

CAVEWOMAN: How Large Language Models Behave Under Linguistic Input and Output Compression

Morayo Danielle Adeyemi¹ Ryan A. Rossi² Franck Dernoncourt²

¹Independent ²Adobe Research

morayo.danielle@gmail.com {ryrossi, franck.dernoncourt}@adobe.com

Abstract

“Talk short. Drop grammar. Save token.” This phrase is widely promoted as a way to reduce inference costs, but whether it saves anything depends on whether you compress the prompt or the response. We present CAVEWOMAN, a two-channel evaluation protocol that scores every question–answer pair on task accuracy, realized per-item cost, and whether the response still says what the model would have written without compression. It is tested on nine models, five short-answer benchmarks, and five grammatical reduction levels, with a ten-language replication and a two-dataset summarization study. Constraining the response reduces realized API cost in dollars by $1.7\times$ on average and up to $3\times$ without losing accuracy. Compressing the prompt raises net cost on three of the four API models, by up to **96%**, and lowers accuracy at every level. Under output compression, **over half** of responses on the six non-reasoning models are correct but no longer entail the response the model wrote without the constraint. How much accuracy a model keeps under compression varies across models and languages. Task accuracy alone does not capture the effects of compression, so compression should be evaluated on accuracy, agreement with the unconstrained response, and cost, each reported separately. Code is available at <https://github.com/danielle34/cavewoman>.

1 Introduction

Inference cost in large language models scales with both input and output token counts, and output tokens are typically priced 4–6 $\times$ higher than input tokens (Ahia et al., 2023; Nag et al., 2024). Existing compression methods reduce one side or the other, either the prompt (Jiang et al., 2023; Pan et al., 2024; juliusbrussee, 2026; wilpel, 2026) or the response (Xia et al., 2025; Song et al., 2025), but the two are studied separately and both are evaluated almost entirely through task accuracy at a reduced

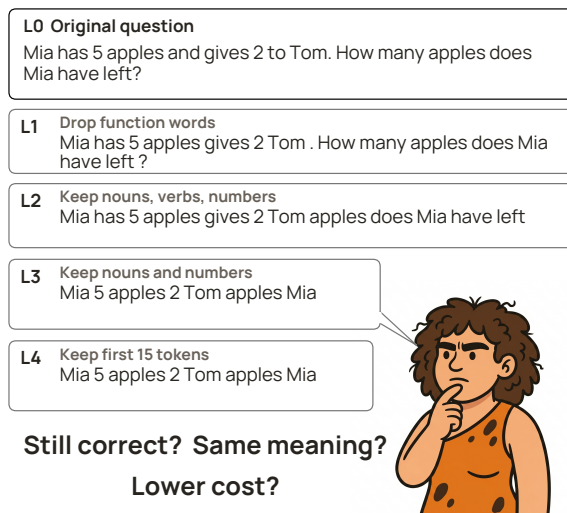


Figure 1: An illustrative question passed through the actual input filter. CAVEWOMAN tests grammatical compression on both the input to a model and the output of the model, and scores each response on accuracy, agreement with the uncompressed response, and cost.

token count. This misses two things. First, it counts a shorter prompt as a cost saving even when the model answers at greater length and the total cost goes up. Second, it scores each generation as right or wrong, which cannot tell a compressed answer that still says what the model would have written from one that no longer does. The second matters wherever the reasoning trace is used, such as in transcripts, audit logs, and displayed reasoning, since a compressed response can keep the answer and still change the reasoning those uses rely on.

