Verification Surfaces in Language-Model Systems: Token, Schema, and Structured-Output Reliability

Michael Rothrock

michael@roth.rocks · https://michael.roth.rocks

Version 1.0 · DOI: 10.5281/zenodo.20331399 · Companion to Trust Topology *(DOI: 10.5281/zenodo.20292194)*

Abstract. *Schema validity is not semantic correctness.* A JSON schema can make malformed output impossible, but it cannot by itself prevent hallucinated entities, wrong sentiment, or inconsistent domain semantics. This paper studies language-model reliability as a problem of verification-surface design: choosing the representation a system exposes and the checks that can be written over it. The practical claim is constructive: when a reliability method fails, the remedy is often not a stronger judge over the same output, but a new surface that makes the target failure visible.

On SemEval-2014 ABSA restaurants ($n = 980$, public), different checks catch different mistakes. A lexical-faithfulness check is excellent at finding unsupported aspects (96.3% precision), but weak at finding wrong polarity (25.9%). A cross-model-disagreement check reverses the pattern: weak on aspect mismatch (14.1%), stronger on polarity mismatch (59.8%). The two checks reject mostly different examples (Jaccard 0.032, 95% CI [0.009, 0.057]), and the same inversion appears with a second open-weight model pair and on a second naturalistic domain (SemEval-2014 laptops, $n = 901$).

The same pattern appears at other layers. In structured-output experiments, hard-constrained decoding guarantees only the provider-compatible part of the schema dialect actually enforced; numeric bounds, regex patterns, and cross-field consistency remain provider- and keyword-dependent. At the token layer, runtime-deployable structural checks over 3×10^6 candidate-token records (three seeds) fire on nearly disjoint cases, and their usefulness changes with the target: `bigram_low_prob` has 46 × lift on rank-disagreement (rank > 256) but 0.87 × lift on gold-correctness.

The conclusion is operational: schema constraints, validators, model disagreement, self-consistency, judges, and token uncertainty are different verification surfaces, not interchangeable reliability methods. The engineering loop is to identify what the current surface cannot see, expose a representation that contains the missing signal, and then verify against that new surface. The question is not which reliability method is best in general; it is which failure mode is visible to the surface you chose, and whether you need to build a new one.

1 Introduction

Monday morning. A developer opens the dashboard for last night's batch: 980 customer reviews fed through an LLM extraction pipeline. Each row has been through the same chain: prompt, schema-constrained JSON, downstream sentiment routing. The developer's job is not to second-guess the model, but to catch the cases where the pipeline broke down before the routing decision goes out. Every row passed schema validation. Row 47: `aspect_term: "price"`, but the review says "the $20 cocktails were fine"; the model emitted an abstract category instead of the literal span. Row 312: Llama says polarity is `positive`, Qwen on the same input says `negative`. Row 619: a token in the generated explanation has 3% probability under a reference model that otherwise agrees with the output. None of these are schema failures; they are pipeline failures, and the schema catches the wrong target.

This paper asks: which of those checks can run automatically, at zero false positives? And which ones cannot, because the verification surface is misaligned with the target?

Schema validity is not semantic correctness. *Surface-target alignment is target-specific*: the right gate depends on which kind of "wrong" you are trying to catch, and the same surface inverts its precision across targets.

Web-form gloss. A dropdown menu makes a misspelled answer physically impossible and a wrong answer effortless. Constrained decoding is the dropdown: it removes invalid outputs from the model's support, and it says nothing about whether the selected option is the right one. Every result in this paper is a version of that distinction, measured.

Reliability in language-model systems is not a global property of the model. The Trust Topology (Rothrock, 2026a) formalizes this: a system deterministically verifies exactly the targets whose pure-positive buckets carry positive mass under the chosen representation. A *verification surface* is the tuple $\left(Y_k, \mathcal{D}_k, F_k^{\text{deploy}}, L_\alpha\right)$, a representation map Y_k, a deterministic predicate class $\mathcal{D}_k$, a deployed gate family, and a target violation event L_α. This paper is the bridge between the flagship's pipeline-level result and general language-model reliability: where the flagship coding-agent study (Rothrock, 2026a) measures verification surfaces over a software pipeline's artifacts, here the same construction drops to the language-model layer itself (token emission and structured output) and maps the general LM reliability mechanisms practitioners already use (schema constraints, validators, model disagreement, self-consistency, judges, and token uncertainty) onto the verification-surface design space to measure the empirical content of this view's predictions.

We report five results.

(1) On naturalistic data, surface-target alignment is sharp and inverts cleanly. On SemEval-2014 ABSA restaurants (Pontiki et al., 2014) ($n = 980$, public), a lexical-faithfulness surface (B) achieves 96.3% precision on the aspect-mismatch target but 25.9% on polarity; a cross-model-disagreement surface (D) inverts (14.1% aspect, 59.8% polarity), with $B \cap D$ Jaccard 0.032. The inversion preserves under a second open-weight pair and a second naturalistic domain (SemEval-2014 laptops, $n = 901$). §3.

(2) Hard-constrained decoding's support guarantee is narrower than "JSON is valid." Across 500 schema-validity prompts and 169 per-constraint tool-call prompts, hard-constrained decoding achieves 100% *structural* validity only for the *provider-compatible* subset of the schema actually enforced; weak-prompt free decoding collapses to 0%. §4.

(3) The per-constraint-kind enforcement matrix is the most practitioner-actionable artifact of this study. Across the provider arms tested, structural and enum constraints are retained, numeric and string-length bounds partially, and cross-field consistency not at all, with the specifics provider- and keyword-dependent. A post-hoc validator is required outside the provider-compatible subset. §4.

(4) On the templated extraction-monitor target, a deterministic cross-model gate matches the LLM judge at a fixed alert budget. At matched rate, a cross-model disagreement gate reaches 68.3% precision versus a Claude Sonnet 4.6 judge at 54.5% and within-model self-consistency at 52.4% (Holm-corrected (Holm, 1979) tests do not separate them on the aggregate target; each beats the marginal at Holm $p < 10^{-8}$), at no extra judge call when a second extractor already exists. §5.

(5) At the token level, gates with structurally different predicates fire on nearly disjoint cases, and the same gate inverts across targets. Over 3,000,000 candidate-token records, cross-family Jaccard is ≤ 0.012 and `bigram_low_prob` lift flips from 46 × at rank-disagreement (rank > 256) to 0.87 × (anti-aligned) at gold-correctness; three negative controls confirm structural specificity. §6.

Across all five, no single verification method dominates: surface choice determines target alignment. Composing target-aligned, disjoint surfaces is the constructive corollary: across three hard domains, a union of the high-purity surfaces raises coverage at preserved purity (§5.5).

1.1 Practical implications

For researchers. The cross-family disjointness measurement argues against assuming a single universal correctness score: no gate or surface tested here behaves as a universal correctness monitor (max cross-family Jaccard 0.012 at the token level; $B \cap D = 0.032$ on naturalistic ABSA). A more precise research program is to catalog which (surface, target) pairs align, build the lattice of decidable predicates over LLM outputs, and treat the (surface, target) tuple as the unit of evaluation. On the templated extraction-monitor target tested here, a deterministic cross-model disagreement gate has higher point-estimate precision than the LLM judge (Gu et al., 2024; Zheng et al., 2023) (68.3% vs 54.5%, §5.6) at the same rate match, though pairwise Holm-corrected (Holm, 1979) tests do not separate them on the aggregate target; each monitor individually exceeds the marginal at Holm $p < 10^{-8}$. The practical implication is that a second extractor, if already part of the monitoring design, supplies a deterministic surface with no extra judge call.

For practitioners. Three operational consequences. (1) *Structured output is necessary but not sufficient.* The provider-compatible schema dialect strips constraint kinds the reference schema asserts (as tested: the Anthropic strict arm (Anthropic, 2026) drops numeric and array bounds; Gemini structured output (Google, 2026) accepts numeric bounds but leaves them to client-side validation rather than decode-time enforcement; Ollama grammar (ggml-org, 2026) enforces integer bounds at the decoder but drops `pattern` at runtime though the guide lists it as supported; no provider arm tested enforces cross-field consistency at decoder level, §4), and structural validity does not preclude lexical hallucination (surface B fires on 13.78% of grammar-constrained ABSA extractions, §3). A post-hoc validator is still required for the constraints the decoder cannot enforce. (2) *For these structured-output targets, target-aligned composition can be higher-leverage than model escalation.* For a fixed model on the targets tested here, surface choice varies precision by orders of magnitude; composing cheap target-aligned surfaces (e.g., a second open-weight extractor's disagreement signal) can be higher-leverage than escalating to a larger model or adding a generic judge. Model upgrade may still be the right move for other tasks; this is a claim about **these** targets. (3) *"Hallucination" is not one failure mode.* Lexical hallucination, cross-field inconsistency, and cross-model uncertainty are structurally distinct (pairwise Jaccard 0.000, 0.037, 0.171 on the hard corpus, §5.4); a monitor calibrated for one does not catch the others. Pick the surface that matches the target you care about.

If you care about	Recommended first surface	Second surface	Caveat
JSON parse / structural validity	hard schema or grammar decoding	post-hoc validator	only the provider-compatible schema subset is decoder-enforced (§4)
Enum / type validity	hard schema / strict structured output	post-hoc validator	well-supported across providers tested (§4)
Numeric / array bounds	grammar-mode (llama.cpp, integer)	post-hoc validator	Gemini accepts but does not decoder-enforce; Anthropic strict strips (§4)
Regex / pattern	Anthropic strict structured output	post-hoc validator	as tested, Gemini under-enforces and Ollama grammar strips `pattern` (§4)
Cross-field relation	post-hoc validator	assertion-with-retry	not decoder-expressible in current dialects (§4)
Entity / lexical faithfulness	lexical-faithfulness surface (B)	retrieval / citation check	low recall for paraphrase (§3)
Label / decision uncertainty	cross-model disagreement (D)	within-model self-consistency	operating point is pair-specific (§3, §5)
Token-level distributional anomaly	structural token predicates (bigram, boundary)	continuous-score AURC ranker (Geifman & El-Yaniv, 2017) over $-\log P_{\text{bi}}$	low overlap with gold-NLL gates (§6, §7)
Runtime token uncertainty	candidate-side gates (top-1 prob, entropy, margin)	cross-model argmax disagreement	teacher-forced NLL is evaluation-only (§6.1)
Global semantic correctness	no single surface — decompose target	task-specific rubric over composed surfaces	unbounded semantic correctness is not generally decidable; decompose into local targets (§9)

Table 1: Decision guide: which verification surface to reach for first by target. Recommended surfaces are drawn from the families measured in §3-§7; the caveat column names the limitation a practitioner should plan around. Operating points and pair-specific magnitudes are documented in the cited sections.

How to read this paper. Practitioners can start with the decision table, §3, and §4. Researchers in LM uncertainty should add §6–§7 and the Supplemental Analysis. §8 maps the constrained-decoding ecosystem onto the framework vocabulary.

The body delivers the five results in the order promised: §3 reports the naturalistic ABSA results, §4 the implementation hierarchy and per-constraint-kind enforcement matrix, §5 the templated cross-class surfaces and monitor comparison, §6 the token-level surfaces, §7 the target-severity sensitivity, and §8 maps the constrained-decoding ecosystem (Outlines (Willard & Louf, 2023), JSONFormer (1rgs, 2023), LMQL (Beurer-Kellner et al., 2023), Guidance (Microsoft, 2024), OpenAI/Anthropic/Gemini structured outputs (Anthropic, 2026; Google, 2026; OpenAI, 2024), DSPy (Khattab et al., 2024)) to this approach's vocabulary. §9 discusses limitations and §10 concludes.

Scope. This is a target-specific verification-surface topology paper, not a constrained-decoding benchmark or a SOTA selective-generation result. We do not compare against learned QC heads or reward models. The paper is self-contained as a language-model study; the flagship paper introduces the framework, and its Formal Supplement gives the capacity results and notation.

2 The verification surface, applied to language models

For readers approaching this paper without the flagship paper, this section is a compact recap of the concept. The empirical results stand on their own. The framework gives them their interpretation.

A *verification surface* at processing stage k against target violation type α is the tuple

$$\Sigma_{k,\alpha} := \big(Y_k : \Omega \to S_k, \mathcal{D}_k, F_k^{\text{deploy}}, L_\alpha\big)$$

where Y_k is the representation map. Here it is a token-level emission of a candidate next token, or, for constrained decoding, a structured-output object. $\mathcal{D}_k$ is the deterministic predicate class over S_k (here, threshold predicates over a smoothed bigram, byte-class predicates, length-bin predicates, etc.), F_k^{deploy} is the deployed gate family (deterministic plus optional stochastic/heuristic gates), and $L_\alpha \subseteq \Omega$ is the target violation event (here, a section-specific violation event such as schema invalidity, aspect mismatch, cross-model disagreement, or token-rank corruption).

