What Constrained Decoding Buys at High Item Counts: Validity, Not Recall

Raunak Sood
University of Southern California
raunakso@usc.edu

September 2026

Abstract

Schema-constrained decoding is widely believed to cost accuracy, because forcing output through a grammar masks tokens the model would otherwise emit. We measure that cost along an axis it has not been measured on: the number of items the model must produce. Holding input length constant by construction, we build documents containing exactly k verbatim-validated sentences, each naming one entity, and vary k from 1 to 16, with 25 documents per level on one model. Under a pre-registered rejection rule (a recall drop of at least 0.15 from $k \leq 3$ to $k \geq 16$ and $p < 0.05$), neither decoding regime rejects the null: unconstrained recall falls 0.131 (Spearman -0.255, $p = 0.027$) and schema-constrained recall does not fall (-0.114, i.e. it rises; $p = 0.73$). The unconstrained decline is itself mostly a sampling effect: items recur across documents, the 21 items drawn at $k = 1$ are easier than the rest wherever they appear, and with item fixed effects the per-item unconstrained slope is -0.006 per doubling ($p = 0.70$). An earlier version of this paper reported a 0.214 drop and a rejection for the unconstrained arm; six of its 75 responses had been scored as zero because the tolerant parser rejected valid-looking JSON containing unescaped quotation marks copied from the filings, and a repair step that recovers them is now part of the scorer. Paired on identical documents, the gap between arms moves -0.061 per doubling of k ($p = 0.020$), but it is carried by $k = 1$, where the constrained arm emits an empty list on 13 of 25 documents; without those documents the interaction vanishes ($p = 0.17$). What does survive with an artifact is validity: the share of responses that are valid JSON as emitted falls with k without a grammar (25/25, 23/25, 21/25) and is 75/75 under one, as a grammar guarantees; all six failures involve the 10 of 97 bank items whose source sentence contains a quoted short form, so the rise with k reflects more chances to draw one. This validity analysis was added after the data were in. The arms also differ in prompt and field set, so they compare two configurations rather than the grammar alone. The study's validity gate, recall of 0.80 at $k = 1$, was never met, so absolute recall levels are not interpretable. We also show that item count, item density and document length cannot be held fixed simultaneously, which determines whether any measurement on this axis is an upper or a lower bound.

1 Introduction

Ask a language model for a list and it will sometimes give you a shorter one than the document supports. Whether that tendency worsens as the list grows is an empirical question with a

measurable answer, and it has not been measured in the setting where it matters most for production systems: structured extraction under a schema.

Two literatures bracket the question without answering it. Long-context work measures how retrieval degrades with input length and with the position of the evidence [9, 6, 5], and multi-evidence variants report that retrieval falls as the number of inserted facts rises [7, 12]. Long-list extraction is recognised as hard enough to need retrieval augmentation and a pruning stage [11], and extraction benchmarks report performance falling as schemas widen [2]. Structured-output work measures the cost of constrained decoding. The grammar masks tokens the model would otherwise emit [15]; reported accuracy losses of several points on reasoning tasks [14] are contested as artefacts of prompt mismatch [16] or of vocabulary misalignment [1]; grammar constraints help on structured NLP tasks without finetuning [3]; and a benchmark over ten thousand schemas measures validity, coverage and efficiency [4]. Closer to the present axis, Levy et al. [8] show that reasoning degrades as the same task is padded with more tokens, LongGenBench [13] measures how long structured *outputs* degrade with their required length, and NoLiMa [10] separates retrieval from lexical matching in long contexts. None of these varies the number of items to emit while holding input length fixed, and none compares decoding regimes on that axis.

The distinction matters because the two candidate explanations make opposite predictions. If degradation is a property of the model's attention over a long list of targets, a grammar should not help. If it is a property of the model choosing when to stop, a decoding regime that changes the stopping point should help. At this scale the data do not decide between them: once a parser defect is repaired (Section 5), neither arm meets the pre-registered rejection rule, and the measurable difference the grammar makes is to output validity rather than to recall.

Contributions. We fix k, the number of items to emit, as an evaluation axis and hold input length constant by constructing documents rather than annotating them, which makes k exact and the gold set free. The result is reported as a slope, not an existence claim, and at pilot scale it is a null under the pre-registered rule for both decoding regimes, with a measured between-arm gap that rests on the $k = 1$ level. The one effect of the grammar that carries an artifact is on validity: free-form output stops being valid JSON as the list lengthens. The decoding comparison itself was adopted after the original pre-registration (Section 4) and is reported as exploratory. A constraint that any study on this axis inherits is made explicit. Count, density and length are mechanically linked, so fixing two determines the third and decides the direction of the residual bias.