We address both problems with CAVEWOMAN, a two-channel evaluation protocol that scores every question–answer pair on three axes, task accuracy, the dollar cost of each question, and agreement with the model’s own unconstrained response. The protocol measures *input compression* (the prompt is grammatically filtered before the model sees it) and *output compression* (the model is instructed to answer in a grammatically constrained way)

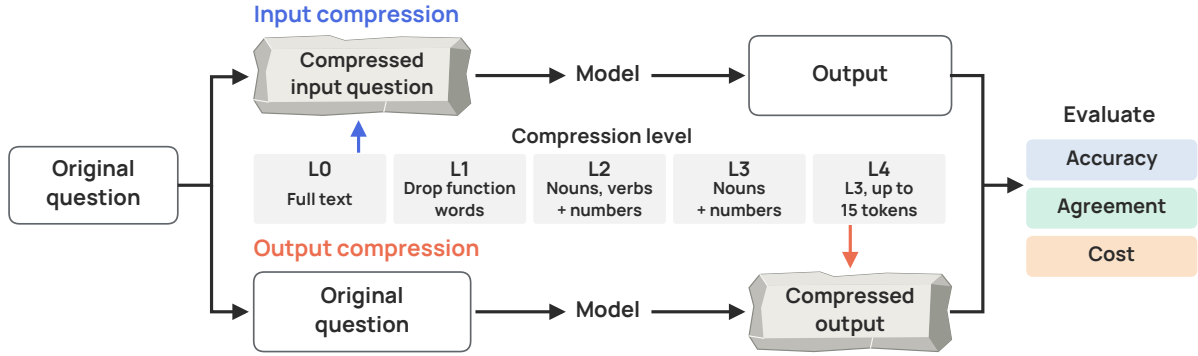


Figure 2: CAVEWOMAN framework. Input compression removes grammatical words from the question at five reduction levels and leaves the system prompt unchanged. Output compression leaves the question unchanged and replaces the system prompt with a level-specific instruction that requires the same grammatical reduction in the response. Each response is then scored for accuracy, for whether it still entails the model’s unconstrained response (bidirectional NLI), and for cost.

at five grammatical reduction levels. We evaluate nine models (Qwen2.5-VL-7B, Qwen3.5-9B, DeepSeek-R1-Distill-Qwen-7B, Gemma-4-E4B, GPT-4o, GPT-5.4, Claude Haiku 4.5, Claude Sonnet 4.6, Kimi-K2.6), on five benchmarks (GSM8K, BoolQ, ARC-Easy, CommonsenseQA, MMLU-STEM). We also extend the output channel to the MGSM dataset of grade-school math word problems in ten languages (German, Spanish, French, Swahili, Chinese, Japanese, Russian, Bengali, Thai, and Telugu), and to the XSum (one-sentence BBC news summaries) and CNN/DailyMail (multi-sentence news highlights) summarization datasets for long-form generation.

Contributions.

1. We propose a two-channel evaluation protocol that scores compression on dollar cost, on accuracy with audited answer extraction, and on twelve semantic measures (§3).
2. We measure cost asymmetry between input and output compression on the same items (§4.1).
3. We show that under output compression a correct answer often no longer says what the model would have written (§4.2).
4. We show that how much accuracy a model keeps varies across models, sizes, and languages (§4.3).

2 Related Work

Input compression. Input compression methods are either hard or soft (Li et al., 2025). Hard meth-

ods prune tokens by self-information (Li et al., 2023), perplexity (Jiang et al., 2023), question-aware scoring (Jiang et al., 2024), or distilled token classification (Pan et al., 2024), and the same idea is applied to whole documents by summarizing retrieved context (Xu et al., 2024). Soft methods encode prompts as gist tokens (Mu et al., 2023), recursive summary vectors (Chevalier et al., 2023), or autoencoder slots (Ge et al., 2024), with rate-distortion bounds in Nagle et al. (2024). Concurrent work reports that prompt compression can raise total inference cost because the response grows (Johnson and Lee, 2026). None of these methods measures whether the response says the same thing as without compression.

Output compression. Output-side methods constrain generation via modified decoding (length controls (Kikuchi et al., 2016), budget-signaling positional encodings (Takase and Okazaki, 2019), countdown mechanisms (Song et al., 2025)) or via post-training and prompting (token-skipping chains (Xia et al., 2025), cognitive-inspired routing (Aytes et al., 2025), difficulty-aware prompting (Han et al., 2025), RL-driven demonstration compression (Huang et al., 2024)). Our output channel belongs to the prompting family. Dropping function words while keeping content words mimics stereotyped prehistoric speech (caveman or cave-woman talk). These output-side methods report only task accuracy at a token budget.