The *target-relative deterministic subfamily* collects deployed gates whose rejection events are deterministic and contain only target-positive cases. Writing $R_{k,g}$ for the rejection event of a deployed gate g, in the deployed-gate-event notation of the flagship and its Formal Supplement:

$$F_{k,\alpha}^{\text{det}} := \Big\{ g \in F_k^{\text{deploy}} : R_{k,g} = \big\{Y_k \in A_g\big\}, A_g \in \mathcal{D}_k, Y_k^{-1}(A_g) \subseteq L_\alpha \ \ \text{modulo} \ \ P\text{-null sets} \Big\}.$$

A gate may be in $F_{k,\alpha}^{\text{det}}$ for one target and not for another.[1] Three measurement attributes characterize the surface:

- $\eta_{k,\alpha}^*$: *capacity*, the supremum of target-positive mass over zero-false-positive rejection regions. The flagship Formal Supplement's capacity characterization identifies this with $P\big(C_{k,\alpha} \mid L_\alpha\big)$, the target-positive mass in pure-positive buckets. These are values of Y_k whose preimages (buckets) lie entirely inside L_α. A bucket is the set of cases the representation renders as the same value; two cases in the same bucket cannot be distinguished by any predicate over Y_k.
- $\rho_{\alpha(g)}$: *purity*, $P\big(L_\alpha \mid R_{k,g}\big)$, the target-positivity rate among gate g's rejections; if $R_{k,g} = \big\{Y_k \in A_g\big\}$ then $\rho_{\alpha(g)} = P\big(L_\alpha \mid Y_k \in A_g\big)$. Membership in $F_{k,\alpha}^{\text{det}}$ requires $\rho_\alpha = 1$.
- $\Gamma_{k,\alpha}$: *target coverage*, observed recall of the full deployed family. Writing $E_{k,\alpha} = \bigcup_{g \in F_k^{\text{deploy}}} R_{k,g}$ for the event that some deployed gate rejects, $\hat{\Gamma}_{k,\alpha} = P\big(E_{k,\alpha} \mid L_\alpha\big)$. We always report $\hat{\Gamma}$ rather than $\hat{\eta}$ when we have not verified the rejection region as zero-false-positive.

A fourth, ω, measures complementarity within or between surfaces and is the Jaccard index of rejection sets (intersection over union: 0 means the two gates reject disjoint cases, 1 means identical rejection sets).

Concretely: η bounds what the surface can catch with zero false positives, ρ measures whether a specific gate is clean, Γ is how much of the target the deployed gates actually rejected, and ω is how much different gates overlap.

The theory's core empirical prediction for language models. The ordinary token-emission representation typically does not place semantic-correctness targets in pure-positive buckets, so the token-level distributional predicates over it (bigram, byte-class, length-bin, boundary) cannot isolate them. The same token can be correct in one context and a hallucination in another. Therefore, formal $\rho = 1$ with nontrivial coverage on a semantic target requires either a richer representation (structured output, schema, grammar) or a richer predicate class (parser, type-checker). In this view, constrained-decoding methods are the LLM community building those richer verification surfaces, mostly without naming them as such.[2]

3 Naturalistic cross-class surfaces on SemEval-2014 ABSA

This is the paper's empirical anchor. On public naturalistic data, two structurally different checks over the same JSON extractions catch almost entirely different mistakes, and which check is more precise flips depending on which mistake you score it against. It tests the verification-surface view at the structured-output layer, on a benchmark the field already uses rather than one built for this study.

[1] All inclusions in this paper are understood up to P-null sets.

[2] In mathematical terminology, these buckets are *fibers* of the representation map; we use *bucket* for accessibility, matching the flagship paper.

Corpus. SemEval-2014 Task 4 restaurant reviews (Pontiki et al., 2014) (the standard aspect-based sentiment analysis benchmark), via the Hugging Face mirror `tomaarsen/setfit-absa-semeval-restaurants` (Aarsen, 2023). We filter to single-aspect rows with polarity in {positive, negative, neutral} (dropping `conflict`) and combine the train + test splits in deterministic order, yielding $n = 980$ naturalistic restaurant-review snippets with annotated aspect terms and polarity labels.

Extraction. Constrained schema `{aspect_term: string, polarity: enum(positive, negative, neutral)}`. Primary pair: `llama3.1:8b` (Grattafiori et al., 2024) and `qwen2.5:7b` (Yang et al., 2024) via Ollama llama.cpp grammar mode (ggml-org, 2026); schema-valid rate 100% on $\frac{980}{980}$ inputs. Robustness pair: Mistral 7B Instruct (Jiang et al., 2023) + Gemma 2 9B Instruct (Gemma Team, 2024), same pipeline, schema-valid rate 100%.

Surfaces. Two cross-class surfaces apply to ABSA.

Surface B, Lexical faithfulness. $Y_B = (\text{input\_text}, \text{extracted.aspect\_term})$; D_B rejects iff the extracted `aspect_term` does not appear in the input text (normalized substring match). Catches hallucinated aspect terms. Aligned target L_B: extracted aspect term is not lexically supported by the source text.

Surface D, Cross-model disagreement. $Y_D = (\text{extraction}_A, \text{extraction}_B)$ for two distinct models; D_D rejects iff `polarity_A != polarity_B`. Catches cross-model polarity disagreement. Aligned target L_D: cross-model disagreement, or low-consensus extraction.

3.1 Rejection rates and per-target alignment (Llama 3.1 8B + Qwen 2.5 7B)

The table reads off the inversion directly: each surface's precision swings sharply with the target it is scored against.

Surface	**Rejections**	**Rate**	**Wilson 95% CI** (Wilson, 1927)
B (lexical faithfulness)	$\frac{135}{980}$	13.78%	[11.76%, 16.08%]
D (cross-model polarity)	$\frac{92}{980}$	9.39%	[7.72%, 11.38%]

Table 2: Per-surface rejection counts on SemEval-2014 ABSA restaurants ($n = 980$). Surface B fires on 13.78% of rows, a substantial natural hallucination rate, where the constrained extractor emits abstract aspect categories (e.g., `price`, `service`, `ambience`) instead of the literal span (e.g., `$20`, `staff`, `mezzanine`) despite grammar enforcement.

Surface vs. target	$T_{\text{aspect\_wrong}}$	$T_{\text{polarity\_wrong}}$	$T_{\text{any\_wrong}}$
B (lexical) — $\hat{\rho}$	96.30%	25.93%	97.04%
B — $\hat{\rho}$ Wilson 95% CI	[91.62%, 98.41%]	[19.27%, 33.91%]	[92.63%, 98.84%]
B — $\hat{\Gamma}$ (recall)	70.27%	20.23%	42.95%
D (cross-model) — $\hat{\rho}$	14.13%	59.78%	65.22%
D — $\hat{\rho}$ Wilson 95% CI	[8.45%, 22.69%]	[49.57%, 69.22%]	[55.05%, 74.16%]
D — $\hat{\Gamma}$ (recall)	7.03%	31.79%	19.67%

Table 3: Surface-target alignment on ABSA ($n = 980$). Targets: $T_{\text{aspect\_wrong}}$ (aspect_term does not match gold span by fuzzy containment, 18.88% marginal), $T_{\text{polarity\_wrong}}$ (polarity $\neq$ gold, 17.65% marginal), $T_{\text{any\_wrong}}$ (either, 31.12% marginal). *B aligns with the aspect target at* 96.30% *purity (*5.10 × *lift); D aligns with the polarity target at* 59.78% *purity (*3.39 × *lift). Each is far weaker off-target: D drops to* 14.13% *on* $T_{\text{aspect\_wrong}}$*, at or below the* 18.88% *marginal, and B to a weak* 25.93% *on* $T_{\text{polarity\_wrong}}$*, far below its on-target purity. The surfaces specialize.*

Naturalistic data shows sharp target-specific alignment. Surface B's purity flips between 96.30% (aspect target) and 25.93% (polarity target); surface D's flips between 14.13% (aspect) and 59.78% (polarity). The same two surfaces, applied to the same extractions, achieve substantially different target alignments depending on which gold target the rejection set is scored against, the cross-target inversion of §7 reproduced at the structured-output layer on a public naturalistic corpus. The split's direction is structural, each surface examines a different field, so the finding is not that B "discovers" aspect errors but that the confinement

is sharp: B is 5.10 × aligned to aspect and far weaker off-target, D is 3.39 × on polarity and anti-aligned on aspect. The operational consequence is the point: neither surface is a universal correctness monitor, so a pipeline must match the surface to the failure mode it means to catch rather than reach for one universal check.

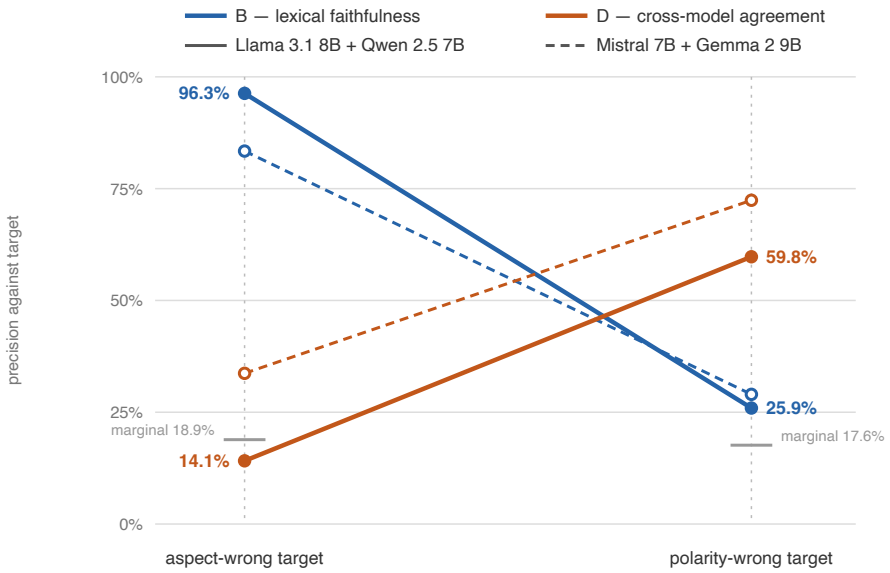

Figure 1: The cross-target inversion as a picture. Each line is one surface's precision, scored against the aspect-wrong target (left) and the polarity-wrong target (right) on SemEval-2014 ABSA ($n = 980$). B falls from 96.3% to 25.9% as the target shifts; D rises from 14.1% to 59.8%. Dashed lines: the Mistral 7B + Gemma 2 9B robustness pair, which shifts the operating points but preserves the crossing. Gray ticks mark each target's marginal rate (18.9%, 17.6%): an unaligned surface would sit there.

Spell-check gloss. A spell-checker catches typos with near-perfect precision and zero precision on subject-verb agreement. A grammar-checker inverts: high on agreement, zero on typos. Same prose, different predicate class, opposite target alignment. The B-vs-D inversion above is the same phenomenon at the structured-output layer.

Cross-surface overlap stays near zero. $B \cap D$ Jaccard $= 0.0318$ with multiset bootstrap 95% CI $[0.0094, 0.0565]$ ($n_{\text{boot}} = 2000$, $\frac{7}{220}$ in intersection/union).

3.2 Robustness pair: Mistral 7B Instruct + Gemma 2 9B Instruct

To check that the cross-target inversion is not specific to Llama 3.1 8B + Qwen 2.5 7B, we re-run the identical ABSA pipeline with a second heterogeneous open-weight pair from different model families. Schema-valid rate remains 100% for both models. Surface rates shift: B fires at 17.24% ($[15.01\%, 19.74\%]$) and D at 10.00% ($[8.28\%, 12.04\%]$). The cross-target inversion preserves with magnitudes that shift but retain ordering: $\hat{\rho}_B$ is 83.4% on $T_{\text{aspect\_wrong}}$ and 29.0% on $T_{\text{polarity\_wrong}}$; $\hat{\rho}_D$ is 33.7% on aspect and 72.4% on polarity. B still specializes for aspect, D still specializes for polarity, with D's polarity-target precision actually higher on this pair (72.4% vs 59.78%). $B \cap D$ Jaccard: 0.0898 $[0.0537, 0.1274]$ ($\frac{22}{245}$), slightly higher than Llama/Qwen but still in the low-overlap regime. *Surface-target alignment preserves across both pairs; operating points are pair-specific.*

Caveats. (a) ABSA gold aspect terms are explicit substrings of the review text; B catches the observed lexical-hallucination rate (13.78% primary pair, 17.24% robustness pair) under this extraction schema, it is not a benchmark on the ABSA leaderboard. (b) D's disagreement rate is specific to each pair and to this extraction schema; we treat D as pair-specific rather than universal. (c) Single-aspect filtering biases the corpus toward shorter snippets; multi-aspect rows would require a schema extension and are deferred.

3.3 Second naturalistic domain: SemEval-2014 laptops

The same benchmark ships a laptop split under the identical `{aspect_term, polarity}` schema, a second naturalistic domain nothing in this study was tuned for. We re-run the identical §3 pipeline on $n = 901$ single-aspect laptop reviews with both open-weight pairs; schema-valid rate stays 100% for all four models. The cross-target inversion reappears in both. For Llama 3.1 8B + Qwen 2.5 7B, $\hat{\rho}_B = 99.2\%$ on aspect and 17.6% on polarity, $\hat{\rho}_D = 34.5\%$ on aspect and 64.5% on polarity ($B \cap D$ Jaccard 0.048 $[0.022, 0.078]$). For Mistral 7B + Gemma 2 9B, $\hat{\rho}_B = 89.0\%$ and 20.7%, $\hat{\rho}_D = 38.1\%$ and 61.9% (same aspect/polarity order; Jaccard 0.073 $[0.046, 0.104]$). B specializes for aspect, D for polarity, and the surfaces stay near-disjoint, reproducing the restaurant result on a second naturalistic domain.