2 Constructing documents with known k

Annotation is the obvious approach and the wrong one. If k is a property of whichever documents happened to be sampled and a human must recover it, k is measured with error that correlates with the outcome, since an annotator's attention flags on exactly the long enumerations where the effect should live.

We construct instead. The item bank is drawn from SEC 10-K filings: sentences each naming exactly one organisation, each verified to appear verbatim in its source document. Filler is drawn from the same corpus with every sentence containing a capitalised run removed, so it cannot carry a legitimate item. A document is k item sentences injected at uniformly random positions into filler, padded to about 9,000 characters (8,991–9,248). Gold is the injected set.

Three properties follow: k is exact, input length is constant by construction rather than merely uncorrelated, and position is averaged over rather than confounded with k.

The item bank is screened by measurement, not by rule. Three syntactic filters for "which sentences transplant into a synthetic document" were tried and all three failed, leaving recall at $k = 1$ far below any usable baseline. The bank is therefore defined empirically: a candidate sentence is retained only if the extractor recovers it at $k = 1$ in a short document. 143 spans survive (`cardinality_bank_screened.parquet`), of which the 97 of at most 200 characters are eligible for documents; the number of candidates screened was not retained in a committed artifact.

A validity gate, which was not met. The pre-registration (`docs/HYPOTHESIS_CARDINALITY.md`, amendment of 2026-09-10) requires recall at $k = 1$ to clear 0.80 before any slope from a cell is interpretable: one item, one document, a sentence the extractor itself produced in the wild. The gate failed three times during construction (0.333, 0.400, 0.400), and the pre-registration's stop rule states that if it failed again after one further rebuild, the constructed design would be abandoned and the study written up as a dilution measurement and a negative report. The rebuild raised $k = 1$ recall to 0.440 under constrained decoding and 0.640 under unconstrained decoding. Neither clears 0.80. The stop rule therefore called for abandoning the design, and we proceeded anyway. The consequence is stated here and carried through the paper: absolute recall levels in this study are not interpretable as properties of the extractor, and only the contrast between the two decoding arms on identical documents is reported. The three failures were nonetheless informative. The decisive diagnosis came from holding the item and k fixed and varying only the filler (recorded in the pre-registration of 2026-09-10; the run output itself was not committed):

filler	recall at $k = 1$
1,500 characters	0.667
4,000 characters	0.583
14,000 characters	0.417

Dilution alone is worth 0.25 recall. The filler had been selected *for containing no entity names*, which made it far more inert than any real prompt, where a passage selector retains only relationship-dense text. Document length was reduced accordingly.

3 The count–density–length trilemma

The dilution measurement exposes a constraint that binds every study of this axis. Item density is mechanically determined by the other two quantities: density = (k×span length)/document length. Fixing any two fixes the third.

Held fixed	Varies	Residual confound
length and count (this work)	density	density aids high k; effect is a **lower bound**
density and count	length	length suppresses high k; an **upper bound**
length and density	—	k cannot vary

We hold length fixed, so item density rises with k: a $k = 16$ document is denser in items than a $k = 1$ document of the same length. Since dilution demonstrably suppresses recall, high-k documents receive a density advantage that works against the hypothesis. Our estimate is therefore conservative.

4 Design and pre-registration

The pre-registration is a dated, amended document (`docs/HYPOTHESIS_CARDINALITY.md`) and the history matters for reading the result. The hypothesis, that recall declines in k at fixed input length, and the rejection rule, which requires *both* a recall drop of at least 0.15 from $k \leq 3$ to $k \geq 16$ *and* $p < 0.05$ on the rank test, were fixed before any cell was run and did not change. The second factor did. The original design was a 2×2 of model size (`qwen3:32b` against `qwen3:30b-a3b`) by reasoning mode (think against no-think). On 2026-09-11 the reasoning-mode cells came back identical to three decimals, because on this stack the flag only changes which response field the same tokens are filed under, and the factor was replaced by constrained against unconstrained decoding. That replacement was recorded before the unconstrained arm was run, and the literature review had already named decoding regime as the strongest competing explanation, but the decoding comparison was not part of the original pre-registration and the interaction test in Section 5 was not pre-specified at all. The repository's first commit (2026-09-14) contains both the pre-registration and the results, so the ordering rests on the document's own dated amendments rather than on commit history. Bucket means are for display; estimation is per document.