Semantic fidelity, verbosity, and cost. Bidirectional NLI entailment checks whether two texts say the same thing regardless of wording (Kuhn et al.,

2023) and has scored summarization faithfulness (Maynez et al., 2020) and inter-system consistency (Laban et al., 2022). Chains of thought can be plausible yet causally disconnected from the prediction (Turpin et al., 2023; Lanham et al., 2023). Length matters. Verbose outputs score lower (Zhang et al., 2025), verbosity carries cost (Borisov et al., 2026), irrelevant padding degrades reasoning (Levy et al., 2024), and chain-of-thought length tracks accuracy independently of trace correctness (Jin et al., 2024; Wang et al., 2024; Sun et al., 2025). Other work trades cost against quality by routing queries between models, through selective routing (Chen et al., 2024), quality thresholds (Ding et al., 2024), and preference routers (Ong et al., 2025). Benchmarking work argues for reporting several metrics (Bowman and Dahl, 2021; Gehrmann et al., 2021; Liang et al., 2023). None of this work measures realized cost, task accuracy, and agreement with the model’s unconstrained output together on the same items, which CAVEWOMAN does.

3 Methodology

3.1 Experimental Design

Setup. Let $\mathcal{M}$ be a fixed language model and let $x = (w_1, \dots, w_n)$ be a question tokenized by spaCy. Each token w has a Penn Treebank part-of-speech tag $g(w)$ from `en_core_web_sm`, which is deterministic for a fixed model version. For a tag set S , write $(w_i)_{i: g(w_i) \in S}$ for the subsequence of x retaining the tokens whose tag lies in S , indices in increasing order, and let trunc_k take the first $\min(|z|, k)$ tokens of z . The reduction level $\ell \in \{0, 1, 2, 3, 4\}$, named L0 through L4, selects one filter from

$$\begin{aligned}\phi_0(x) &= x, \\ \phi_1(x) &= (w_i)_{i: g(w_i) \notin \{\text{DT}, \text{IN}, \text{CC}, \text{RP}, \text{TO}, \text{MD}\}}, \\ \phi_2(x) &= (w_i)_{i: g(w_i) \in \text{NN}^* \cup \text{VB}^* \cup \{\text{CD}\}}, \\ \phi_3(x) &= (w_i)_{i: g(w_i) \in \text{NN}^* \cup \{\text{CD}\}}, \\ \phi_4(x) &= \text{trunc}_{15}(\phi_3(x)).\end{aligned}\tag{1}$$

The family is nested. For every x and every $\ell \geq 1$, $\phi_\ell(x)$ is a subsequence of $\phi_{\ell-1}(x)$, so a larger ℓ applies a stricter reduction and a change between adjacent levels comes from the words removed at that step.

Two conditions. CAVEWOMAN holds $\mathcal{M}$ and x fixed and applies the same reduction ϕ_ℓ at one

of two points (Figure 2). The question and the system prompt are formatted with the model’s chat template, held fixed throughout.

Condition A (input compression). The model receives the filtered question $\phi_\ell(x)$ under a neutral system prompt. Condition A tests whether $\mathcal{M}$ needs a full grammatical question.

Condition B (output compression). The model receives the unmodified question x , and the system prompt instructs it to answer in the reduced form that ϕ_ℓ produces. Prompts are task-neutral, so $\mathcal{M}$ infers the answer format from x and a final-line Answer: <answer> convention. Condition B tests whether $\mathcal{M}$ answers correctly when its response is restricted to the reduced form.

3.2 Levels of Linguistic Reduction

Reduction Levels

L0: Unconstrained baseline. ϕ_0 keeps every word. Condition A presents x unchanged. Condition B uses two unconstrained prompts, L0, which asks for step-by-step reasoning in full sentences, and L0-neutral, which asks only for the answer, so that compression is measured against both (Appendix B.2).

L1: Telegraphic. ϕ_1 removes the closed-class tags DT, IN, CC, RP, TO, and MD: determiners, prepositions, conjunctions, particles, infinitival *to*, and modals.