Aspect extraction is harder on laptops: the aspect-error rate is 31–37% versus 19% on restaurants, because technical aspect terms (`USB port`, `track pad`) are easier to drop or paraphrase than restaurant terms like `food` or `service`. That is why D's off-target aspect precision sits in the 34–38% band here rather than near zero, a band already seen on the restaurant Mistral/Gemma pair (33.7%). The specialization directions and the target-driven flip are unchanged across both domains and both pairs.

4 Implementation hierarchy and per-constraint-kind matrix

§3 established that surface-target alignment is sharp on naturalistic data. A separate empirical question, equally central: given the schema-validity target alone (the predicate "the output is a well-formed instance of the declared schema, including types and bounds"), how does the *schema-valid output rate* depend on whether the prompt, API parser, or decoder enforces the schema predicate? Hard-constrained decoding gives a support guarantee only for the provider-compatible structural schema that the implementation actually enforces. It achieves 100% structural validity on that subset by support restriction. This is partly known operational wisdom (OpenAI structured outputs (OpenAI, 2024), Outlines, JSONFormer, LMQL, llama.cpp grammar (ggml-org, 2026) all document it). The novel content here is scope: which constraint kinds the provider-compatible schema dialect actually retains, which it strips, and where the support guarantee therefore ends.

A note on terminology. $\hat{\rho}$ is the empirical purity of a rejection. The implementation-hierarchy table does not measure rejection purity: hard-constrained decoding restricts decoder support so the decoder never emits schema-invalid outputs. We therefore report the *schema-valid output rate* $P\big(\text{output} \in L_\alpha^{\text{complement}}\big)$ rather than $\hat{\rho}$. For hard-constrained decoding under a correctly-implemented grammar, this rate is 1 by support restriction.

4.1 Implementation hierarchy on schema validity

We measure four implementations of the same schema target on a 200-review clean corpus and a 100-review hard corpus (templated synthetic product reviews; corpus details in §5.1), plus a weak-prompt variant that reduces the system instruction to the literal string `"Extract the product review as JSON."`.

Implementation	**Where $\mathcal{D}$ is enforced**	**Clean ($n = 200$)**	**Hard ($n = 100$)**	**Weak prompt ($n = 200$)**
Cell entries are schema-valid output rates (Wilson 95% CI), not $\hat{\rho}$.				
Frontier APIs				
Free decoding — Anthropic	post-hoc validator only	100.00%	100.00%	0.00% [0.00, 1.88%]
Hard-constrained strict tool-use — Anthropic	decoder; structure, types, enum; bound rejected	90.50% [85.64%, 93.83%]	93.00% [86.25%, 96.57%]	94.00% [89.81%, 96.53%]
Free decoding — Gemini	post-hoc validator only	100.00%	100.00%	0.00% [0.00, 1.88%]
Hard-constrained structured output — Gemini	decoder; structure, types, enum	100.00%	100.00%	100.00% [98.12%, 100.00%]
Open-weight models (via Ollama)				
Free decoding — Llama 3.1 8B	post-hoc validator only	97.00% [93.61%, 98.62%]	77.00% [67.85%, 84.16%]	0.00% [0.00, 1.88%]
Hard-constrained (`format=schema`) — Llama 3.1 8B	decoder; llama.cpp grammar	100.00%	100.00%	100.00%
Free decoding — Qwen 2.5 7B	post-hoc validator only	98.50% [95.68%, 99.49%]	90.00% [82.56%, 94.48%]	0.00% [0.00, 1.88%]
Hard-constrained (`format=schema`) — Qwen 2.5 7B	decoder; llama.cpp grammar	100.00%	100.00%	100.00%

Table 4: Schema-valid output rate for the product-review schema-validity target across implementations and corpora. Free decoding collapses under the weak prompt; the Gemini and llama.cpp grammar arms hold at 100% by support restriction; Anthropic strict tool-use falls to 90.5–94.0% because its dialect cannot represent the integer `rating` bound. *Hard-constrained is not sufficient unless the decoder can express the constraint the target requires*—per-arm analysis in the text.

The support-guarantee hierarchy is monotonic, even though observed validity can be model-, prompt-, and corpus-specific. (a) Free decoding + post-hoc validation: schema specification cannot live entirely in the model's prior. Weak prompts collapse to 0 on every free-decoding arm, frontier and open-weight,[3] and even under a detailed prompt free decoding degrades on adversarial inputs (Llama 3.1 8B falls to 77% on the hard corpus, a 23-point gap with its own constrained arm). (b) Hard-constrained API (Gemini structured output, llama.cpp grammar via Ollama): achieves rate 1 across all inputs and prompt strengths by support restriction. The decoder masks tokens that would violate the provider-compatible constraints actually retained by the declared schema; outputs violating those retained constraints are outside the decoder's support. This is the formal support-guarantee regime.[4]

The tier label is not the guarantee. Anthropic `strict: true` tool-use is decoder-masked like regime (b) for structure, types, and enums, but its schema dialect rejects the integer bound, so it cannot mask that constraint—the model emits out-of-range ratings (e.g., 10, 12, −1)—and on this bounded target reaches only 90.5–94%, not rate 1 (table above). The guarantee is exactly the set of constraints the decoder can both accept and mask: llama.cpp masks the integer bound by grammar (ggml-org, 2026); Gemini accepts the bound and

[3]Field-name drift dominates the weak-prompt failures: the model renames `product` to `product_name`, wraps the object in markdown fences, or nests fields the schema does not declare.

[4]The support guarantee is conditional on the decoder implementation, not a free property of the API: a bug in the grammar engine, a token-boundary edge case, or a provider downgrade to soft-constrained behavior would invalidate it. We treat the implementation as a trusted component, as is standard when reporting on hard-constrained decoding.

the model complies, though it is not decoder-enforced (Google, 2026) (§4 schema acceptance); Anthropic strict cannot represent the bound at all (Anthropic, 2026). *Hard-constrained* names the mechanism, not the covered constraint set.

A distinct tier: the API parser. Between prompt-only enforcement and decoder-level masking sits a third, widely-deployed option that is not a support guarantee at all: the soft-constrained tool-use API (non-strict Anthropic tool-use; the mode most function-calling integrations use by default). Here the schema is enforced by the provider's API parser *after* generation, not by the decoder *during* it, and the failure mode is characteristic and distinct from both neighbours. The parser eliminates field-name drift outright —the schema rides in the tool definition, so the arm is 100% structurally valid on the clean corpus and prompt-independent (100% under the weak prompt, where free decoding collapses to 0): schema *location*, not prompt strength, is what carries the constraint. But the parser checks structure and primitive type only, never values: on the adversarial hard corpus the model emits placeholder strings (`"<UNKNOWN>"`) in fields the schema declares `int` / `bool`, and the API relays them as structurally well-typed (91% schema-valid). The API parser is thus a genuine middle of the enforcement spectrum—stronger than the prompt (no drift, prompt-independent), weaker than the decoder (it cannot reject a structurally-valid but semantically-wrong value) —and the empirical realization of the framework's central axis: *where* the predicate is enforced, prompt → parser → decoder, is a gradient, not a switch. Its per-constraint profile (structural 100%; pattern and enum largely retained; numeric bounds and cross-field consistency not) is in the matrix of §4.[5] The practitioner consequence is concrete: soft-constrained tool-use removes the need to name fields in the prompt, but a post-hoc validator on values remains mandatory.

4.2 Type-checked function calls

The hierarchy reproduces on a typed function-call task: five tool signatures (`book_flight`, `set_thermostat`, `create_calendar_event`, `transfer_money`, `send_message`) with regex patterns, numeric bounds, and required fields, 95 natural-language requests. Non-strict tool-use modes eliminate JSON-decode failures and primitive-type violations but leave residual failures concentrated on regex-pattern violations these API modes do not enforce as tested (Anthropic non-strict: $\frac{91}{95}$; Gemini function-calling: $\frac{88}{95}$, all 7 failures on `pattern` constraints). The per-constraint-kind matrix below pins this down precisely.

4.3 Per-constraint-kind enforcement matrix

A single schema-valid rate hides which kinds of constraint a provider actually enforces; the matrix below separates them, so a practitioner can read off exactly what survives the decoder and what still needs a post-hoc validator. The implementation hierarchy reports a single rate on structural schema validity; the function-call task reports an aggregate `valid-call` rate. We now extend both to a *per-constraint-kind* enforcement matrix across six implementations and five constraint kinds (structural, pattern, enum, numeric/string/array bounds, cross-field consistency). This corpus is the 95-prompt function-call corpus extended to $n = 169$ with 74 hand-authored, constraint-tagged prompts: valid controls plus cases that each deliberately violate a single constraint kind (pattern, enum, numeric/string/array bounds, or cross-field). It is a controlled diagnostic, not a naturalistic benchmark—the deliberate per-kind violations are what make each kind's enforcement (or non-enforcement) observable. Schemas are the five tools, augmented with explicit enum fields (e.g., `book_flight.class ∈ {economy, business, first}`).

Two provider arms, Anthropic strict and Ollama grammar, accept different JSON-Schema dialects, so their provider-compatible schemas were generated by stripping unsupported keywords. Anthropic strict retains structure, pattern, enum, and string-length bounds, but strips numeric/integer and array bounds (the API rejects them outright—HTTP 400 *for `integer` type, properties `maximum, minimum` are not supported* —so the §4 strict-tool-use arm runs on the stripped schema); Ollama grammar retains structure, enum, and most bounds, but strips all `pattern` keys. Provider expressivity is therefore reported as a separate "schema acceptance" table.

[5]The soft-constrained arm's per-constraint breakdown is reported in the schema-acceptance and per-constraint-kind matrix below; here we report only its single schema-validity rate to place it on the enforcement spectrum.

Implementation	Structural	Pattern	Enum	Bounds	Cross-field
Free decoding + post-hoc (Anthropic)	76.9% (130/169)	66.7% (18/27)	45.5% (10/22)	41.4% (12/29)	33.3% (7/21)
Anthropic non-strict tool-use	100.0% (169/169)	85.2% (23/27)	90.9% (20/22)	48.3% (14/29)	33.3% (7/21)
Anthropic `strict: true` tool-use	100.0% (169/169)	100.0% (27/27)	100.0% (22/22)	44.8% (13/29)	33.3% (7/21)
Gemini function calling / structured tool schema	99.4% (168/169)	48.1% (13/27)	100.0% (22/22)	89.7% (26/29)	33.3% (7/21)
Ollama Llama 3.1 8B + grammar	100.0% (169/169)	51.9% (14/27)	100.0% (22/22)	89.7% (26/29)	28.6% (6/21)
Ollama Qwen 2.5 7B + grammar	100.0% (169/169)	51.9% (14/27)	100.0% (22/22)	89.7% (26/29)	33.3% (7/21)

Table 5: Per-constraint-kind emitted schema-validity rate. Each cell is the fraction of prompts for which the emitted tool call satisfies the constraints of the named kind that the prompt's reference schema specifies, with raw counts in parentheses. Cross-field prompts contain 7 valid controls and 14 invalid cases; the common $\frac{7}{21}$ result is the valid-control baseline, not evidence of decoder-level cross-field enforcement. The matrix is a dated snapshot (May 2026: `claude-haiku-4-5`, `gemini-2.5-flash`, `llama3.1:8b` / `qwen2.5:7b` via Ollama llama.cpp grammar); provider enforcement changes over time.

Schema variant	Overall	Structural	Pattern	Enum	Bounds	Cross-field
Anthropic `strict: true`-compatible	80.0% (32/40)	100.0% (20/20)	100.0% (7/7)	100.0% (3/3)	28.6% (2/7)	0.0% (0/3)
Ollama llama.cpp grammar	72.5% (29/40)	100.0% (20/20)	0.0% (0/7)	100.0% (3/3)	85.7% (6/7)	0.0% (0/3)

Table 6: Provider schema-acceptance rate: of the 40 JSON-Schema constraints in the full 5-schema reference set, how many are retained in each provider-compatible schema variant. Only Anthropic strict (which 400s on unsupported keywords) and the Ollama grammar (which crashes on pattern) require a stripped variant; Gemini structured output, free decoding, and non-strict tool-use accept the full reference schema unchanged, so they do not appear here — their gap is enforcement, not acceptance (per-constraint matrix). Anthropic strict retains $\frac{2}{7}$ bound fields, both string-length fields; numeric/integer bounds are entirely stripped. Ollama llama.cpp grammar mode accepts bounds but not regex patterns. Neither provider accepts cross-field constraints in the schema dialect.