Both arms use `qwen3:14b` at temperature 0 with a 32,768-token context and a 24,000-token generation ceiling, on the same 75 documents, 25 at each of $k \in \{1, 4, 16\}$. The constrained arm is parsed with the strict schema parser, since its output is grammar-guaranteed to validate. The unconstrained arm is prompted for JSON (a three-field format instruction that the constrained arm does not receive; the constrained schema instead requires `confidence` and permits `revenue_pct`, so its output carries five fields) and parsed post hoc with a deliberately tolerant parser that strips reasoning blocks and code fences and scans for the last balanced object: a parse failure would score as a recall failure and confound the arm under test. That is exactly what happened in the first scoring of these runs: six of the 75 unconstrained responses are complete objects whose string values contain unescaped quotation marks copied verbatim from the filings, such as `Express Scripts, Inc. ("Express Scripts")`, which is not valid JSON, and the parser scored all six as zero links. They carry 39 links, 38 of which match gold. The scorer now escapes interior quotes and re-parses, failing closed when the repair does not yield valid JSON (`repair_quotes` in `run_decoding_arm.py`, with unit tests), and every number in this paper is from the repaired scoring. The asymmetry in parsing favours the unconstrained arm, if anything.

The ceiling control matters and is easy to get wrong. An earlier attempt used a 30B mixture-of-experts model, which writes 7,000 to 12,000 tokens unconstrained against an 8,192-token window. Four of six $k = 16$ documents truncated mid-array, parsed to zero items, and would have reported constrained decoding as hugely superior at the level that decides the study. That arm was discarded rather than repaired. In the reported runs no response approaches the ceiling: the longest is 903 tokens (constrained) and 642 (unconstrained), against

24,000. The cache does not record a stop reason, so the absence of truncation is inferred from that margin rather than observed.

5 Results

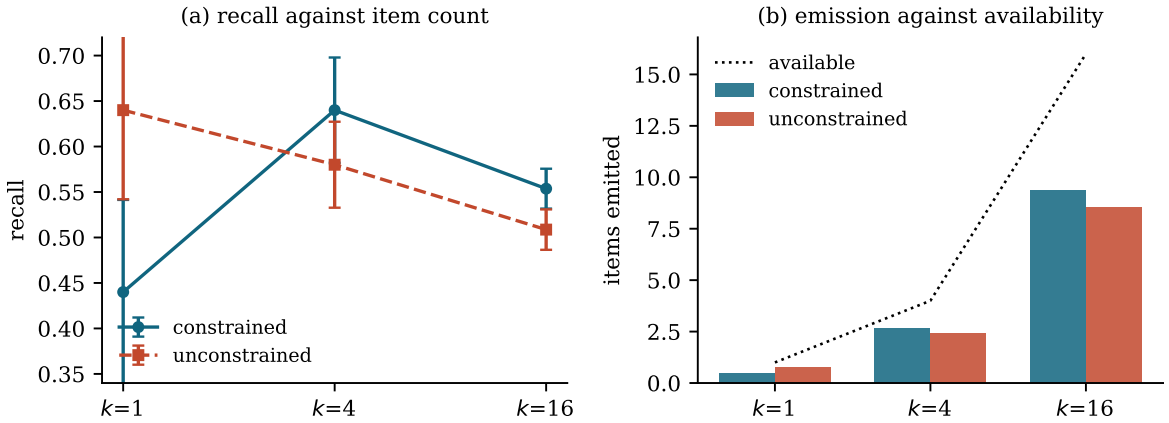


Figure 1: (a) Recall against item count for both decoding regimes, 75 matched documents per arm, error bars one standard error. The arms cross between $k = 1$ and $k = 4$. (b) Items emitted against items available. Both arms fall away from availability as k grows; the constrained arm slightly less.

k	constrained	unconstrained	paired difference	p
1	0.440	0.640	+0.200	0.096
4	0.640	0.580	−0.060	0.185
16	0.554	0.509	−0.045	0.026

Table 1: Recall by item count and decoding regime, paired on identical documents; p from a paired t-test on the per-document differences (`score_decoding.py`).

