints and run three conditions: unconstrained generation, the original backend, and the repaired backend. Prompts, model weights, BF16 precision, greedy decoding, maximum 64 new tokens, and stopping

rules are fixed across conditions. The forward pass contains 96 calls. A reverse-order pass contains the same 96 calls and reproduces every output. All 192 calls complete without a runtime error. They emit 2,580 output tokens and use 690.2 aggregate A100 GPU-seconds.

We score two distinct quantities. Exact target copying compares the entire generated string with the registered target. Path admission walks that target's canonical token sequence through the corresponding backend. We also teacher-force each model along the target prefix and record the rank of the first token excluded by the original backend. That token is the local argmax in all 32 model–case scores. This rank does not imply that the whole target is the globally most probable sequence. The forward/reverse runs are a deterministic replay check, not independent samples. The finite panel supports a selected-case mechanism claim only; it supplies neither a deployment denominator nor an end-to-end task metric.

F AgentGraph fixed-state and historical evaluations

The fixed-state comparisons in Section 6.2 hold the checkpoint and state fixed to test payload masking and repair. They complement the constructed-input generations in Section 6. The historical task-score comparisons in Appendix F.3 vary training or mask use instead. Appendix A specifies state selection, and Appendix I records the training settings and backbones.

F.1 Selected fixed-state panel

The path-level agent analysis uses 64 saved states evaluated under two checkpoints. State selection used task motif from one checkpoint's mask-off histories and did not use the later pair outcome. Applying both checkpoints yields 128 mask-off/mask-on pairs. The state, checkpoint, route decision, generation budget, stopping rule, and greedy decoding are fixed within each pair; only the payload mask is changed after the route token. Table 12 separates the route composition, raw and normalized differences, and truncated pairs.

Table 12: Composition and sensitivity counts for the selected fixed-state panel. Selection makes this a mechanism diagnostic, not a prevalence sample.

Panel quantity	Count
Saved states	64
Checkpoint–state pairs	128
Final-route pairs	78
Tool-route pairs	50
Raw output differences	61
Normalized output differences	19
Historically truncated pairs	8
Untruncated pairs	120
Raw differences among untruncated pairs	54
Normalized differences among untruncated pairs	12

All 61 raw differences occur on the final route, while all 50 tool-route pairs match. At each first difference, the mask-off output is independently schema-valid, the configured character walk accepts the decoded increment, and the mask omits the canonical next token. The excluded token counts are 38 for token 10152, 22 for token 9262, and one for token 21555: 30 cases are on validation and 31 on development. All three token IDs are present in the tokenizer trie; the failure concerns enumeration of boundary-crossing or escape fragments, not a missing vocabulary entry. Four long development states reached an input limit, giving eight checkpoint–state pairs. These historically truncated pairs are retained and labeled. Removing them leaves differences in 54 of 120 raw pairs and 12 of 120 normalized pairs, so truncation does not explain every split. All 1,473 queried prefix counts and membership decisions agree with the saved replay. Warmed forward, reverse, and shuffled orders each match the 61 fresh baselines (183 controls).

F.2 Repair intervention on the selected states

We keep the mask on and compare the original LM Format Enforcer 0.11.3 token enumerator with the targeted repair on the same 64 states and two checkpoint seeds. The state, route, model weights, greedy decoding, 192-token cap, and complete-object stop rule are fixed within each of 128 pairs. The repair changes only the two token-enumeration source files described above. The original condition reproduces the archived mask-on output in all 128 pairs. Table 13 summarizes path restoration, changed payloads, schema checks, invalid controls, and independent decoder replay.

Table 13: Same-state original-versus-repair intervention. The 61 restored paths are exactly the 61 pairs with changed payload tokens. The panel was selected for the mechanism and is not a prevalence or task-success sample.

Selected-state repair check	Count
Checkpoint–state pairs	128
Primary generations, original / repair	128 / 128
Exact cache-order repeats	32 / 32
Baseline-defined excluded paths restored	61 / 61
Pairs with changed payload tokens	61 / 128
Schema-valid complete outputs, original / repair	124 / 127
Newly invalid repaired complete outputs	0
Invalid controls admitted, original / repair	0 / 14, 0 / 14
Independent decoder replays, original / repair	128 / 128, 128 / 128

On validation, both conditions produce 64/64 schema-valid outputs despite restoring 30 excluded paths. On development, repair restores the other 31 paths and schema-valid outputs change from 60/64 to 63/64. All 61 restored paths occur on the final route; all 50 tool-route payloads remain unchanged.