Schema acceptance and emitted validity are separate axes. Anthropic strict can be evaluated on all 169 prompts because the sent schema is provider-compatible for every prompt, but the provider-compatible schema is not the full reference: it retains pattern and enum, while numeric/integer bounds are stripped. That is why the strict row can show 100.0% pattern/enum but only 44.8% full-reference bounds. Conversely, Ollama's 51.9% pattern cell is post-hoc full-reference pattern validity despite the grammar schema containing no `pattern` keys; it is not evidence of grammar-level regex enforcement. The same separation applies to numeric bounds. Gemini's structured-output dialect accepts `minimum`/`maximum`, but Google's documentation (Google, 2026) guarantees syntactic JSON conformance while explicitly directing client-side validation of values; Gemini's bounds cell is therefore emitted validity, not decode-time enforcement. llama.cpp's grammar compiler (ggml-org, 2026), by contrast, compiles integer `minimum`/`maximum` into the GBNF grammar (integer types only; `number` ranges and `multipleOf` are not expressed), so Ollama's integer-bound enforcement is genuinely decoder-level. Numeric bounds thus land at three different tiers across the hard-constrained arms: stripped then post-hoc-validated (Anthropic strict), accepted but unenforced (Gemini), and decoder-enforced for integers (Ollama grammar).

No implementation tested supplies decoder-level cross-field support. Cross-field constraints such as "origin and destination must differ" or "event start must not be in the past" are not expressible in these schema dialects. The observed $\frac{7}{21}$ results are the valid-control baseline by construction. This matches §8′s prediction that schema/grammar/type families fail by specification gap on constraints the formal language cannot express.

The matrix is provider-specific, not a single hard-vs-soft ladder. The structural hierarchy reproduces (schema-carrying arms $\geq 99\%$ structurally valid, free decoding 76.9%). The new result is that non-structural guarantees split by provider and keyword: among the arms tested, Anthropic strict is strongest on retained pattern/enum; on bounds, Ollama's integer enforcement is decoder-level while Gemini's high rate is emitted, not enforced (above); Anthropic non-strict's non-structural validity is empirical rather than decoder-masked; and no arm tested covers cross-field consistency.

4.4 Assertion-with-retry as oracle channel

A small oracle-channel demonstration completes the implementation menu. We classify each of 75 short utterances into one of three intents (`book` / `cancel` / `reschedule`). The system prompt names the valid set without bolding the constraint; the post-hoc predicate is exact-string match on the lowercase-stripped output. On surface failure, we issue a corrective user turn and re-sample, up to three retries.

74 of the 75 cases pass the predicate on the first attempt (98.67%, Wilson 95% [92.83%, 99.76%]); a single corrective turn recovers the lone failure, lifting the valid-output rate to $\frac{75}{75}$ (100%, [95.13%, 100.00%]). No case needed more than one retry (mean 1.013 attempts per case).

This is the flagship paper's oracle channel pattern: every escape from the surface predicate is routed back through a corrective turn. But the convergence to 1 is empirical, not formal $\rho = 1$—it depends on the corrective channel landing the next sample inside the predicate's preimage, and on the retry budget. The result is the empirical support/repair regime typical of a revision pipeline, and the oracle-channel classification with an empirical validity rate is the right one for this method family.

5 Templated cross-class surfaces across domains, and a monitor comparison

§3 established that verification surfaces are target-specific: structurally different checks over the same extractions catch different mistakes. §4 established a limit: cross-field consistency is the one constraint kind a schema can assert that no decoder can enforce, since a JSON-Schema or grammar dialect cannot express a relation between two fields. §5 joins the two. It adds surface C, the cheap post-hoc check for exactly that constraint, on a review-extraction schema rich enough to carry a cross-field relation (rating↔recommend); §3′s `{aspect_term, polarity}` schema has none, which is why C could not be tested there.

C is not interesting because it works. A five-line predicate trivially flags a rating-recommend contradiction; the question is when running it is worth the trouble. The one constraint no decoder can enforce is also one that models usually satisfy on their own, so C is a guard that mostly stays silent. Whether it earns its place is a triage question in two parts: does C fire, which tracks input difficulty and model quality; and when it fires, is the contradiction a real error, which tracks the domain. Surfaces B and D carry over from §3 as the same classes on the review schema and reproduce the §3 specialization here. The section closes with a separate, practical result: whether a cheap deterministic monitor can stand in for an expensive LLM judge on D's target.

Scope note. §5 is not a benchmark of language-model capability. This view's claim is about verification-surface design: given a model-task pair with measurable natural failures, an appropriately chosen surface rejects those failures with measurable purity, and structurally different surfaces over the same outputs catch structurally different failures. Open-weight models give a measurable failure regime on the corpora used here.

5.1 Setup

Corpus. A 1000-review clean corpus generated from a fixed set of fourteen review templates (seed = 42; the original $n = 200$ result reproduces under the scaled run) and a 100-review hard corpus of adversarial / minimal / sarcastic snippets (seed = 99). Each clean review carries a deterministic gold annotation (seed = 42), source product, rating $\in [1, 5]$, recommend boolean, and verdict polarity, used as target ground truth without human labels. The corpus is synthetic and dated to the paper's preparation; it postdates every public model's training cutoff, ruling out benchmark leakage.

Models. The cross-class results below use an *open-weight pair*, `llama3.1:8b` and `qwen2.5:7b`, both via Ollama, chosen because their natural failure rates fall in a few-percent-to-double-digit regime where surface alignment is measurable.

5.2 Three cross-class surfaces

§3 ran two of these surfaces, B and D, on naturalistic data. A surface is defined by its predicate class, not a fixed field, so the same two classes carry over to the review schema with task-appropriate predicates: B matches the product against the input where §3 matched the aspect term, and D compares rating and recommend where §3 compared polarity. Here we also add a third, C: the post-hoc deterministic check for the cross-field consistency constraint §4 showed no decoder can enforce. Each surface is a verification surface in this approach's sense: a representation Y, a deterministic predicate D, and an implicitly aligned target L.

Surface B, Lexical faithfulness. $Y_B = (\text{input\_text}, \text{extracted.product})$; D_B rejects iff the extracted product does not appear in the input text (case-insensitive substring or ≥ 0.5 token overlap). Catches hallucinated product names. Aligned target L_B: extracted product is not lexically supported by the source text.

Surface C, Cross-field consistency. $Y_C = (\text{extracted.rating}, \text{extracted.recommend})$; D_C rejects iff `(rating ≥ 4 AND recommend != True)` or `(rating ≤ 2 AND recommend != False)`. Catches rating/recommend contradictions. Aligned target L_C: extraction is internally inconsistent.

Surface D, Cross-model disagreement. $Y_D = (\text{extraction}_A, \text{extraction}_B)$ for two distinct models; D_D rejects iff `|rating_A - rating_B| > 1` OR `recommend_A != recommend_B`. Catches cross-model disagreement. Aligned target L_D: cross-model disagreement, or low-consensus extraction.

Y_B, Y_C, Y_D are structurally different representations: B operates over (input text, output substring), C over a $5 \times \{0, 1\}$ joint variable, D over a pair of full structured outputs.

5.3 Per-target $\hat{\rho}$ on the clean corpus (open-weight pair)

On the clean corpus the surfaces show their ordinary operating regime: which failure modes the model actually makes, and which surface aligns with each. For the primary Llama / Qwen pair, B fires rarely and C not at all (clean reviews contain no rating/recommend contradictions to catch), while D carries most of the signal. There are four ground-truth targets against the gold annotations: T_{product} = product wrong, T_{rating} = rating $\neq$ gold, $T_{\text{recommend}}$ = recommend $\neq$ gold, T_{any} = any of the three wrong. The cross-class surfaces operate over Llama 3.1 8B's hard-constrained extractions; surface D compares against Qwen 2.5 7B as the second extractor.

Surface	Rejections	Rate	Wilson 95% CI
B (lexical faithfulness)	$\frac{3}{1000}$	0.30%	[0.10%, 0.88%]
C (cross-field consistency)	$\frac{0}{1000}$	0.00%	[0.00%, 0.38%]
D (cross-model disagreement)	$\frac{141}{1000}$	14.10%	[12.08%, 16.39%]

Table 7: Per-surface rejection counts over Llama 3.1 8B hard-constrained extractions on the 1000-review clean corpus (scaled from $n = 200$). B fires rarely (3 hallucinations); C never fires (the template generator does not produce rating/recommend contradictions); D fires where Llama and Qwen disagree on 14.1% of cases, the visible part of the dominant failure mode (rating-extraction uncertainty); most rating errors are silent, both models inferring the same wrong rating.

Two counts are in play here, and they are not the same: how often a surface fires (the rejection rates above) and how often the extraction is actually wrong (it disagrees with the gold annotation). The task is hard by design. Each review's gold rating is a uniform draw from $\{1, 2, 3, 4, 5\}$, but only two of the review templates state it as a number (e.g., "rating around 4/5", "would rate it 4 stars"); for the rest the model must infer the rating from tone, and it gets the rating wrong on 48.5% of reviews, recommend on 18.3%, product on 2.3%, and at least one field wrong (T_{any}) on 57.2%. The surfaces catch only a slice of that: D fires on the 14.1% of cases where Llama and Qwen disagree, but the two models often read the same tone the same wrong way, so most rating errors are silent and slip past both. Sparse firing on a task with a 57.2% error rate is a coverage limit, not an easy task; that gap, between how much is wrong and how much each surface catches, is what the purity and recall table below measures.

Surface vs. target	T_{product}	T_{rating}	$T_{\text{recommend}}$	T_{any}
B (lexical) — $\hat{\rho}$	100.00%	100.00%	66.67%	100.00%
B — $\hat{\Gamma}$ (recall)	13.04%	0.62%	1.09%	0.52%
C (cross-field) — $\hat{\rho}$	—	—	—	—
D (cross-model) — $\hat{\rho}$	4.96%	48.23%	46.81%	75.18%
D — $\hat{\Gamma}$ (recall)	30.43%	14.02%	36.07%	18.53%

Table 8: $\hat{\rho}$ and $\hat{\Gamma}$ for each surface against each target on $n = 1000$. Surface B's three rejections are high-purity against the gold targets (100% on T_{product}, T_{rating}, T_{any}; 66.7% on $T_{\text{recommend}}$); coverage is correspondingly near zero. C never fires (purity undefined). D's purity is highest against T_{any} (75.2%, $\frac{106}{141}$ rejections coincide with at least one gold-violation) and against T_{rating} (48.2%, the dominant natural failure); near-zero alignment with T_{product} (4.96%, expected, Llama and Qwen rarely disagree on product). Marginal target rates: $T_{\text{product}} = 2.30\%$, $T_{\text{rating}} = 48.50\%$, $T_{\text{recommend}} = 18.30\%$, $T_{\text{any}} = 57.20\%$.

Different surfaces align with different targets. Purity is alignment, not determinism: D deterministically flags disagreement, but disagreement only proxies a gold error (Llama and Qwen can differ while Llama is right), so D's rejections coincide with the gold targets only in part. Surface B's purity is 100% on the product target at the cost of low coverage; D's purity is 75.2% on T_{any}, a 1.31 × lift over the 57.2% marginal. C does not fire on the clean corpus, so it has no purity to report, but the zero is a result, not a blank. The model's wrong extractions stay internally consistent: even when Llama gets the rating wrong, it keeps rating and recommend aligned, so a contradiction check has nothing to catch. That is the easy end of the difficulty axis §5.5 develops, and weaker pairs or harder inputs (§5.4, §5.5) are where C begins to show relevance.

Swapping the extractor pair changes which failures occur. On the same corpus, Mistral 7B + Gemma 2 9B fire B at $\frac{2}{1000}$ (0.20%) and D at $\frac{37}{1000}$ (3.70%, versus 14.10% for Llama/Qwen, since the two models agree more often), and, unlike Llama/Qwen, produce 11 rating-recommend inversions that activate C at 100% purity against T_{any} (the weak-pair row of §5.5). Failure modes depend on the model that produces them, not just the inputs.

This model-dependence is itself a signal. The surface firing rates here and the pairwise rejection-set overlaps of §5.4 and §5.5 (the overlap ratio ω of §2) both move with the extractor pair. This is the label-free model-quality reading of ω: the flagship formalizes it ((Rothrock, 2026a), with ω rising as model quality falls) and the medical companion (Rothrock, 2026b) realizes it across segmentation models, where the same construction doubles as a label-free model-quality benchmark.[6]

5.4 Pairwise Jaccard between rejection sets (hard corpus)

The main cross-class test asks: when all three surfaces actually fire, do they reject the same cases or different ones? The clean corpus could not answer that, since C and B barely fired there, so we use the 100-review hard corpus, where adversarial / minimal / sarcastic inputs make the open-weight models fabricate product names (B fires), introduce rating/recommend contradictions (C fires), and disagree between the two extractors (D fires).

Surface	Rejections	Rate	Wilson 95% CI
B (lexical faithfulness)	$\frac{28}{100}$	28.00%	[20.14%, 37.49%]
C (cross-field consistency)	$\frac{15}{100}$	15.00%	[9.31%, 23.28%]
D (cross-model disagreement)	$\frac{13}{100}$	13.00%	[7.76%, 20.98%]

Table 9: Per-surface rejection counts on the 100-review hard corpus, Llama 3.1 8B hard-constrained extractions, Qwen 2.5 7B comparator. All three surfaces fire substantially because the corpus exercises all three failure modes.

[6]We establish here only the LM-layer model-dependence the reading rests on; calibrating ω to a graded LM-quality axis would need a quality-ordered extractor sweep, which we leave to follow-on work.