The pre-registered test rejects for neither arm. Unconstrained decoding drops 0.131 from $k \leq 3$ to $k \geq 16$ with Spearman -0.255, $p = 0.027$: the rank test clears $p < 0.05$ but the drop is below the 0.15 the rule requires. Constrained decoding drops -0.114, meaning recall rises, with Spearman $+0.041$, $p = 0.73$, satisfying neither condition. An earlier version of this paper reported 0.214 and $p = 0.011$ for the unconstrained arm and called it a rejection; the difference is entirely the six parser failures described in Section 4, which sat at $k = 4$ (two) and $k = 16$ (four).

The $k = 1$ comparison favours unconstrained decoding by 0.200 at $p = 0.096$. Per-document recall at $k = 1$ is close to a Bernoulli draw, and the variance arithmetic implies 106 documents are needed to detect a 0.15 difference at 80% power against the 25 run.

The interaction test. The pre-registered test is applied to each arm separately and never tests whether the slope differs between arms. That test was added after the data were in and is reported as such (`scripts/interaction_test.py`). Regressing the paired difference, unconstrained minus constrained, on $\log_2 k$ gives a slope of -0.061 per doubling (s.e. 0.026,

$p = 0.020$); Spearman of the paired difference against k is -0.291 ($p = 0.011$). At $k = 1$ the constrained arm returns an empty list on 13 of 25 documents, against 8 of 25 unconstrained, and those 13 documents carry the interaction: with them removed the slope is $+0.027$ ($p = 0.17$) and Spearman is -0.034 ($p = 0.79$). Within the unconstrained arm alone, recall from $k = 4$ to $k = 16$ falls from 0.580 to 0.509 with Spearman -0.233 ($p = 0.10$); within the constrained arm the same comparison falls from 0.640 to 0.554 with Spearman -0.237 ($p = 0.10$). Above $k = 1$ the two arms decline together. The between-arm difference in recall is therefore a property of the one level the study's own power arithmetic and validity gate both flag as unreliable, and it is not a finding.

Validity, the effect that has an artifact. The share of responses that are valid JSON as emitted, before any repair, is 25/25, 23/25 and 21/25 at $k = 1, 4, 16$ without a grammar and 75/75 with one (`json_valid` in the scorer; `paper/json_validity.txt` from `scripts/json_validity.py`, and the "valid JSON" column of `project/reports/decoding_result.txt`). The failures are not truncation and not refusal: they are complete objects that break the format by quoting a filing's own short form inside a string. That is the concrete thing a grammar buys at high item counts in this study, and it is the thing a production system without a repair step would have lost.

Emission counts. At $k = 16$ the constrained arm emits 9.4 of 16 available items and the unconstrained arm 8.5, at precision 0.946 and 0.953 respectively. Both arms stop well short of the 16 available. The schema is a bare `list[Link]` with no minimum length, so the grammar permits closing the array after any element, including after none, and the constrained arm does stop early and emits an empty list on 13 of 25 documents at $k = 1$. An earlier draft attributed a larger emission gap to a partially filled array failing to parse; that is false for this schema, and most of the gap was the parser defect. The manipulation that would test a grammar-level stopping account, a `minItems` bound or a required count field, was not run.

Cost. Unconstrained decoding uses 1.1× the tokens at $k = 1$ and 0.7× at $k = 16$, but the constrained schema requires two extra fields per item, so the ratio is not a cost of decoding.

Item effects (exploratory). Documents draw items from a 97-item bank, so items recur across documents and k levels. Regressing per-item hit on $\log_2 k$ with standard errors clustered by document gives an unconstrained slope of -0.035 ($p = 0.069$); adding item fixed effects gives -0.006 ($p = 0.70$), and the constrained slope is $+0.020$ ($p = 0.27$) (`scripts/item_fe.py`). The 21 items drawn at $k = 1$ score 0.578 unconstrained recall when they appear at $k \geq 4$, against 0.506 for every other item. The document-level decline in the unconstrained arm therefore reflects easier items at $k = 1$ more than item count. A rerun should cross items with k.

6 Scope conditions

One model, dense, one family; nothing here separates architecture from decoding regime. The reasoning-mode flag on this stack changes only which response field the same tokens arrive in, verified by identical evaluation counts, so this work says nothing about inference-time reasoning. The item bank is drawn from a single domain and was screened with a different model (`qwen3:30b-a3b`) from the one evaluated (`qwen3:14b`), so the bank is not guaranteed

recoverable at $k = 1$ by the evaluated extractor, which is one candidate explanation for the failed gate. Absolute recall levels are not interpretable, both for that reason and because the validity gate was not met. The between-arm recall contrast rests on $k = 1$, the level that is underpowered by the study's own arithmetic, and the pre-registered rule rejects for neither arm. The arms differ in prompt and field set as well as in grammar, and items were sampled independently at each k rather than crossed with it. Twenty-five documents per level and one model is a pilot scale; the next run needs at least 100 documents per level and more than one model family.