A post-hoc join to the frozen task gold finds 58/78 original and 64/78 repaired answer-rubric matches among the 39 selected final-route states crossed with two seeds. All six additional matches occur under seed 1 in distinct irreversible-action-avoidance tasks. In all 128 state–seed pairs, the repaired text and token IDs exactly match the previously frozen unconstrained reference. These counts label restored reference outputs; they are not independent downstream outcomes or full-episode task scores. This post-hoc analysis uses the frozen scorer and paired output records.

The completion summary, experiment manifest, and independent output audit agree. The original generation controller did not persist worker exit codes or whether a SIGTERM fallback was needed. Worker termination cannot therefore be fully reconstructed, although the retained output records have the independent checks described above. The output changes do not establish improved population-level answer accuracy or whole-task success.

F.3 Historical task scores from a separate intervention

Table 14 compares archived training methods on 4,000 validation tasks with masking disabled. The JSON SFT and Agent-FLAN rows have the same reported metrics, while the token-routing and head-routing rows have lower success. DAgger and GRPO-KL are single-seed diagnostics; no DAgger candidate update was accepted. These comparisons predate the path-repair experiments and do not isolate an enumeration change.

Table 15 holds the tool-routing checkpoint fixed and compares historical mask-off and mask-on task scores. Masking raises legacy argument validity and first-action scores but lowers the reported success on both splits. The compatible-pair endpoint definitions and uncertainty calculations are given in Appendix H.

Table 16 groups the archived trace differences. Most paired traces are identical; among differences, the recorded oracle-loss class is the largest on both splits. These retrospective labels do not provide exhaustive executable-frontier verification.

Table 14: Archival AgentGraph-v2 validation, 4,000 tasks, greedy, mask off. DAgger follows Ross et al. (2011); GRPO follows Shao et al. (2024). In the DAgger diagnostic, every candidate round failed its registered validity criterion and was rolled back; the displayed final metrics equal the source checkpoint. GRPO-KL is also single-seed. These archival runs predate the prospective path-level protocol.

Method	Seeds	Success	Legacy arg. validity	First action
JSON SFT	5	88.1±0.2	85.9	89.0
Agent-FLAN SFT	5	88.1±0.2	85.9	89.0
Tool routing (tokens)	5	86.1±0.4	85.5	88.9
Tool routing (head)	5	82.1±1.1	85.3	88.8
Tokens + train-time CD	5	76.2±1.9	97.9	98.1
DAgger rollback (diag.)	1	73.7	98.3	98.5
GRPO-KL (diag.)	1	66.6	97.0	97.2
Frozen zero-shot	1	11.2	73.7	36.7

Table 15: Archival mask on versus off on a fixed tool-routing checkpoint (five seeds, budget 192/9, greedy). Repeat B matches A on every seed and condition. Historical interval estimates reported in Appendix H use 10,000 checkpoint-seed cluster-bootstrap draws. These archival runs predate the prospective path-level protocol.

Split	Condition	Success	Legacy arg. validity	First action
Validation	Unconstrained	88.6±0.4	88.0±0.4	88.9±0.0
Validation	Constrained dec.	76.2±4.2	98.0±0.8	98.1±0.7
Development	Unconstrained	82.3±0.2	86.8±0.6	80.6±0.6
Development	Constrained dec.	75.8±2.7	95.3±1.1	91.4±0.7

Table 16: Historical post hoc stored-trace labels on 20,000 paired mask-off/mask-on traces (five seeds × 4,000 tasks). Most traces match; when they differ, oracle loss is the largest recorded class. These labels summarize stored traces; they are not exact executable-frontier rescoring, and the archive has no independent verification of complete state coverage with zero disagreement.

Historical trace label	Validation	Development
Identical traces	73.2	76.1
Oracle loss	17.2	13.4
Later path	6.6	1.9
Executable dead end	1.1	4.5
Budget only	1.8	0.2
Both off oracle	0.1	3.9
Recorded frontier flag (off / on)	99.9 / 84.7	91.8 / 79.5

Table 17 compares token routing with JSON SFT across three Qwen3.5 sizes. The token-routing success mean is lower at each size, while the legacy argument-validity differences are small and change sign. These cross-training results do not measure the within-checkpoint effect of the LMFE repair.

Table 17: Token routing versus JSON SFT on AgentGraph-v2 validation (4,000 tasks, greedy, mask off).

Backbone	Method	Seeds	Success	Legacy arg. validity
Qwen3.5-0.8B	JSON SFT	5	88.1±0.2	85.9
Qwen3.5-0.8B	Tool routing (tokens)	5	86.1±0.4	85.5
Qwen3.5-2B	JSON SFT	3	87.9±0.2	86.0
Qwen3.5-2B	Tool routing (tokens)	3	86.0±0.2	85.9
Qwen3.5-4B	JSON SFT	3	88.0±0.1	86.0
Qwen3.5-4B	Tool routing (tokens)	3	87.8±0.3	86.1

G PUBLIC FUNCTION-CALLING EVALUATION

We evaluated the archived policies on our fixed 4,441-case BFCL V4 archive (Patil et al., 2025; Mao et al., 2025). The evaluator scored 45 model–method–seed rows, or 199,845 cases. An adapter failure affected 135 cases in each original row. We replaced only those registered cases, preserved the other 193,770 records, and independently verified all row manifests and the aggregate completion record. The earlier, contaminated scores are not a model baseline; Table 18 reports only the verified scores.