	B	C	D
B	1.000	0.000 [0.000, 0.000]	0.171 [0.054, 0.300]
C	0.000 [0.000, 0.000]	1.000	0.037 [0.000, 0.120]
D	0.171 [0.054, 0.300]	0.037 [0.000, 0.120]	1.000

Table 10: Pairwise Jaccard between rejection sets on the hard corpus, with multiset percentile bootstrap 95% CIs ($n_{\text{boot}} = 2000$). $B \cap C = 0.000$ (zero joint rejections of 43 in union), $C \cap D = 0.037$ (1 of 27, CI includes zero), $B \cap D = 0.171$ (6 of 35, CI excludes zero). Hallucinated products coincide with cross-model uncertainty on a non-trivial fraction of cases, consistent with confused inputs both making the model invent values *and* making the two models diverge.

Structurally different surfaces fire on different cases. The naturalistic ABSA result (§3) at $n = 980$ gives a much tighter $B \cap D = 0.032$ $[0.009, 0.057]$ for the same two surface classes on that task.

5.5 When does C add value? Difficulty, domain, and cross-class robustness

C guards a failure mode that is both rare and decoder-inexpressible, so whether it is worth running is a triage decision in two independent parts: whether C fires, and whether its firings are real errors.

Does C fire? On natural input from a capable pair, cross-field contradictions essentially do not occur; they surface only under a weaker model or adversarial input.

Regime	**Corpus (n)**	**C fires**	**Rate (Wilson 95%)**
Strong pair, natural	product clean, Llama / Qwen (1000)	0	0.00% [0.00%, 0.38%]
Weak pair, natural	product clean, Mistral / Gemma (1000)	11	1.10% [0.62%, 1.96%]
Strong pair, adversarial	product hard, Llama / Qwen (100)	15	15.00% [9.31%, 23.28%]

Table 11: Surface C firing rate by regime. Natural input from a capable pair produces no cross-field contradictions; they appear only under a weaker pair or adversarial input.

When C fires, is it a real error? A contradiction is a gold error only where the domain's cross-field relation is load-bearing for correctness. We exercise C deliberately across four domains, each a stress corpus of four 25-case quota buckets (lexical-distractor, internal-consistency-trap, cross-model-ambiguity, easy filler), all extracted by the same Llama 3.1 8B + Qwen 2.5 7B pair (schema-valid rate = 100% across 1800 extractions).

Domain	**Relation**	**C fires**	**∩ gold**	**Purity $\hat{\rho}_C$ (Wilson 95%)**
Restaurant	rating → recommend	25	0	0% [0%, 13%]
Medical	severity → action	36	11	31% [18%, 47%]
Support	severity → action	25	21	84% [65%, 94%]

Table 12: Surface C purity against the gold decision target on each domain's stress corpus (C deliberately exercised). A contradiction is a gold error only where the domain's relation is materially significant.

In support, where severity must drive action, a contradiction is a genuine mistake (84%); in restaurant, the rating/recommend relation is half-decorative, so a contradiction does not coincide with a gold field-error (0%). The predicate is identical across domains; whether violating it means "wrong" is a property of the domain, not the check. C's value is conditional on both axes: it earns deployment where contradictions occur (hard inputs, weaker models) and where the relation is materially significant.

The decomposition still holds across domains. Independent of C's hit rate, the three surfaces stay near-disjoint in every domain, and B and D keep their §3 specialization. Every rejection lands on the aligned field: all of D's rejections are gold errors on the decision target (restaurant $\frac{6}{6}$, support $\frac{24}{24}$, medical $\frac{17}{17}$) and all of B's are gold errors on the entity-naming field (restaurant $\frac{5}{5}$, support $\frac{13}{13}$, medical $\frac{31}{31}$), giving $\hat{\rho} = 1$ in-sample for both. These counts are small, so this is target alignment in every domain, not a zero-false-positive guarantee: neither predicate is pure by construction.

Domain ($n = 100$ hard)	B ∩ C	B ∩ D	C ∩ D	Max cross-class
Restaurant reviews	0.000 [0.000, 0.000]	0.000 [0.000, 0.000]	0.000 [0.000, 0.000]	0.000
IT support tickets	0.000 [0.000, 0.000]	0.194 [0.063, 0.343]	0.000 [0.000, 0.000]	0.194
Synthetic medical notes	0.047 [0.000, 0.105]	0.143 [0.048, 0.263]	0.128 [0.044, 0.233]	0.143
Product reviews (baseline)	0.000 [0.000, 0.000]	0.171 [0.054, 0.300]	0.037 [0.000, 0.120]	0.171

Table 13: Pairwise Jaccard of B/C/D rejection sets on each domain's hard corpus ($n = 100$), multiset percentile bootstrap 95% CIs ($n_{\text{boot}} = 2000$). Maximum cross-class Jaccard across all four domains is support's $B \cap D = 0.194$ $[0.063, 0.343]$, comparable to the baseline's 0.171. No cross-class Jaccard exceeds 0.20 at the point estimate. Medical's $B \cap C = 0.047$ CI $[0.000, 0.105]$ includes zero.

§3′s two-surface specialization thus becomes a three-surface decomposition across all four domains tested here, with C the conditional third member whose value depends on the two axes above.

Composing the aligned surfaces. The per-surface purities above are a composition recipe: the union of zero-observed-FP, low-overlap surfaces preserves zero observed false positives in-sample, and their recalls combine by inclusion-exclusion (adding when the overlap is small), so composing the aligned surfaces raises coverage at preserved purity. Against the any-error target, B (entity) and D (decision) have no false positives in-sample, so composing them both-rises in every domain.

Hard domain ($n = 100$)	Best single ($\hat{\Gamma}$)	$B \cup D$ recall	$B \cup D$ purity
Medical (any-error 50%)	B, 62.0%	84.0%	100% (0 FP / 42)
Restaurant (any-error 49%)	D, 12.2%	22.4%	100% (0 FP / 11)
Support (any-error 72%)	D, 33.3%	43.1%	100% (0 FP / 31)

Table 14: Composing the two zero-FP surfaces ($B \cup D$) on each hard domain's any-error target. Union recall is the disjoint-union recall ($|B| + |D| - |B \cap D|$)/ |any-error|; coverage rises in every domain while purity stays at zero observed false positives. Best single is the higher-recall of B and D. $n = 100$ per domain, so purity here is zero observed FP, not a proof of $\hat{\rho} = 1$.

The union catches more at no observed cost to purity: B's 62.0% recall rises to 84.0% in medical, D's 12.2% to 22.4% in restaurant, and D's 33.3% to 43.1% in support, with zero false positives throughout. A third surface earns inclusion only where it is itself aligned: in support, where C is high-purity, $B \cup C \cup D$ lifts coverage to 73.6% at 94.6% purity; in restaurant, where C fires only false positives (0% purity above), adding it collapses union purity to 30.6%, and medical is the same pattern. The selection rule is the measurement: compose a surface iff it is pure on the target at hand. These are small hard corpora ($n = 100$) and zero-FP means zero observed false positives, not a proof; the demonstration is of the technique, a both-rise wrung from surfaces a pipeline already runs, not a general guarantee.

5.6 Monitor comparison vs LLM-judge and self-consistency baselines

On this templated monitoring target, a cheap deterministic check matches an expensive LLM judge, and which monitor wins shifts with the specific error type. We compare the deterministic structural surface D against two standard runtime-monitoring baselines on the same target family:

- *LLM-as-judge* (Zheng et al., 2023): Claude Sonnet 4.6, gold-blind, ranking per-field error probabilities.
- *Self-consistency instability monitor* (within-model; cf. (Wang et al., 2023) for the underlying multi-sample idea): 10 additional Llama 3.1 8B re-samples at $T = 0.7$, ranked by a weighted instability composite rather than the original majority-vote procedure.

Both baselines are rate-matched to D's native 14.5% rate (145/1000) on an independent 1000-case templated corpus, a five-seed regeneration distinct from the §5.3 corpus (where D fires at 14.1%).

Aggregate target. On the headline target $T_{\text{any\_observable}}$ (29.10% marginal), the deterministic surface leads on the point estimate:

- D: 68.28% precision (2.35 × lift, Holm $p = 5.6 \times 10^{-25}$).
- Judge: 54.48% (1.87 × lift, Holm $p = 9.6 \times 10^{-11}$).

- Self-consistency: 52.41% (1.80 × lift, Holm $p = 4.0 \times 10^{-9}$).

Pairwise Holm-corrected tests do not separate D from the judge or self-consistency on this aggregate target.[7]

Target-specific separations. Two are Holm-significant: D exceeds self-consistency on recommendation errors (Holm $p = 0.027$), and self-consistency exceeds the judge on product hallucinations (Holm $p = 0.039$). So the "different surfaces, different alignments" prediction holds at the monitor-vs-monitor level in the weaker form: target-specific preferences exist and are not noise. Because these monitors catch different errors, combining them would add coverage rather than pick the single winner the head-to-head comparison here selects. That additive, one-stage gain is demonstrated for the deterministic surfaces in §5.5; the multiplicative, cross-stage case is the complementarity result of the flagship and medical companions (Rothrock, 2026a; 2026b).[8]

Operational reading. The headline is a cost result. A frontier LLM judge that reads each review did not outperform cross-model disagreement, a content-blind signal that needs only a second extractor. At a matched alert budget the deterministic surface is ahead on the point estimate (68.3% versus 54.5%) and, after correction, statistically indistinguishable from the judge: the paid monitor bought no measurable edge on this target. The claim is bounded (one target, $n = 1{,}000$, underpowered to separate the monitors), so it is not "deterministic beats judge" but the weaker, more useful one: where a cheap structural surface already aligns with the failure mode, escalating to an expensive semantic monitor is not obviously worth it. The cost asymmetry makes the choice concrete. If a second extractor already runs, for ensembling or redundancy, its disagreement is a monitor at no extra call. And even as a fresh addition it is usually the cheaper option: extraction is a narrower task than gold-blind judging and runs on a smaller model (here, open-weight extractors against a frontier judge), so a second extractor buys a monitor cheaper than a judge and, on this target, no worse. Full methodology and per-target breakdown are in the Supplemental Analysis.

5.7 Aggregate reading

§5 produced two results. First, three structurally distinct checks (B lexical, C cross-field, D cross-model) decompose structured-output failures into low-overlap rejection sets (maximum cross-class Jaccard 0.194 across four domains), so no single one is a universal correctness monitor. C is the post-hoc surface for the cross-field constraint §4 showed no decoder can enforce, and its utility is qualified: it fires only under hard inputs or weaker models, and its firings are real errors only where the domain's cross-field relation is load-bearing. Second, on D's target a cheap deterministic disagreement gate matched an expensive LLM judge and a self-consistency monitor at a fixed alert budget, with no measurable edge for the paid judge. Both turn on this view's diagnostic, *which (surface, target) pairs align*, not on a search for a single universal score.

6 Token-level diagnostic surfaces

The structured-output results in §3-§5 demonstrate that surface choice determines target alignment at the JSON-extraction layer. The token-emission layer tests the same claim in the paper's most controlled measurement regime: four runtime-deployable structural gates over 3,000,000 candidate-token records, plus two teacher-forced gold-NLL evaluation gates on a matched held-out slice, all from controlled transformer seeds, with gold next-token labels and matched negative controls.

The gates here are deliberately unsophisticated, just simple deterministic predicates over the ambient token-emission representation. That is the test: if which errors a check catches is set by its structure (what it

[7]Power note: the aggregate D-vs-judge comparison has 74 discordant cases with an observed 47 : 27 split; separation at the Holm-adjusted threshold requires roughly 51 : 23, and 80% power for a split of the observed size requires on the order of 200 discordant cases, $n \approx 3{,}000$ at the observed discordance rate. Non-separation at $n = 1{,}000$ bounds the test's power; it is not evidence of equivalence.

[8]On these clean extractions the non-D surfaces rarely fire (cf. §5.3), so composing them with D adds little here; surfaces compose where their failure modes occur, on the hard corpus, where §5.5 shows B and D both-rise at preserved purity and a misaligned surface dilutes.

looks at and the rule it applies), not by detector sophistication, the simplest predicates should already show it. They do, and three matched negative controls confirm the alignment is structural rather than threshold noise, the cleanest such control in this paper. The section then compares gate overlap across the token and structured-output layers.

6.1 Corpus, models, surfaces, and gates

Corpus. A deterministic next-token-prediction trace over FineWeb SP1024 (Penedo et al., 2024) (vocabulary = 1,024 SentencePiece BPE tokens). Per token: previous token, candidate argmax, reference argmax, rank of candidate prediction under reference. Calibration uses 300,000 disjoint positions. Evaluation uses 3,000,000 positions (1,000,000 per candidate seed). Bootstrap CIs use 10,000 case-resampling iterations (seed = 42). Binomial intervals use Wilson 95%.

Models. Reference: a 6-layer transformer with $d = 448$ and sequence length 4,096. Training used seed = 99 for 30,000 steps on FineWeb SP1024. Candidates: three architecturally identical transformers at seeds = 7, 42, 123. We trained the reference and candidates from identical configurations and changed only the random seed. This design minimizes distributional drift, so rank-corruption captures genuine candidate-vs-reference disagreement rather than systematic architectural mismatch.