7 Conclusion

On the axis of how many items a model must emit, this pilot does not find that the grammar costs recall, and it does not find that the grammar protects it. Under the pre-registered rule neither decoding regime rejects the null. The between-arm gap is real on the full sample and is carried by the $k = 1$ level, where the constrained arm under-emits and the study's validity gate was not met; above $k = 1$ the arms decline together. What the grammar demonstrably buys is validity: without it, 4 of 25 responses at $k = 16$ are not valid JSON as emitted (Fisher $p = 0.11$ against 25/25 at that level; 69/75 against 75/75 pooled), and a scorer without a repair step reads those as zero recall, which is how an earlier version of this paper came to report a rejection it did not have.

For practitioners the implication is narrower than the earlier title suggested: on this model, free-form generation with post-hoc parsing, often proposed as an escape from grammar masking [14], produces output that must be repaired before it can be scored, and whether it costs recall is not settled at 25 documents per level. For evaluation, a single F_1 over a benchmark whose k distribution is unstated is not comparable across benchmarks, because it averages over a mix nobody reported, and a tolerant parser is part of the instrument and has to be tested like one.

Reproduction. Code, pre-registration, the constructed evaluation set and all 926 raw model responses (the 150 analysed here and earlier runs) are released at `https://github.com/quantraunak/cardinality-eval`. The reported result re-scores without a GPU.

References

[1] L. Beurer-Kellner, M. Fischer, and M. Vechev. Guiding LLMs the right way: Fast, non-invasive constrained generation. In *ICML*, 2024. arXiv:2403.06988.

[2] N. Ferguson, J. Pennington, N. Beghian, A. Mohan, D. Kiela, S. Agrawal, and T. H. Nguyen. ExtractBench: A benchmark and evaluation methodology for complex structured extraction. arXiv:2602.12247, 2026.

[3] S. Geng, M. Josifoski, M. Peyrard, and R. West. Grammar-constrained decoding for structured NLP tasks without finetuning. In *EMNLP*, 2023. arXiv:2305.13971.

[4] S. Geng et al. Generating structured outputs from language models: Benchmark and studies. arXiv:2501.10868, 2025.

[5] K. Hong, A. Troynikov, and J. Huber. Context rot: How increasing input tokens impacts LLM performance. Chroma technical report, `https://www.trychroma.com/research/context-rot`, 2025.

[6] C.-P. Hsieh et al. RULER: What's the real context size of your long-context language models? arXiv:2404.06654, 2024.

[7] LangChain. Multi needle in a haystack. `https://www.langchain.com/blog/multi-needle-in-a-haystack`, 2024.

[8] M. Levy, A. Jacoby, and Y. Goldberg. Same task, more tokens: The impact of input length on the reasoning performance of large language models. In *ACL*, 2024. arXiv:2402.14848.

[9] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[10] A. Modarressi, H. Deilamsalehy, F. Dernoncourt, T. Bui, R. A. Rossi, S. Yoon, and H. Schütze. NoLiMa: Long-context evaluation beyond literal matching. arXiv:2502.05167, 2025.

[11] S. Singhania, S. Razniewski, and G. Weikum. Recall them all: Retrieval-augmented language models for long object list extraction from long documents. arXiv:2405.02732, 2024.

[12] M. Song, M. Zheng, and X. Luo. Counting-Stars: A multi-evidence, position-aware, and scalable benchmark for evaluating long-context large language models. In *COLING*, 2025. arXiv:2403.11802.

[13] Y. Wu, M. Hee, Z. Hu, and R. K.-W. Lee. LongGenBench: Benchmarking long-form generation in long context LLMs. arXiv:2409.02076, 2024.

[14] Z. R. Tam, C.-K. Wu, Y.-L. Tsai, C.-Y. Lin, H.-y. Lee, and Y.-N. Chen. Let me speak freely? A study on the impact of format restrictions on performance of large language models. In *EMNLP Industry Track*, 2024. arXiv:2408.02442.

[15] B. T. Willard and R. Louf. Efficient guided generation for large language models. arXiv:2307.09702, 2023.

[16] .txt Engineering. Say what you mean: A response to "Let Me Speak Freely". `https://blog.dottxt.ai/say-what-you-mean.html`, 2024.