Table 18: Recorded accuracy on the archived BFCL V4 case set after the registered adapter repair; not a verified full-V4 leaderboard aggregate. Each seed has 4,441 cases. Means and ranges describe trained checkpoint seeds; methods are not paired decoding interventions. The Qwen3-4B row uses Instruct-2507 (Yang et al., 2025; Team, 2025), whereas the Llama row uses Llama-3.1-8B-Instruct (Grattafiori et al., 2024). The single-seed DAgger and GRPO diagnostics are omitted from this table.

Backbone	Training method	Seeds	Mean (%)	Seed range (%)
Qwen3.5-0.8B	Standard JSON SFT	5	28.07	26.39–30.89
	Token-native SFT	5	25.40	25.20–25.58
	Token-native schema SFT	5	25.31	25.31–25.31
	Protocol-head SFT	5	25.40	25.02–26.66
	Agent-FLAN SFT	5	27.50	26.08–28.89
Qwen3.5-2B	Standard JSON SFT	3	35.02	31.16–37.38
	Token-native SFT	3	26.70	25.53–27.79
	Token schema SFT	3	25.31	25.31–25.31
	Catalog schema SFT	3	36.59	26.57–44.07
Qwen3-4B	Standard JSON SFT	3	43.29	41.48–44.97
Llama-3.1-8B	Standard JSON SFT	3	35.97	31.68–38.68

Of 4,441 cases, 1,124 need no call (`irrelevance` and `live_irrelevance`); 3,317 require a call. The 0.8B token-native-schema and 2B token-schema policies score exactly 25.31%: every seed solves all no-call cases and no call-requiring cases. Token-native SFT solves 5–25 call-requiring cases per 0.8B seed; at 2B it solves 121–487 but misses 56–377 no-call cases. The 2B catalog-schema mean exceeds standard JSON by 1.57 points, but matched-seed differences span -4.59 to $+7.54$. These cross-training results do not estimate a within-checkpoint decoder effect or identify the path-level mechanism.

H HISTORICAL PAIRED-ANALYSIS DETAILS

Table 19 separates legacy success from execution compliance on compatible mask-off/mask-on pairs. Legacy success combines a strict execution contract with the finite answer rubric; execution

compliance omits that rubric. AgentGraph-v2 exact success instead requires the oracle after trusted tools. These historical endpoints do not diagnose canonical-path exclusion. The table reports lower legacy success with masking but small positive execution differences on both splits.

Table 19: Mask on minus mask off on compatible historical pairs. Counts only; no significance test. Legacy success includes the finite answer rubric; execution does not. These are neither AgentGraph exact success nor original-versus-repaired enumerator comparisons.

Split	Endpoint	Pairs	Off	On	Δ (pp)
Validation	Legacy success	17,800	15,502	13,724	-9.989
Validation	Execution	17,800	15,954	16,002	$+0.270$
Development	Legacy success	17,780	14,226	13,905	-1.805
Development	Execution	17,780	14,695	14,703	$+0.045$

For the mask comparison, every task is evaluated with the mask off and on under the same checkpoint seed. Repeat B is a determinism check and does not increase the sample size. We first compute the paired on-minus-off effect within each of the five checkpoint seeds, then average the five seed effects with equal weight. The reported uncertainty uses 10,000 bootstrap draws over checkpoint seeds while keeping each shared-task pair intact. It therefore reflects variation over the five trained checkpoints rather than treating 20,000 task rows as independent training replicates.

The historical execution-versus-answer analysis uses 35,580 compatible task–checkpoint pairs. Execution compliance asks whether the recorded tool-use contract holds; strict legacy success also applies the finite answer rubric. The reported identification interval assigns every unknown answer-rubric case first against and then in favor of the observed condition difference. It is a finite-rubric bound rather than a sampling confidence interval: $[-14.691, -2.354]$ points on validation and $[-3.138, -0.051]$ on development. The bounds remain negative under the registered finite rubric, but its literal output and nested-string requirements prevent a claim of semantic harm. Validation has 52 execution improvements and four regressions; development has 30 and 22. The historical tools are stateless, so execution compliance is not a planning oracle. On AgentGraph-v2, checkpoint-seed cluster-bootstrap intervals for mask-on changes in legacy argument validity and exact success are [9.2, 10.5] and $[-16.1, -9.8]$ points on validation, and [7.6, 9.4] and $[-8.9, -4.6]$ on development. These older scores predate the prospective mechanism studies.