Six gates in two structural families. *Family 1, Structural anomaly* contains four gates over candidate argmax, with no model logits:

1. `bigram_low_prob`, reject if train-only smoothed bigram $P_{\text{bi}}(\hat{y} \mid \text{prev})$ falls below calibrated ε
2. `first_byte_mismatch`, reject if first-byte class is not train-observed for the prior token at class_min_prob mass
3. `token_length_mismatch`, reject if byte-length bin lies in a length_tail_prob tail of the conditional length distribution
4. `word_boundary_mismatch`, reject if word-boundary class is not train-observed for the prior token

Family 2, Gold-NLL contains two gates consuming the per-token NLL (negative log-likelihood) on the actual continuation, i.e. *teacher-forced*: the model is scored against the true next token from the corpus, information that exists only after the fact:

5. `model_surprise`, reject if candidate NLL on gold next token exceeds τ_{surprise}
6. `cross_model_nll`, reject if $|\text{NLL}_{\text{s42}} - \text{NLL}_{\text{s7}}| > \tau_{\text{disagree}}$

Each family uses a structurally distinct representation map (candidate argmax plus train-derived lookup tables for Family 1, NLL-of-target stream for Family 2). In the formal sense of §2 each family is a single verification surface (one representation map and its predicate class), and the six checks above are the deployed gates over them: four on Family 1, two on Family 2. Cross-family overlap therefore measures cross-surface overlap; within-family overlap measures cross-gate overlap on one surface.

Deployability axis. The two Family-2 gates use the candidate's NLL on the actual continuation, unavailable at autoregressive generation time. They are *evaluation gates*: valid for measuring gate-target alignment on a held-out corpus, but not deployable as online verifiers. Family 1 gates are deployable. We additionally instrument four runtime-deployable candidate-side analogues: `low_top1_prob`, `high_entropy`, `low_margin`, `cross_model_argmax`. The four are not interchangeable with the gold-using Family-2 gates (pairwise Jaccard ≤ 0.005 between deployable and gold-NLL analogues at their respective operating points), but they preserve target-specific alignment at runtime; the Supplemental Analysis reports the per-gate detail.

Calibration. We select the bigram threshold ε on the calibration positions to maximize Wilson lower-bound precision under a target rejection budget. The selected operating point rejects 0.241% of calibration positions with calibration $\hat{\rho} = 0.97\%$ on the rank > 128 target (base rate $\approx 0.025\%$, roughly $39\times$ lift). We calibrate the other Family-1 gates similarly. We calibrate Family-2 thresholds on a disjoint 100K-position slice to match `bigram_low_prob`'s rejection rate. We freeze all thresholds for evaluation.

Targets. Two families: reference-disagreement (rank > 64, rank > 128, rank > 256, and exact-mismatch, model-vs-model on matched architecture and seed) and gold-correctness (gold_top1_wrong, candidate top-1 differs from corpus continuation; literal top-K and gold-NLL severity targets in §7). **Reproducibility.** mixlab `-logprobs-out` mode is public (github.com/mrothroc/mixlab). Exact commands and configs are in the reproducibility materials.

6.2 Primary results and negative controls

The four Family-1 gates read the same verification surface, the candidate token against training statistics, and differ only in their predicate: bigram probability, first byte, byte-length, or word boundary. We run all four against one target, rank-disagreement (rank > 128), to see what simple deterministic gates extract from the ambient token representation.

Gate	$\hat{\rho}$	$\hat{\rho}$ Wilson 95% CI	$\hat{\Gamma}$	Rejection rate	TP / FP / FN / TN
`bigram_low_prob`	1.197%	[0.958%, 1.496%]	10.284%	0.212%	76 / 6,271 / 663 / 2,992,990
`first_byte_mismatch`	0.084%	[0.048%, 0.147%]	1.624%	0.476%	12 / 14,265 / 727 / 2,984,996
`token_length_mismatch`	0.000%	[0.000%, 0.450%]	0.000%	0.028%	0 / 850 / 739 / 2,998,411
`word_boundary_mismatch`	0.129%	[0.072%, 0.230%]	1.488%	0.285%	11 / 8,532 / 728 / 2,990,729
`combined_union`	0.322%	[0.264%, 0.394%]	12.855%	0.982%	95 / 29,367 / 644 / 2,969,894

Table 15: The four Family-1 gates plus their portfolio union, against the rank > 128 reference-disagreement target on 3,000,000 held-out tokens (the target is candidate-vs-reference disagreement, not absolute next-token correctness). Purity ($\hat{\rho}$) tracks the predicate, not the rejection rate: it spans more than an order of magnitude (0.00% to 1.20%), and `first_byte_mismatch` fires most often (0.476%) yet has near-lowest purity. The union does not dominate: $\hat{\Gamma} = 12.86\%$ (2.6 points above the strongest single gate), but $\hat{\rho} = 0.32\%$, below bigram's 1.20%. None meets the formal $\rho = 1$ criterion; we report $\hat{\rho}$ and $\hat{\Gamma}$ rather than deterministic $\hat{\eta}$.

The table shows two things. The gates are target-specific: purity tracks which feature matches the target, with `bigram_low_prob` (distributional anomaly) aligning to this distributional target and `token_length_mismatch` (length) scoring zero. And the alignment is structural, not bought by detector sophistication: three negative controls matched to bigram's 0.212% rejection rate, random vocabulary permutation ($\hat{\rho} = 0.000\%$), random bigram (0.063%), and random rejection (0.032%), all score at least $19\times$ below the gate's 1.20%. The lift is genuine, roughly $48\times$ the target's 0.025% base rate.

The absolute numbers are modest: the strongest gate reaches 1.20% purity, the union only 0.32%, and none of the tested ambient gates meets the deterministic $\rho = 1$ criterion, so we report $\hat{\rho}$ and $\hat{\Gamma}$ rather than $\hat{\eta}$. Composition therefore dilutes at the token layer: the gates are near-disjoint but individually low-purity, so pooling their rejections lowers purity even as coverage rises. This is the mirror image of the structured-output layer, where the aligned surfaces have no in-sample false positives and their union both-rises (§5.5); composition is constructive exactly when the composed surfaces are target-aligned, which the framework measures rather than assumes.

6.3 The ambient gates: disjoint and robust

Two properties characterize the ambient token gates.

Disjointness. Structurally distinct surfaces should fire on different cases, and they do. With each surface at its native operating point, cross-family (cross-surface) Jaccard is uniformly ≤ 0.012 (largest, `bigram` $\cap$ `cross_model_nll`, 0.0119), while within-family (same-surface, cross-gate) overlap runs higher but stays below 0.04 (the two gold-NLL gates, sharing one NLL stream, overlap most at 0.0305). The full 6×6 matrix is in the Supplemental Analysis.

Robustness. To check that the pattern is not specific to the SP1024 / 6-layer transformer setup, we replicate end-to-end under two single-axis perturbations: a larger tokenizer (SP8192, $8\times$ vocab) and a different recurrent block family (canonical Mamba-3 (Lahoti et al., 2026)), holding recipe and seeds fixed.

Both preserve the pattern: cross-family disjointness holds across all three regimes (maximum pairwise Jaccard 0.019–0.031, always the within-Family-2 NLL pair; cross-family ≤ 0.012), the top-two ordering at rank > 128 is unchanged (`bigram_low_prob` > `cross_model_nll`), and no family inversion appears. Magnitudes shift honestly, `bigram_low_prob` sharpens (up to 124 × under Mamba) while `first_byte_mismatch` and `word_boundary_mismatch` weaken. Full per-gate tables are in the reproducibility materials.

6.4 Cross-layer reading

Across §6 the diagnostic pattern holds at the token layer: the gates are target-specific (§6.2) and near-disjoint and robust (§6.3). The token layer shows that pattern with lower overlap and a finer gate-level decomposition than the structured-output layer. On the token-emission layer (§6.3), gate-pair Jaccard stays at ≤ 0.031 across the main and robustness regimes, and cross-family (cross-surface) overlap at ≤ 0.012. On the structured-output layer the comparable surface pairs run higher but still low: $B \cap D = 0.032$ on naturalistic ABSA (§3); 0.000, 0.037, and 0.171 on the product hard corpus (§5.4); and at most 0.194 across the domain stress corpora (§5.5). A plausible reading is that structured-output failures are more semantically correlated at the case level: an input that elicits a hallucinated product is also more likely to elicit a between-model rating disagreement than two distinct token-level gates are to coincide.

The cross-model surface recurs across layers. One surface appears at both layers in the same form: cross-model disagreement (surface D of §5; `cross_model_argmax` at the token layer). Its profile is consistent across both. Surface D fires on 14.1% of Llama / Qwen extractions and aligns with the any-error target at 1.31 × over marginal (recall 18.5%); the token version fires on 26.3% of positions, aligns with gold-correctness at 1.50 ×, and recalls 96% of rank-disagreement cases. Both are broad, high-recall gates whose rejections only proxy error, unlike the narrow, disjoint gates above. The label-free ω model-quality reading (firing falls as the pair strengthens) is a structured-layer result (§5); we do not vary model quality at the token layer, so the token instance shows the surface and its profile, not its quality-dependence. A model quality examination is future work.

7 Sensitivity across corruption-target severity

A gate that looks aligned to a target might only be aligned to the particular way that target was defined. This section stress-tests that: it sweeps how strictly "wrong" is defined and checks whether each gate's alignment survives. Often it does not. A gate that reliably flags one definition of "wrong" can be useless, or even anti-aligned, against another, and the gate that ranks best on one target can rank worst on a different one.

The practical consequence is simple: a gate cannot be chosen without first naming the target it is meant to catch. The §6 gates are evaluated against two target families. *Reference-disagreement targets* ($L = \{\hat{y} \text{ has rank } > k \text{ under the reference}\}$, $k \in \{64, 128, 256\}$) use a held-out same-architecture different-seed reference and measure model-vs-model disagreement. *Gold-correctness target* ($L = \{\hat{y} \neq \text{gold next token}\}$) is the absolute-correctness signal.

Gate	Reject rate (eval)	$T_{\text{rank>64}}$	$T_{\text{rank>128}}$	$T_{\text{rank>256}}$	gold_top1_wrong
Family 1 (structural anomaly)					
bigram_low_prob	0.212%	2.27% (31.2 ×)	0.80% (33.7 ×)	0.24% (46.3 ×)	48.96% (0.87 ×)
first_byte_mismatch	0.478%	0.23% (3.2 ×)	0.10% (4.4 ×)	0.04% (8.2 ×)	63.75% (1.13 ×)
token_length_mismatch	0.030%	0.00% (0.0 ×)	0.00% (0.0 ×)	0.00% (0.0 ×)	58.55% (1.04 ×)
word_boundary_mismatch	0.290%	0.55% (7.6 ×)	0.24% (10.1 ×)	0.07% (13.5 ×)	61.13% (1.08 ×)
Family 2 (gold-NLL)					
model_surprise_s42	0.304%	0.36% (5.0 ×)	0.13% (5.5 ×)	0.03% (6.5 ×)	99.87% (1.77 ×)
cross_model_nll	0.230%	1.39% (19.1 ×)	0.56% (23.7 ×)	0.22% (42.5 ×)	86.37% (1.53 ×)

Table 16: Per-gate precision ($\hat{\rho}$) and enrichment (in parentheses, = precision / marginal_target_prevalence) for six gates across four targets, on the single-seed s42 1M-token target-severity slice (vs. the 3M-token 3-seed pool of §6′s headline table; smaller-sample point estimates here are slightly lower for bigram_low_prob rank > 128: $\hat{\rho} = 0.80\%$ vs 1.197% pooled). Target marginal prevalences on eval slice: $T_{\text{rank>64}} = 0.073\%$, $T_{\text{rank>128}} = 0.024\%$, $T_{\text{rank>256}} = 0.005\%$, $T_{\text{gold\_top1\_wrong}} = 56.5\%$. Enrichment > 1.0 × means the gate concentrates the target above its base rate; < 1.0 × means anti-aligned.

Gate ordering inverts across target families. At rank > 128: bigram (33.7 ×) > cross_model_nll (23.7 ×) > word_boundary (10.1 ×) > model_surprise (5.5 ×) > first_byte (4.4 ×) > token_length (0 ×). At gold-correctness: model_surprise (1.77 ×) > cross_model_nll (1.53 ×) > first_byte (1.13 ×) > word_boundary (1.08 ×) > token_length (1.04 ×) > bigram (0.87 ×, *anti-aligned*), a near-full reversal. The gold enrichments are all small by arithmetic necessity rather than gate failure: the target's 56.5% base rate caps enrichment at 1.77 × ($= \frac{1}{0.565}$), which model surprise nearly saturates, so what is informative on gold is the ordering, not the magnitude. Even within the gold-NLL family the two gates diverge: cross-model NLL transfers to rank-disagreement far better than model surprise (23.7 × vs 5.5 × on rank > 128). *The same gates achieve different relative orderings on different targets; gate choice is target-specific even within a fixed surface.*

Per-gate recall. On rank > 128: cross_model_nll 5.5%, bigram_low_prob 7.1%, word_boundary_mismatch 2.9%, model_surprise 1.7%, first_byte 2.1%, token_length 0%. On rank > 256 (rarest target), bigram and cross_model_nll tie at 9.8%.