L2: Keyword only. ϕ_2 keeps only nouns, verbs, and cardinal numbers.

L3: Noun-phrase skeleton. ϕ_3 keeps only nouns and cardinal numbers.

L4: 15-token budget. ϕ_4 applies ϕ_3 and then truncates the result to its first 15 tokens.

At L4 the Condition B prompt asks for at most 15 tokens, and generation is capped at 20. The per-level token limits apply in both conditions, so in Condition A the model’s response budget also shrinks as the level rises. Figure 10 (Appendix A.4) shows how often each model reaches the limit. Figure 1 shows one question filtered through all five levels, and a second worked example is in Appendix B (Figure 11).

3.3 Evaluation Metrics

We score every generation on three axes. The first is task accuracy, by regex extraction with a 0.01 tol-

erance on numeric answers, with extraction failures judged by an LLM judge that is validated against a second judge and four human annotators (Appendix C). The second is the dollar cost of each question, computed from its input and output token counts at the provider’s published per-token prices (Appendix A.1). The third is agreement with the unconstrained response, whether the response still entails, and is entailed by, the response the same model gave without compression, scored by bidirectional NLI (He et al., 2021) and checked against eleven other measures (two variants of our judge, four other NLI judges, a question-answering check, two word-overlap scores, and two learned similarity scores; Appendix D). A response that passes this check still says what the model would have written.

3.4 The Four Outcomes

For each level we build a 2×2 table that crosses task correctness with *agreement with the unconstrained response* (Table 1). The main text uses the L0-neutral response as the reference; Appendix H gives the same results against the step-by-step L0. C_2 (correct, but not entailing the reference response) and C_3 (incorrect, but entailing it) are outcomes that accuracy-only evaluation merges with C_1 and C_4 . We call the responses in C_2 *correct but changed*, since they keep the answer but no longer entail the reference response.

	Entails L0	Does not entail L0
Correct	C_1	C_2
Incorrect	C_3	C_4

Table 1: The four outcomes for one response, by whether it is correct and whether it entails, under bidirectional NLI, the unconstrained response the same model gave on the same channel (L0-neutral in the main text). We report the table at L1, where the NLI judge is reliable (Appendix D).

3.5 Datasets

We use five datasets spanning four task types and three answer formats, GSM8K (Cobbe et al., 2021), BoolQ (Clark et al., 2019), ARC-Easy (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), and the STEM split of MMLU (Hendrycks et al., 2021), restricted to 20 STEM subjects. Each model answers 11,465 questions at each of the five levels under both conditions. Sizes are in Appendix A.2. Three further datasets appear in the appendices,

MGSM for the multilingual study (Shi et al., 2023) and XSum (Narayan et al., 2018) and CNN/Daily-Mail (Hermann et al., 2015) for long-form generation.

3.6 Models

We evaluate Qwen2.5-VL-7B (Bai et al., 2025), Qwen3.5-9B (Qwen Team, 2026), DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025), Gemma-4-E4B (Google, 2026), GPT-4o (Hurst et al., 2024), GPT-5.4 (OpenAI, 2026), Claude Haiku 4.5 (Anthropic, 2025), and Claude Sonnet 4.6 (Anthropic, 2026). All eight are evaluated on every benchmark under both conditions, as is a ninth, Kimi-K2.6 (Team, 2026), which returned empty visible text on nearly every question under output compression and is reported on raw extraction only (Appendix F). The eight carry judge-corrected answer labels. We call DeepSeek-R1 and Qwen3.5-9B the *reasoning models*, because their visible output contains a reasoning trace that shares the token budget (Appendix F), and the other six the *non-reasoning models*. GPT-5.4 reasons before answering, but its trace is not returned, so its visible response is scored like the other five; its billed output does include the hidden trace (Appendix F).

3.7 Implementation Details

Input filtering uses spaCy 3.8.14 with en_core_web_sm 3.8.0 on Penn Treebank POS tags. Decoding is greedy (do_sample=False), with per-level max_new_tokens set to $\{400, 300, 200, 150, 20\}$ for L0 to L4. Bidirectional NLI uses cross-encoder/nli-deberta-v3-base, and cosine similarity uses all-MiniLM-L6-v2 (Reimers and Gurevych, 2019). The prompts and filter rules are in Appendix B. Qwen3.5-9B runs with thinking on, its default; we also run it with thinking off (Appendix G).