The stored-trace labels in Table 16 were computed after generation from archived traces. They are descriptive and do not provide exhaustive executable-frontier rescoring. In particular, the archive lacks independent verification of 100% reachable-state coverage with zero verifier disagreement. We use these labels to localize historical changes, not to claim a complete planning-mechanism decomposition.

I AGENTGRAPH SPLITS AND BACKBONES

AgentGraph-v2 has 12 domains and 120 typed tools. Training has 40,000 tasks; validation and development have 4,000 each. Development holds out complete domains. A separate 8,000-task blind split was reserved but not generated or evaluated.

Supervised runs use one epoch, maximum length 2,048, LoRA (Hu et al., 2022) rank 16 and alpha 32, learning rate 2×10^{-5}, and an effective batch of 128. The saved DAgger run is a single-seed rollback diagnostic: every candidate round misses the registered 0.98 validity criterion, no update is accepted, and the final metrics equal its source checkpoint. GRPO-KL is also a single-seed diagnostic. GRPO was introduced as a variant of PPO (Shao et al., 2024; Schulman et al., 2017); this algorithmic lineage does not denote an additional PPO baseline in our experiments. These settings follow the saved effective run configurations. Table 20 identifies the pinned backbones used by the path studies, generation panel, and scale comparisons.

Table 20: Pinned backbone revisions for the original experiments (12-character commit prefixes). The experiment manifests record the full revisions. The CPU path controls load tokenizer files only; the paired generation panel loads the first two model checkpoints.

Role	Repository	Revision
Path studies / paired generation	Qwen/Qwen3.5-0.8B	`2fc06364715b`
Exact-language control / paired generation	meta-llama/Llama-3.1-8B-Instruct	`0e9e39f249a1`
Scale	Qwen/Qwen3.5-2B	`15852e8c1636`
Scale	Qwen/Qwen3.5-4B	`851bf6e806ef`

J Comparison with prior work

Table 21 expands the comparison in Section 2. It distinguishes tokenizer-aware construction, engine-level path checks, output-quality studies, and the parser-preserving LMFE intervention.

Table 21: Relation to adjacent results. Valid-token masking, multi-engine path checks, and mask-disagreement analysis were known. The present study pairs fixed-path measurements with a targeted LM Format Enforcer repair and selected output interventions.

Prior line	Established result	Measurement added here
DOMINO and SynCode (Beurer-Kellner et al., 2024; Ugare et al., 2025)	Tokenizer alignment and lookahead in grammar masks.	Per-case disagreement on fixed JSON-Schema inputs and a targeted LM Format Enforcer repair under an unchanged character parser.
Cognetta and Okazaki (Cognetta & Okazaki, 2025)	Finite-state transduction gives tokenizer-aware representations of regular languages.	Native JSON-Schema stack diagnosis and a source-level repair on selected model outputs using native JSON-Schema interfaces.
GreatGramma (Park et al., 2025)	Reports incorrectly masked valid tokens in the tested SynCode implementation on code grammars.	Pinned LM Format Enforcer JSON payloads, the shared native-panel/repair holdout, and selected paired generations.
JSONSchemaBench (Geng et al., 2025)	Multi-engine walks of tokenized valid JSON with EOS; finds over-constraining.	Crossed schema/text order on fixed paths and one source-level, output-changing repair.
Vendor engineering study (Picard & .txt Team, 2026)	Compares token masks across llguidance, XGrammar, and dotjson on 5,047 schemas; reports model accuracy on BFCL's `live_multiple` subset.	Fixed complete canonical paths, a within-LMFE parser-preserving source repair, and same-checkpoint selected outputs; no BFCL task-accuracy repair estimate.
StructureBench (Xiong et al., 2026)	Multi-decoder, multi-task evaluation separates output structure from semantic correctness, including tool calls.	Fixed canonical-token path checks and one source-level repair with selected paired outputs; no comparable task-accuracy estimate.
Chavan (Chavan, 2026)	Five small models under native, Outlines, and XGrammar decoding; separates schema validity from content accuracy.	Fixed token-path membership, backend disagreement, and a targeted enumerator repair; no model-scale claim.
Grammar-aligned decoding (Park et al., 2024)	A locally valid mask may distort the grammar-conditioned model distribution.	Path membership and selected output sensitivity; no full-distribution preservation claim.
XGrammar (Dong et al., 2025)	Efficient context-dependent token–grammar interaction.	Pinned native-stack order sensitivity, not internal fault localization.
Constraint-tax studies (Ray, 2026; Li et al., 2026)	Reports validity–quality tradeoffs or suppression of XML-style tool-call tags under the tested JSON constraints.	Exclusion after route choice, inside a payload; paired output consequence.