7.1 Selective-prediction view (AURC for continuous-score gates)

Three gates have continuous underlying scores: bigram_low_prob's $-\log P_{\text{bi}}$, model_surprise_s42′s NLL on the actual continuation, and cross_model_nll's $|\text{NLL}_{\text{s42}} - \text{NLL}_{\text{s7}}|$. We report Area Under Risk-Coverage (AURC (Geifman & El-Yaniv, 2017); lower is better) and lift over random.

Continuous-score gate	$T_{\text{rank>64}}$	$T_{\text{rank>128}}$	$T_{\text{rank>256}}$	gold_top1_wrong
bigram_neg_log_prob — AURC (lift over random)	2.99 ×	4.22 ×	9.03 ×	1.10 ×
model_surprise_s42 — AURC (lift)	3.38 ×	3.41 ×	4.62 ×	2.62 ×
cross_model_nll — AURC (lift)	2.47 ×	2.74 ×	3.17 ×	1.63 ×

Table 17: Lift over random (= marginal_target_prevalence / gate_AURC). Lift 1.0 × = random. NLL-based gates achieve substantially higher lift on gold-correctness (2.62 ×, 1.63 ×) than bigram (1.10 ×). The two NLL-based gates achieve their highest lift on rank > 256 (4.62 ×, 3.17 ×), but bigram's lift on the same target is highest of the three (9.03 ×). *No single continuous-score gate dominates across all targets.*

When Y[9] is real-valued (e.g., bigram_low_prob's $-\log P_{\text{bi}}$), $\mathcal{D}$ contains a one-parameter family of threshold tests $\{Y > \theta\}$, giving a graded family of operating points; when Y is binary (e.g., first_byte_mismatch), $\mathcal{D}$'s only nontrivial test is Y itself, giving a single operating point. Both are valid surface choices, with different evaluation primitives. Choosing a surface only makes sense relative to a stated target: because the gate

[9]In the framework's sense, a representation Y is the artifact a stage emits, or any feature of it: the code an LLM writes, an image mask, a Markdown file, or as little as a scalar score or a single bit. It can itself be the stochastic output of a model; what the framework requires to be deterministic is not Y but the predicate class $\mathcal{D}$ of tests run over it. The token-level gates in this section span the range, from continuous surprisal scores to one-bit structural flags.

ranking inverts across targets, one surface is a good detector for one notion of "wrong" and a poor detector for another.

Gold-correctness severity and literal top-K. To check that the cross-target inversion is not specific to reference-disagreement, we extend the sweep with gold-NLL severity targets (cal-slice percentile anchors at p_{90}, p_{95}, p_{99}) and literal "true token not in candidate top-K" targets. Structural gates stay near-marginal on gold-correctness severity (`bigram_low_prob` $1.49 - 2.17\times$). `model_surprise_s42` enriches cross-seed s7-side targets at $9.55\times$, $18.7\times$, and $81.7\times$ across p_{90}, p_{95}, and p_{99}; the literal top-K analogue gives $2.89\times$, $3.81\times$, and $5.46\times$ at $K = 5$, 10, and 20. Two seeds trained from different initializations separately assign high NLL to the same gold tokens at the tail (Pearson $r \approx 0.96$). Full table and the cross-pipeline noise-floor analysis are in the Supplemental Analysis.

8 Constrained decoding as verification-surface design

§3-§5 measure structured-output surfaces empirically; §6-§7 measure token-level gates on token-level verification surfaces. This section places those measurements inside the wider constrained-decoding ecosystem, reading the major method families in the theory's vocabulary. After §3-§7′s empirical content, the taxonomy reads as the unifying frame: each family is a verification-surface choice distinguished by where the predicate lives and what target it aligns with.

To obtain nontrivial zero-FP coverage on a given target requires a richer surface: either a representation whose buckets separate target-positive cases, or a predicate class able to express the separating condition. The constrained-decoding ecosystem can be read, in this view, as the LLM community building richer verification surfaces, mostly without naming them as such.

8.1 Method families in framework vocabulary

- *Schema-constrained decoding* takes two forms. Per-token logit-masking implementations (Outlines) mask logits against a JSON schema S at each generation step; structural-skeleton implementations (JSON-Former) fill the fixed schema/JSON tokens directly and generate only the content tokens, against a supported JSON Schema subset. Both achieve a formal support guarantee for "well-formed against the enforced subset of S"; failure mode is the specification gap.
- *Grammar-constrained decoding* (Guidance, llama.cpp grammar mode) generalizes to arbitrary regular or context-free grammars G, same formal support-guarantee regime, same specification-gap failure mode.
- *Declarative-constraint decoding* (LMQL) specifies output constraints in a declarative query language and enforces them through token-level validation and masking; this is a related but distinct mechanism from arbitrary CFG decoding.
- *Hard-typed function calls* (Anthropic `strict: true`, OpenAI strict structured-output, Gemini structured-output) apply the schema-constrained construction to function signatures: a formal support guarantee on the provider-accepted JSON-Schema subset, with provider-specific gaps in constraint kinds (see §4).
- *Soft-typed function calls* (Anthropic non-strict tool-use, prompt-only schemas) pass the schema as prompt or training signal without decoder-level masking; empirical validity-rate regime; failure mode is silent coercion at the API parser.
- *Assertion-with-retry* (DSPy Assertions (Singhvi et al., 2023) and analogous output-validator patterns) deploys a deterministic predicate over free decoding and escalates on failure, to retry, self-correction, or a human; empirical repair rate at deployment with an operationally-growing $\mathcal{D}_{k(t)}$; failure mode is retry oscillation and predicate drift.
- *Reasoning-agent feedback and verification surfaces.* Process reward models (Lightman et al., 2024) deploy a step-level verifier $\mathcal{D}_{\text{step}}$ that scores each intermediate reasoning step; MiroThinker-H1 (MiroMind Team, 2026) explicitly composes a step-level surface with a global-chain audit $\mathcal{D}_{\text{chain}}$, so $\mathcal{D}_{\text{agent}} = \mathcal{D}^*_{\text{step}} \cap \mathcal{D}_{\text{chain}}$.[10]

- *Free decoding plus post-hoc validation*, the §6 regime, leaves the decoder unconstrained and applies deterministic predicates afterward; empirical $\hat{\rho}$, alignment-dependent on the chosen validator, failure mode is coverage gap.

Reward-model-style and learned-QC surfaces (process reward models as scorers, learned QC heads, judge models, calibrated-softmax thresholding) are also verification surfaces in this view's vocabulary, but they introduce a second supervised model whose target alignment is itself a design choice; we leave them to follow-on work.

8.2 Putting it together

Method family (mapped conceptually)	**Regime**	**Aligned target**	**Cost / mechanism**
Schema-constrained, per-token logit masking (Outlines)	formal support guarantee	well-formed against schema S	per-token logit masking + schema spec
Schema-constrained, structural-skeleton fill (JSONFormer)	formal support guarantee on supported JSON Schema subset	well-formed against supported subset	fills fixed JSON tokens, generates only content
Grammar-constrained (Guidance, llama.cpp grammar)	formal support guarantee	string $\in L(G)$ for grammar G	per-token decoder work + grammar spec
Declarative-constraint (LMQL)	formal guarantee for constraints actually enforced by the runtime	output satisfies declarative spec	query-level constraints + token-level validation/ masking
Type-checked function calls: OpenAI strict structured-output	documented support guarantee on accepted JSON-Schema subset	well-typed against provider's accepted JSON-Schema subset	provider-side decoder masking; subset of JSON-Schema accepted
Reasoning-agent intermediate-step verification (PRMs, step-checkers, MiroThinker-H1 mid-chain)	empirical target-satisfaction rate per step	next-step coherence / process correctness	trained PRM or rule-based step-checker; per-step inference
Reasoning-agent global-chain audit (Reflexion, MiroThinker-H1 global)	empirical target-satisfaction rate on full chain	task-level correctness / policy compliance	post-chain inference pass; predicate over the full trace

Table 18: Constrained-decoding and reasoning-agent methods mapped to verification-surface choices conceptually (not measured here): formal-vs-empirical regime, aligned target, and cost/mechanism (where the predicate lives). OpenAI strict structured-output is a documented row (provider documentation); the measured provider arms (Anthropic, Gemini, Ollama, §4) are in the companion table below. *Each family is a different surface choice, with a different regime and a different target alignment.*

[10]Adjacent agent-reasoning methods that fit this view's vocabulary but operate differently include ReAct (Yao et al., 2023) (interleaved reasoning and acting; it exposes a reasoning/action trace that can be checked, not itself a verifier surface) and Reflexion (Shinn et al., 2023) (verbal self-reflection modeled as an empirical correction channel, not a deterministic audit surface). Both operate in an empirical target-satisfaction regime at runtime, and the framework predicts low cross-kind Jaccard at matched coverage between verifier-side and feedback-side surfaces.

Method family (measured here)	**Regime**	**Aligned target**	**Cost / mechanism**
Type-checked function calls: Anthropic `strict: true` tool-use (§4)	formal support guarantee on provider-compatible schema subset	well-typed against declared signature; pattern + enum + string-length retained; numeric/integer/ array bounds stripped	signature spec + decoder-level masking
Type-checked function calls: Gemini structured-output (§4)	formal support guarantee on provider-compatible schema	structural + enum; numeric bounds accepted in the dialect but not decoder-enforced (Google, 2026)	function-call schema spec + decoder-level masking
Type-checked function calls: Ollama grammar mode, llama.cpp (§4)	formal support guarantee on provider-compatible schema (structure + enum + bounds; `pattern` stripped)	well-typed against grammar dialect; cross-field constraints not expressible	grammar compilation + per-token masking
Type-checked function calls: Anthropic non-strict tool-use (§4)	empirical schema-valid output rate	structurally-typed; pattern / bound not enforced	API-side parser only
Assertion + retry, DSPy (§4 vignette, $n = 75$)	oracle channel; empirical repair rate	predicate-defined; grows over time	retry / human / model correction loop
Free decoding + post-hoc validation (§4 free arms, §6)	empirical target-satisfaction rate	validator-defined; alignment-dependent	validator implementation only
Token-level structural-anomaly gates (§6 Family 1)	empirical $\hat{\rho}$; orders-of-magnitude variation	tail-corruption targets, target-specific	runtime-deployable; no extra inference, instrument decoder
Token-level gold-NLL gates (§6 Family 2)	empirical $\hat{\rho}$ on held-out corpus	tail-corruption / surprise targets	evaluation only (uses the gold next token)
Token-level deployable uncertainty gates (§6.1)	empirical $\hat{\rho}$	gold-correctness and rank-disagreement, gate-specific	runtime-deployable; from candidate softmax
Cross-class structured-output surfaces (§3, §5 B/C/D)	empirical $\hat{\rho}$	lexical / cross-field / cross-model targets	deterministic predicates over structured outputs

Table 19: Verification methods measured in this paper (§3-§7), mapped to the same three axes as the table above. This approach's central claim, different surfaces, different alignments, applies across all families regardless of whether they are measured (this table) or mapped conceptually (table above).

The unifying view changes how the LLM ecosystem reads. Outlines is not "better than" DSPy assertions; they are surface choices with different formal regimes, different target alignments, and different costs. Free decoding plus regex validation is not "weaker than" schema-constrained decoding; it operates in a different regime where empirical rejection purity or target-satisfaction rate is the relevant measurement. The "no universal QC method" prediction follows immediately: if any feasible surface had nontrivial zero-FP coverage across all targets, the surface-target distinction would be unnecessary. The constrained-decoding ecosystem's actual structure (many methods, each aligned with a different target class, no method dominant) is what the framework predicts, and the measured cases in §3-§7 instantiate it.

Runtime cost and primary failure modes. Schema- and grammar-constrained decoding (Outlines, JSONFormer, LMQL, Guidance, OpenAI/Anthropic strict, Gemini structured output, llama.cpp grammar) pay grammar compilation plus per-token mask overhead and fail by *specification gap*[11]: constraints outside

[11]Beyond the specification gap, constraining the emitted tokens carries a second cost: an autoregressive model reasons through the tokens it emits, so a schema with no field for intermediate reasoning removes the model's scratchpad, and key order fixes

the accepted dialect, such as cross-field consistency, and provider-specific gaps in regex, numeric, or array bounds. Soft-constrained tool-use APIs add no decoder cost but allow silent coercion at the parser. Assertion-with-retry adds $(1 + N) \times$ decoder time with N the retry budget; the failure mode is retry oscillation and predicate drift. Reasoning-agent verifiers add about 1.1–2 $\times$ end-to-end latency; the failure mode is verifier hallucination or PRM mis-calibration. Free decoding plus post-hoc validation has negligible runtime overhead but fails by coverage gap. Token-level deterministic gates (§6 Family 1, deployable uncertainty gates) are decoder-side instrumentation with no extra inference; the failure mode is low recall per gate, requiring composition. *The "no single method dominates" thesis is that no method family combines negligible overhead with a failure mode that covers the other families' residual failure modes.*

9 Discussion and limitations

Language models are a clean testbed for the verification-surface view. Rice-style undecidability (Rice, 1953) rules out a general computable checker for unrestricted nontrivial semantic correctness, so the diagnostic question (which targets can the chosen surface possibly succeed on?) is central rather than optional. The constrained-decoding ecosystem already produces many constructive surface examples (schema, grammar, type, assertion); the theory's role is to name what the community is doing rather than to compete with it. The empirical base is broad: million-token scales, multiple seeds, negative controls on freely available data, public naturalistic benchmarks.