4 Results

Figure 4 shows accuracy for every model, benchmark, and level under both input and output compression. Under input compression accuracy falls on nearly every model and benchmark from L1 on, and at L2 and stricter it falls even more under a shrinking token limit (Appendix A.4). Accuracy changes against L0 are significant in 87% of the 320 per-cell Wilcoxon tests on raw per-item correctness after Benjamini–Hochberg correction (Appendix D).

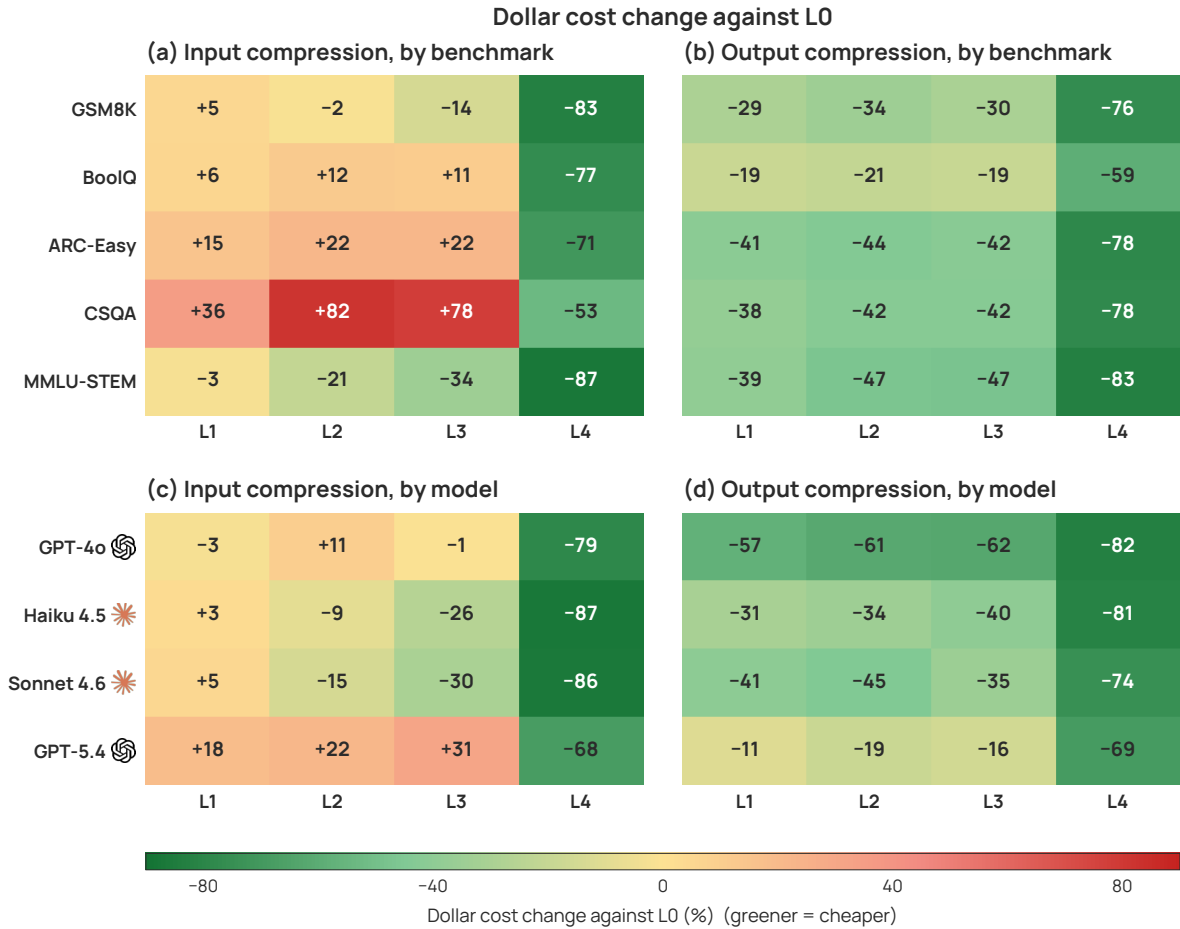


Figure 3: Change in the dollar cost of a question against the unconstrained baseline (L0) under input compression (left) and output compression (right). (a, b) By benchmark, averaged over the four API models. (c, d) By model, from the mean cost of a question over the five benchmarks. Red is an increase and green a reduction. Token counts and limits per model are in Figure 5. These panels show cost only and say nothing about accuracy, which is in Figure 4.

4.1 Finding 1: Cost asymmetry between channels

Finding 1. At the telegraphic level, output compression cuts cost on 90% of the API model and benchmark pairs, by **1.7×** on average, without losing accuracy, while input compression raises cost by **12%** on average and lowers accuracy.