K STUDY UNITS AND INTERPRETATION

Table 22 identifies the observation unit for each experiment. Fixed JSON paths, selected checkpoint–state pairs, and archived task evaluations have different denominators and interventions. The native panel, repair, ablation, and selected generation reuse the same holdout; replay passes test reproducibility rather than add independent cases. Section 7 relates these units to the interpretation of path admission and output changes. Appendix L adds the shared 192-task BFCL selection, its backend-specific evaluable subsets, token–prefix mask checks, and completion-eligible timing comparisons; these do not enlarge the factorial sample.

Table 22: Observation units and claim scope. No row estimates deployment prevalence, and the paired-generation row is not an end-to-end agent study.

Result family	Observation unit	Supported interpretation
Finite lexical banks	Registered schema–text–token path	Exact exclusion counts and factor margins for the two constructed 192-case banks
Matched native stacks	Same path under two backend stacks	Stack-specific path admissibility and paired disagreement on the fixed bank
Native backend panel	Same holdout path under four public stacks and two tokenizer files	Native path-rejection sets and transport across two pinned tokenizer files; no common character-language premise
Exact-language control	Fixed text under backend, template, and tokenizer	All selected paths remain admissible under equivalent singleton languages; no model weights loaded
Paired model generation	Selected model–case under three decoder conditions	Rank-one exclusion and output sensitivity under two pinned checkpoints; no prevalence, semantic-accuracy, or task-success estimate
Property order	Base object under four order cells	Within-base native-stack schema/text order interaction; no isolated token-enumeration effect
Repair holdout	Registered holdout path under original and repair	Targeted removal of the diagnosed LMFE exclusion class with preserved controls
Alternative search	One complete target string	Existence of at least one EOS-valid alternative token path for each of three anchors
Selected agent panel	Fixed checkpoint–state pair	Mechanism carryover and output sensitivity in a selected, mechanism-enriched panel
Selected-state repair	Same state and checkpoint under original versus repaired token enumeration	Path restoration and payload sensitivity in a selected panel; no task-success effect
Historical mask study	Shared task within checkpoint seed	Archived paired validity and task-score differences, subject to provenance limits
BFCL V4 archive	Public evaluation case within trained checkpoint seed	Descriptive transfer on the fixed archived case set; neither a verified full-V4 leaderboard aggregate nor a within-checkpoint decoder effect

L BFCL-SCHEMA EXTENSION AND IMPLEMENTATION COST

The BFCL path and release experiments share one frozen 192-task selection, distinct from the original factorial holdout. Their evaluable subsets depend on preprocessing and backend behavior; repeated conditions and controls do not add independent tasks. The five strings restored under Qwen and Llama are reused for the post-hoc alternative-path search and the prefix analyses. The set-normalized token-isolation analysis is a post-outcome correction. The later v3 character-set cache is evaluated separately by predeclared field comparisons, complete-mask checks, and CPU timings.

L.1 BFCL-SCHEMA PATHS AND ALTERNATIVE TOKENIZATIONS

Frozen selection and target construction. The source is the 2025.12.17 release of `bfcl-eval`. We select 20 tasks from each of `simple_python`, `simple_java`, `simple_javascript`, `multiple`, `parallel`, `parallel_multiple`, `live_simple`, and `live_multiple`, plus 16 each from `live_parallel` and `live_parallel_multiple`, for 192 tasks. Within a category, tasks are sorted by the SHA-256 digest of the seed, category, and task ID, separated by NUL bytes, with task ID breaking ties; the first quota-sized subset is retained. The seed is `lmfe-real-schema-overhead-v1-20260924`. The release extension reuses these tasks without resampling.

For a task with several calls, we select the call with the smallest SHA-256 digest of the seed, task ID, call index, and function name, again separated by NUL bytes. Each argument uses the first official alternative; an optional field is omitted when its first alternative is empty. Only documented BFCL type aliases are normalized. Each target is checked against the converted schema by an independent Draft 2020-12 validator. Four targets fail because instance values do not match their declared types: `simple_java` cases 87 and 58, and `simple_javascript` cases 37 and 23. They remain preprocessing failures and are not replaced, leaving 188 mapped targets.

Evaluability and path outcomes. For each tokenizer and LMFE condition, the 192 task rows consist of four preprocessing failures, 23 runtime/compile errors caused by `Unsupported type None` on `type: any`, and 165 non-null path results. A path result can be a first rejection rather than completion through EOS. Among those 165, the character parser accepts all characters and can end on 164. The remaining case, `live_parallel_multiple_3-2-1`, has a native rejection and fails the full-character test in both implementations. Thus 188, 165, and 164 refer respectively to mapped targets, evaluable native paths, and character-accepted paths, not three independently selected panels.