This view is complementary to existing work: LLM evaluation (surface choices for judge-model targets), constitutional AI / RLHF (training-time surface design), compiled / structured generation (constructive surface design at the decoder), runtime safety monitors (oracle-channel surfaces). Each is exploring a different region of the surface-target alignment space.

9.1 Common pitfalls

Four practical anti-patterns follow from the evidence here. *Do not treat schema validity as semantic correctness*: a grammar-enforced output can still hallucinate entities, invert polarity, or violate the constraints the provider's dialect strips (§3, §4). *Do not treat an LLM judge as a universal verifier*: it is a verification surface like any other, beaten here on point-estimate precision by a deterministic cross-model gate (§5.6), and must be evaluated against the target it is asked to cover. *Do not evaluate monitors only on aggregate correctness*: the same surface inverts its precision across targets (§3, §7). *Do not compare methods without specifying target and cost regime*: schema decoding, validators, disagreement, self-consistency, judges, and token-uncertainty surfaces align with different targets, and a single-number comparison elides the dimension this paper argues is central.

9.2 Limitations

Primary-token benchmark remains centered on one main setup. The §6 token-level table uses SP1024 BPE tokenization and a 6-layer transformer (4 seeds via mixlab). Robustness checks in §6.3 cover tokenizer (SP8192) and architecture (canonical Mamba-3-style block family), the two axes a reviewer most often demands. The checks support the pattern-level claim but broader replication across larger pretrained models, additional tokenizers, and production-scale LMs remains future work.

Reference-relative targets are a controlled diagnostic. The rank $> k$ and exact-mismatch targets in §6 are defined relative to a held-out same-architecture different-seed reference; this is a controlled diagnostic, seeds matched on architecture, tokenizer, recipe, step count, but not equivalent to absolute next-token correctness. The current gold-correctness targets include top-1 (gold_top1_wrong), literal top-K (§7), and gold-NLL severity diagnostics at p_{90}, p_{95}, p_{99} cal-slice percentile anchors.

what conditions what (a rationale field declared after the artifact cannot shape it). Format restriction is therefore commonly reported to degrade generation, not only expressiveness. In this view the schema is the representation Y, so the choice that sets what is verifiable also sets what the model can compute; we note this but do not measure it.

Structured-output corpora are partly synthetic, by design for C. §5 uses templated product reviews, the multi-domain replication of §5.5 uses three additional synthetic domains, and §3 adds two public naturalistic corpora (SemEval-2014 ABSA restaurants $n = 980$ and laptops $n = 901$). The two-domain ABSA replication addresses the templated-only critique for surfaces B and D. Surface C (cross-field entailment) is appropriately templated: it is a schema-property predicate that requires an explicit cross-field consistency relation in the output schema, which natural extraction targets do not generally supply. The four templated relations across §5 + §5.5 (rating↔recommend twice; severity↔action twice) constitute C's evidence base. Beyond these two naturalistic domains, wider coverage for B and D (multi-aspect ABSA, FUNSD-style schemas, biomedical NER) is a clear next step.

Two open-weight pairs at the structured-output layer. §3-§5 use Llama 3.1 8B + Qwen 2.5 7B (primary) and Mistral 7B + Gemma 2 9B (robustness check). The cross-target inversion preserves across both pairs with shifted operating points; B specializes for aspect, D specializes for polarity on both. Further pair shifts (Phi, larger Llama/Qwen sizes, frontier-model pairs) would extend the regime classification. The §3 pipeline is reproducible from public ingredients alone, the Hugging Face ABSA mirror, two Ollama models, and the §3 extraction schema, so running it on a new pair is an afternoon of local compute; we invite exactly that replication.

Frontier provider internals are opaque. §4′s distinction between hard-constrained and soft-constrained tool-use is based on documented provider behavior and observed failure-mode patterns (patterns and bounds not enforced in non-strict Anthropic tool-use as tested; structure, types, and enums enforced at the decoder by Gemini structured output, while numeric bounds are accepted in Gemini's schema dialect but—per Google's documentation (Google, 2026)—left to client-side value validation rather than decode-time enforcement; llama.cpp grammar mode (ggml-org, 2026) compiles integer bounds into the grammar). *Anthropic's* `strict: true` *tool-use is exercised only in §4′s per-constraint matrix using a provider-compatible stripped schema.* A reviewer-relevant assumption: *support guarantees in this paper are conditional on the decoder implementation correctly enforcing the declared grammar, and on the provider's strict schema dialect retaining the constraints in the reference schema.* Open-weight implementations (llama.cpp grammar mode via Ollama, Outlines, JSONFormer) are inspectable end-to-end; provider APIs are not.

Reward-model and learned-QC surfaces deferred. Process reward models, judge models, and learned-QC classifiers are out of scope. Their target alignment is determined by their training labels; a side-by-side comparison against the deterministic surfaces here, and against the reasoning-agent verification surfaces named in §8, is a natural follow-on.

Semantic correctness remains outside schema validity. Schema-valid output is a structural target. Whether the extracted product is the *correct* product, whether the rating reflects the *correct* sentiment polarity, whether the recommendation is *semantically faithful*, these are different targets that §3 and §5 cross-class surfaces partially probe but do not measure end-to-end. Semantic correctness is domain-defined rather than a single universal target: what counts as correct differs by domain, and a general checker is undecidable in the limit and messy in practice, even for human annotators. The follow-on is therefore a domain's labeled corpus and oracle, not a universal LM-semantic surface. Semantic correctness is outside this layer, not outside the framework: the flagship reaches semantic targets through the software pipeline's test and cross-model review gates (Rothrock, 2026a), and the medical companion reaches them as clinical detection scored against gold (Rothrock, 2026b), both the same verification-surface construction applied at their own layers.

10 Conclusion

Reliability in language-model systems is target-specific. The same gate or surface inverts when the target shifts: bigram_low_prob's lift from 46 × on rank-disagreement (rank > 256) to 0.87 × (anti-aligned) on gold-correctness; surface B's precision from 96.3% on aspect to 25.9% on polarity; surface D's from 14.1% on aspect to 59.8% on polarity. Gates on distinct token-level surfaces and structured-output surfaces with structurally different representations fire on different cases: cross-family Jaccard ≤ 0.012 at the token level;

$B \cap D$ Jaccard $= 0.032$ $[0.009, 0.057]$ on naturalistic ABSA. Hard-constrained decoding's support guarantee covers only the provider-compatible subset of the schema dialect; numeric bounds, regex patterns, and cross-field consistency remain provider- and keyword-dependent. No single method dominates: surface choice determines target alignment.

This work positions verification surfaces, not models, not "uncertainty" in the abstract, as the unit of reliability for LM systems. The constrained-decoding ecosystem is a natural testbed for verification-surface theory; the framework supplies the vocabulary, and the ecosystem supplies the empirical content.

For the LLM-uncertainty research agenda, the consequence is to reframe the problem: instead of pursuing a single universal correctness score, catalog which (surface, target) pairs align and treat the tuple as the unit of evaluation. For engineering practice, the consequence is to compose cheap target-aligned surfaces against the failure modes that matter rather than escalating model size or layering generic judges against underspecified targets. Reliability is not a model property to be optimized; it is a (surface, target) alignment to be designed.

References

1rgs. (2023). *JSONFormer: Bulletproof JSON generation from any LLM*. https://github.com/1rgs/jsonformer

Aarsen, T. (2023). *Dataset Card for "tomaarsen/setfit-absa-semeval-restaurants"*. Hugging Face. https://huggingface.co/datasets/tomaarsen/setfit-absa-semeval-restaurants

Anthropic. (2026). *Strict Tool Use*. Anthropic. https://platform.claude.com/docs/en/agents-and-tools/tool-use/strict-tool-use

Beurer-Kellner, L., Fischer, M., & Vechev, M. (2023). Prompting Is Programming: A Query Language for Large Language Models. *PLDI 2023*.

Geifman, Y., & El-Yaniv, R. (2017). Selective Classification for Deep Neural Networks. *Advances in Neural Information Processing Systems 30*. https://papers.neurips.cc/paper/7073-selective-classification-for-deep-neural-networks

Gemma Team. (2024). Gemma 2: Improving Open Language Models at a Practical Size. *arXiv preprint arXiv:2408.00118*. https://arxiv.org/abs/2408.00118

ggml-org. (2026). *GBNF Guide*. GitHub. https://github.com/ggml-org/llama.cpp/blob/master/grammars/README.md

Google. (2026). *Structured Outputs*. Google AI for Developers. https://ai.google.dev/gemini-api/docs/structured-output

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., & others. (2024). The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*. https://arxiv.org/abs/2407.21783

Gu, J., Jiang, X., & Shi, Z. (2024). A Survey on LLM-as-a-Judge. *arXiv preprint arXiv:2411.15594*. https://arxiv.org/abs/2411.15594

Holm, S. (1979). A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, *6*(2), 65–70. https://www.jstor.org/stable/4615733

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. de las, Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Renard Lavaud, L., Lachaux, M.-A., Stock, P., Le Scao, T., Lavril,

T., Wang, T., Lacroix, T., & El Sayed, W. (2023). Mistral 7B. *arXiv preprint arXiv:2310.06825*. https://arxiv.org/abs/2310.06825

Khattab, O., Singhvi, A., Maheshwari, P., Zhang, Z., Santhanam, K., Vardhamanan, S., Haq, S., Sharma, A., Joshi, T. T., Moazam, H., Miller, H., Zaharia, M., & Potts, C. (2024). DSPy: Compiling declarative language model calls into self-improving pipelines. *ICLR 2024*. https://arxiv.org/abs/2310.03714

Lahoti, A., Li, K., Chen, B., Wang, C., Bick, A., Kolter, J. Z., Dao, T., & Gu, A. (2026). Mamba-3: Improved Sequence Modeling Using State Space Principles. *ICLR 2026*.

Lightman, H., Kosaraju, V., & Burda, Y. (2024). Let's Verify Step by Step. *ICLR 2024*. https://arxiv.org/abs/2305.20050

Microsoft. (2024). *Guidance: A guidance language for controlling large language models*. https://github.com/guidance-ai/guidance

MiroMind Team. (2026). MiroThinker-1.7 & H1: Towards Heavy-Duty Research Agents via Verification. *arXiv preprint arXiv:2603.15726*. https://arxiv.org/abs/2603.15726

OpenAI. (2024). *Introducing Structured Outputs in the API*. OpenAI. https://openai.com/index/introducing-structured-outputs-in-the-api/

Penedo, G., Kydlíček, H., L. Ben allal, Lozhkov, A., Mitchell, M., Raffel, C., Von Werra, L., & Wolf, T. (2024). The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. *NeurIPS 2024 Datasets and Benchmarks*. https://proceedings.neurips.cc/paper_files/paper/2024/hash/370df50ccfdf8bde18f8f9c2d9151bda-Abstract-Datasets_and_Benchmarks_Track.html

Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., & Manandhar, S. (2014). SemEval-2014 Task 4: Aspect Based Sentiment Analysis. *Proceedings of the 8th International Workshop on Semantic Evaluation (Semeval 2014)*, 27–35. https://aclanthology.org/S14-2004/

Rice, H. G. (1953). Classes of Recursively Enumerable Sets and Their Decision Problems. *Transactions of the American Mathematical Society*, *74*(2), 358–366.

Rothrock, M. (2026b). *Target-Specific Verification Surfaces for Cross-Stage Quality Assurance: A Medical Image Segmentation Case Study*. Zenodo. https://doi.org/10.5281/zenodo.20331363

Rothrock, M. (2026a). *Trust Topology: Verification Surfaces as the Unit of Reliability*. Zenodo. https://doi.org/10.5281/zenodo.20292194

Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., & Yao, S. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning. *NeurIPS 2023*. https://arxiv.org/abs/2303.11366

Singhvi, A., Shetty, M., Tan, S., Potts, C., Sen, K., Zaharia, M., & Khattab, O. (2023). DSPy Assertions: Computational Constraints for Self-Refining Language Model Pipelines. *arXiv preprint arXiv:2312.13382*. https://arxiv.org/abs/2312.13382

Wang, X., Wei, J., & Schuurmans, D. (2023). Self-Consistency Improves Chain of Thought Reasoning in Language Models. *ICLR 2023*. https://arxiv.org/abs/2203.11171

Willard, B. T., & Louf, R. (2023). Efficient Guided Generation for Large Language Models. *arXiv preprint arXiv:2307.09702*. https://arxiv.org/abs/2307.09702

Wilson, E. B. (1927). Probable Inference, the Law of Succession, and Statistical Inference. *Journal of the American Statistical Association*, *22*(158), 209–212.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., & others. (2024). Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*. https://arxiv.org/abs/2412.15115

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. *ICLR 2023*. https://arxiv.org/abs/2210.03629

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., & Stoica, I. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. *NeurIPS 2023 Datasets and Benchmarks*. https://arxiv.org/abs/2306.05685

Version 1.0 · July 2026.