Per-item cost for an API-served model is $C = n_{\text{in}} \cdot p_{\text{in}} + n_{\text{out}} \cdot p_{\text{out}}$ in US dollars at May 2026 pricing. Figure 3 shows the cost change against L0 of the same condition, by benchmark and by model. At the telegraphic level, output compression is cheaper on 18 of the 20 model and benchmark pairs. On average it saves 1.7× in dollars per question, and up to 3.1× (GPT-4o on ARC-Easy); the two exceptions, GPT-5.4 on BoolQ and

CommonsenseQA, break even. Per model the saving is 2.4× on GPT-4o, 1.7× on Sonnet, 1.4× on Haiku, and 1.1× on GPT-5.4 (Figure 3d). Input compression at L1 raises cost by +6% averaged over the four API models, by +12% averaged over the 20 model and benchmark pairs, and by +96% on GPT-5.4 on CommonsenseQA.

For input compression, stripping function words from the prompt saves 15–16 input tokens on every API model, but the model then answers at greater length, which increases the cost because output tokens are priced higher. The total cost under input compression is higher on three of the four API models (+18.4% on GPT-5.4, +5.1% on Sonnet, and +3.1% on Haiku, against -2.6% on GPT-4o; Figure 3c and Figure 5). Output compression at L1 is cheaper even though the compression instruction makes the system prompt longer, because the