Table 23 reports both native rejection and strict exclusion. Qwen and Llama each change from six native rejections to one; the five repaired cases are exactly the character-accepted exclusions. All five changes are original-only, with no reverse change. Three omitted increments are a period, quote, and comma (`."`,); two are Unicode-escape prefixes (`\u`). The same five strings are affected under both tokenizers. Phi has no strict exclusion under either condition; its single native rejection is the same character-rejected case. The full row partition is therefore $4+23+6+159=192$ for original Qwen/Llama and $4+23+1+164=192$ after repair, with the latter partition also holding for Phi in both conditions. The study's 3,456 outcome rows include repeated conditions and controls rather than 3,456 distinct target paths.

Table 23: BFCL-schema path outcomes. Native rejection uses 165 evaluable paths; strict exclusion conditions on the 164 character-accepted paths. The same five Qwen/Llama strings are repaired. Each configuration also has four preprocessing failures and 23 backend errors outside these denominators. Both LMFE conditions admit 0/165 invalid controls per tokenizer. These are path outcomes, not BFCL function-calling accuracy.

	Native rejection		Strict exclusion	
Tokenizer	Original	v2	Original	v2
Qwen	6/165	1/165	5/164	0/164
Llama	6/165	1/165	5/164	0/164
Phi	1/165	1/165	0/164	0/164

Invalid controls. Each of the 165 evaluable LMFE paths has one frozen invalid control, usually formed by deleting a required property and, in a few cases, by replacing an object with an array. Neither original LMFE nor v2 admits a complete control path with EOS: the admission count is 0/165 for every tokenizer and condition. These controls are distinct from the 12 factorial holdout controls and from the missing-field and duplicate-key families in Appendix L.2.

Tokenizer files. The path and release tests load tokenizer files without model weights. Table 24 identifies the files, decoders, and EOS choices. The local Qwen file has a recorded SHA-256 checksum but no recorded Hugging Face commit. It is not assigned the earlier `2fc06364715b` snapshot in Table 20; byte identity between those Qwen files has not been established. Full file checksums and snapshot identifiers are retained in the frozen inventory; the table gives prefixes.

Table 24: Tokenizer configurations for the BFCL path and release tests. Identifiers are snapshot prefixes for Llama/Phi and a file-checksum prefix for Qwen, not interchangeable revision types. No model weights are loaded for these path tests.

Tokenizer and decoder	Frozen identifier	EOS ID
Qwen3.5-0.8B; ByteLevel	SHA-256: `5f9e4d4901a92b99`	248046
Llama-3.1-8B-Instruct; ByteLevel	Snapshot: `0e9e39f249a1`	128009
Phi-3.5-mini-instruct; Sequence	Snapshot: `2fe192450127`	32000

The EOS tokens are `<|im_end|>` for Qwen, `<|eot_id|>` for Llama, and `<|endoftext|>` for Phi. Phi's decoder composes Replace, ByteFallback, Fuse, and Strip; this distinction determines compatibility with the frozen Outlines adapter below.

Alternative tokenizations. A descriptive post-hoc search finds an alternative path through original LMFE for every affected case–tokenizer combination: all ten witnesses reach EOS and are exactly one token longer than the corresponding canonical encoding. These are five BFCL strings crossed with two tokenizers, distinct from the three hand-written anchors in Table 6. The witnesses establish reachability, not minimum path length, and do not remove the original canonical-path exclusion.

L.2 Additional release versions and adapter boundaries

The release extension records 2,304 outcomes across XGrammar 0.2.7, llguidance 1.8.0, and Outlines Core 0.2.14 configurations. It reuses the same 192 selected tasks and four preprocessing failures, leaving 188 mapped targets. Table 25 separates resolved path verdicts from backend errors and unsupported adapters. A resolved verdict means that replay returns a path outcome without a backend exception; it need not admit every token or query EOS. These inputs differ from the older factorial holdout, so subtracting its rejection rates would not isolate a version effect.

Table 25: Native path rejections and unresolved outcomes on 188 mapped BFCL targets. Qwen/Llama counts apply separately to each tokenizer. Outlines' three resolved verdicts all reject; its 185 errors are not admissions.

Stack	Qwen and Llama (each)	Phi
XGrammar 0.2.7, default	1/188 rejected	188/188 rejected
llguidance 1.8.0	3/188 rejected	188/188 rejected
Outlines Core 0.2.14	3/188 resolved; 3/3 rejected; 185 errors	0/188 resolved; 188 unsupported

Outlines replay coverage. For both Qwen and Llama, 185 mapped targets raise `ValueError` during valid-path replay with the message `No next state found for the current state`. The failing query commonly involves EOS: token 248046 for Qwen or 128009 for Llama. Only three mapped targets return an error-free path verdict. Each is labeled

`canonical_path_excluded` in the record and stops before completion, with its EOS field unqueried. We report these as native path rejections: the outcome label alone does not establish the full-character-acceptance premise of strict exclusion. The other 185 targets have unknown path outcomes, not admitted paths. By contrast, XGrammar's default Qwen/Llama runs resolve all 188 targets with one rejection and no backend errors.

Table 26 identifies the three resolved cases and their first-rejection positions. They are the same cases under the two tokenizers, although the first position differs for one case.

Table 26: The three Outlines Core 0.2.14 cases with an error-free path verdict under Qwen and Llama. Entries are first-rejection positions; none of these paths reaches EOS. Case IDs identify BFCL tasks, not extra observations.

BFCL case ID	Qwen	Llama
`parallel_multiple_140`	10	9
`live_parallel_3-0-3`	5	5
`live_parallel_multiple_3-2-1`	3	3

Phi adapter and initial admission. All 188 mapped Phi targets are unsupported by the frozen Outlines byte-vocabulary adapter. Initialization raises `NotImplementedError` because the adapter accepts only ByteLevel decoders, whereas the Phi decoder is a Sequence (Table 24). There are zero measurable path verdicts, in addition to the four preprocessing failures. This is an adapter boundary, not evidence that all Phi paths are accepted or rejected. For each XGrammar and llguidance Phi run, all 188 paths instead receive a first-position rejection: 184 omit token 8853, whose increment is `{"`, and four omit token 6571, `{}`. This localizes those configured-stack failures to initial admission without attributing them to the same internal enumeration rule as LMFE.

Property-order option controls. Enabling XGrammar's `any_order=True` does not change Qwen/Llama valid-path counts in this extension. For each tokenizer, however, it admits 58/180 missing-required-field controls and 183/184 duplicate-key/value controls; default mode admits 0/180 and 0/184 (Table 27). These families are constructed from the 188 mapped targets and are distinct from LMFE's 165 invalid controls. Duplicate keys are a separately specified control family, not a general JSON Schema validity theorem. This option comparison does not replace the 75.0-point factorial contrast in Section 5.2.

Table 27: XGrammar 0.2.7 control admissions for each of Qwen and Llama. The denominators are the available missing-field and duplicate-key/value controls among 188 mapped targets, not LMFE's 165-control set.

Control family	Default admitted	`any_order` admitted
Missing required field	0/180	58/180
Duplicate key/value	0/184	183/184

L.3 Teacher-forced prefixes and canonical-token isolation

The prefix experiment crosses the five BFCL cases with two models and two prompt conditions, giving 20 teacher-forced records. It compares next-token choices at fixed target prefixes, rather than allowing each condition to choose its own preceding trajectory. Table 28 reports the original-versus-v2 masked argmax changes: six of ten records under the BFCL prompt and all ten under the exact-copy prompt. These are not the 32 rank-one scores from the earlier 16-case generation panel, and they are not measurements of completed BFCL calls.

A follow-up adds only the excluded canonical token to the original mask. At a tested prefix, the isolated mask is $A_{\text{iso}} = A_{\text{original}} \cup \{t_{\text{canonical}}\}$. Its masked argmax equals the full v2 argmax in all 20 records. This isolates the top-token effect on the tested prefixes; it does not show that v2 admits only one new token or reproduces the isolated mask's entire distribution.

Table 28: Teacher-forced masked argmax changes on five BFCL cases, each under two models and two prompts. Counts are paired records, not independent schemas or full generations.

Prompt	Records	Original/v2 argmax differs
BFCL	10	6/10
Exact copy	10	10/10

Normalization correction. The parent probability analysis normalized a list in which token ID 1 appeared twice; the isolation study normalized the unique token set. The preregistered total-variation (TV) replication check therefore failed. The 20/20 argmax agreement is a post-outcome correction result, not a successful preregistered distributional replication. We do not report the parent TV values as distances between unique-token distributions; those values require recomputation with consistent deduplication.

L.4 Production-tokenizer mask differences and cached equivalence

The mask-difference audit examines 2,201,656 recorded token–prefix memberships that are admitted by v2 but not original LMFE. Every one is accepted by the live character parser at the corresponding prefix, and there are zero original-only memberships. The count refers to recorded membership checks, not necessarily distinct prefixes or independently sampled strings. This extends the scale of the 57-token synthetic-vocabulary diagnostic in Appendix D.7; it is not an independent JSON Schema validator or a proof for all reachable continuations.

Repair v3 adds a post-hoc character-set cache. Its validation compares 576 rows and 4,608 predeclared fields with v2, with complete agreement. Separately, Table 29 reports equality of the entire v2 and v3 masks at every audited prefix under the three tokenizer configurations. This is stronger than matching only the canonical token's membership on those prefixes, but does not extend equality to untested schemas or parser states.

Table 29: Full-mask equality for v2 and the later cached v3 on the audited prefixes. These prefix counts are distinct from the 576-row field comparison and from the 2,201,656 newly admitted membership checks.

Tokenizer	Prefixes checked	Identical full masks
Qwen	3,318	3,318
Llama	3,099	3,099
Phi	3,815	3,815

L.5 Single-host CPU timing studies

Timed work and eligibility. The replay timer accumulates mask-query durations only up to the first canonical-token rejection; later tokens and EOS are not timed after such a rejection. A truncated observation is therefore not a complete-path time. The full-path ratios in Table 30 exclude these observations. For each case–repetition, all four observations in the ABBA block—two original and two repaired—must have no first exclusion and must admit EOS after the complete canonical path. Every included pair thus evaluates the same complete token sequence and EOS, rather than comparing early termination against successful completion.

Let $T^{(1)}_{v,c,r}$ and $T^{(2)}_{v,c,r}$ be the two path times for repair variant v in case c, repetition r, and let $T^{(1)}_{o,c,r}$ and $T^{(2)}_{o,c,r}$ be the original times in the same ABBA block. The paired ratio is

$$R^{(v)}_{c,r} = \frac{\left(T^{(1)}_{v,c,r} + T^{(2)}_{v,c,r}\right)/2}{\left(T^{(1)}_{o,c,r} + T^{(2)}_{o,c,r}\right)/2}. \tag{2}$$

The reported statistic is the median of the eligible ratios in Equation 2, not a ratio of marginal median times. The analysis also computes 5th/95th percentiles of these ratios; the table reports their

medians. The two 23,040-observation studies are separate runs, and their raw observation counts are not the number of eligible full-path ratios.

Table 30: Median paired complete-path CPU time ratios on one host. Each 23,040-observation ABBA study retains only blocks completing the path and EOS in both conditions. The last two rows share one smaller direct panel with separate completion filters, not two additional independent replications.

Study and ratio	Qwen	Llama	Phi
ABBA: v2/original	10.5300	52.2600	6.2100
Separate ABBA: v3/original	5.8460	15.0460	6.0690
Direct panel: v3/v2	0.2180	0.1070	0.4210
Direct panel: v3/original	4.2550	12.0850	6.0400

Table 31 gives the v2/original eligible cases and 30 timing repetitions per case, not additional target strings. Construction and first-mask ratios are close to one despite the larger full-path ratios.

Table 31: Eligible v2/original ABBA complete-path comparisons. Each case has 30 repetitions; each paired ratio averages two observations per condition. These counts do not specify the separate v3 study or the direct panel.

Tokenizer	Eligible cases	Repetitions/case	Paired ratios
Qwen	55	30	1,650
Llama	53	30	1,590
Phi	56	30	1,680

Cached v3. The separate v3/original ABBA study uses the same completion rule and Equation 2. Table 32 reports its absolute median v3 path times separately from the v2 ratios. Dividing independent ratio medians does not recover a paired v3/v2 speedup; dividing an absolute median by a ratio median does not recover original latency.

Table 32: Absolute median v3 complete-path times in its separate ABBA study. These are CPU replay times in milliseconds, not v2 times or complete-model generation latency.

Tokenizer	v3 median time (ms)
Qwen	153.1500
Llama	287.2600
Phi	32.6700

Direct three-condition panel. The direct study crosses 32 cases, 12 repetitions, three conditions (original, v2, v3), and three tokenizers, giving 3,456 rows. Hot queries use a frozen prefix reachable in all conditions, with equal query counts and 10 internal repetitions. Complete-path timing still stops at a rejection. The finalizer admits v3/original comparisons only when the original and v2 path statuses are complete; v3/v2 comparisons use the dedicated v2/v3 completed-path eligibility test. The eligible median ratios appear in the last two rows of Table 30; hot-query ratios are close to one. This smaller direct panel does not replace either ABBA study. Caching reduces path cost, but median repair/original ratios remain above one on the included completed paths.

Recorded execution environment. The runs use one CPU-only x86_64 host with Linux `5.15.0-130-generic` and 64 logical CPUs; no CUDA devices are visible. Durations use `time.perf_counter_ns`. No separate warm-up iterations are discarded: a warm query repeats a query at an already reached prefix. Peak process RSS is recorded through `ru_maxrss`. The processor model, CPU affinity, and thread count were not frozen. These measurements therefore describe the observed single-host runs, not hardware-independent speed ratios, end-to-end GPU decoding costs, or deployment latency.