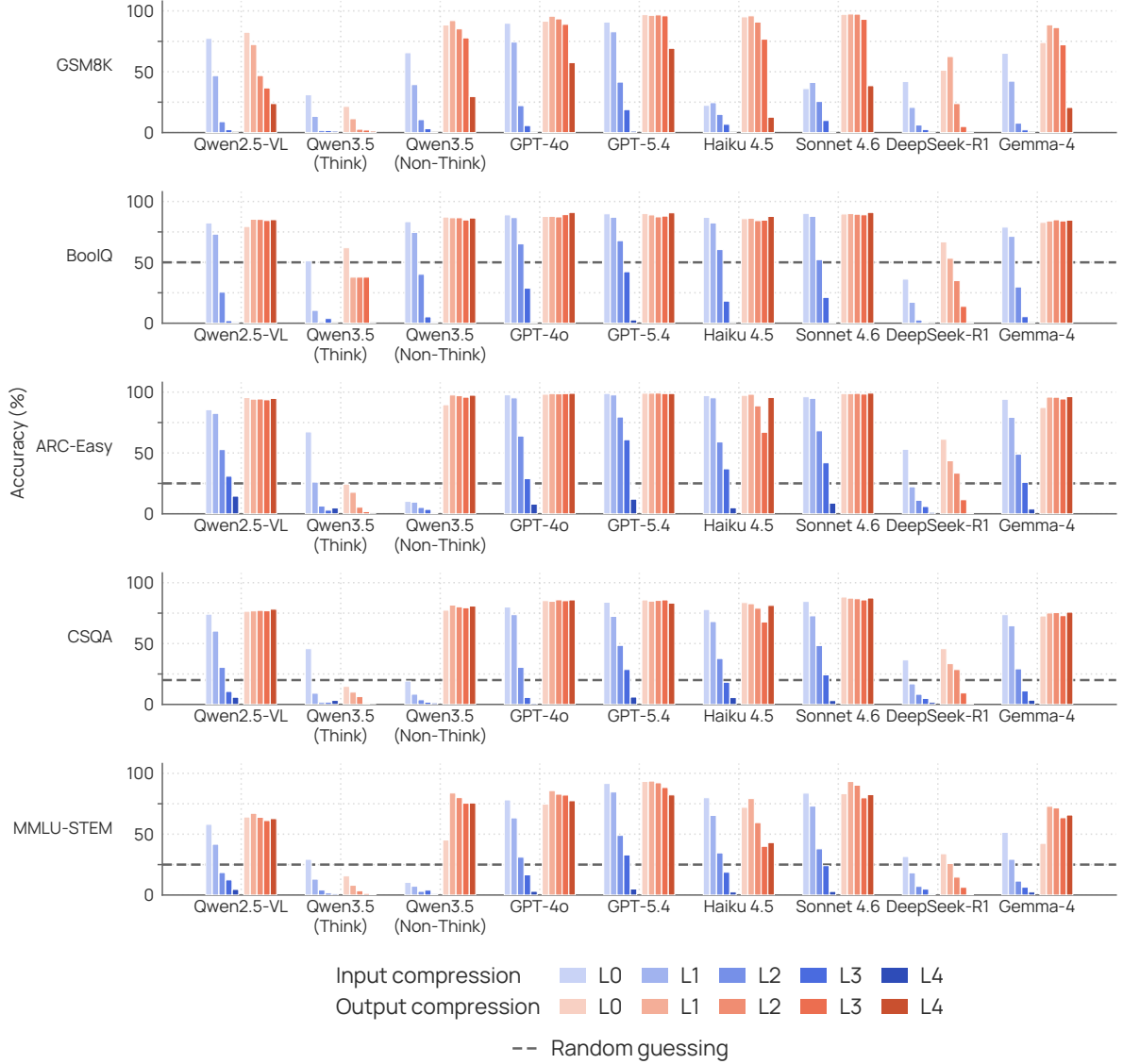


Figure 4: Accuracy for the eight main models across the five benchmarks and five reduction levels, with Qwen3.5-9B shown with thinking on (Think) and off (Non-Think, Appendix G). Blue bars are input compression and orange bars output compression, darker for stricter levels, and the dashed gray line is random guessing.

output-token saving is larger than the added input cost. Models follow the instruction. On 500 questions per model and benchmark, the six part-of-speech classes that L1 removes make up 27.6% of the tokens in unconstrained responses and only 5.5% of the tokens in L1 responses (Appendix D).

By model (Figure 3c, d), output compression saves at every level on every model, but by very different amounts. GPT-4o saves the most because it follows the instruction most tightly, cutting its answers from 242 to 53 output tokens at L1, whereas Haiku still writes 142 tokens and saves about half as much. GPT-5.4 has the least to save, since its unconstrained answers are already short at 83 to-

kens, so its saving stays between 11% and 19% until the L4 limit. Under input compression, GPT-5.4 costs more at every level through L3 because its answers grow longer as more of the prompt is removed, while its accuracy falls (Figure 4); Haiku and Sonnet become cheaper from L2 on only because the shrinking token limit cuts their answers short (Figure 5).

Savings are largest on the three multiple-choice benchmarks (ARC-Easy, CommonsenseQA, MMLU-STEM), where the L1 response is little more than the answer letter, and smallest on BoolQ, whose unconstrained answers are already short. On GSM8K, where the answer needs multi-step arith-

metic, savings fall between the two and accuracy is preserved on the four API models. Measured in dollars per correct answer, the L1 constraint makes GPT-4o $2.4\times$ cheaper and Sonnet $1.8\times$. Priced at public API rates, all four open-weight models also cost less at L1 (Appendix A.3). Part of the saving comes from removing the L0 prompt’s step-by-step instruction. Deleting that sentence lowers GPT-4o’s cost by 46% with no accuracy loss on any model, and against that neutral baseline the telegraphic level saves $1.2\times$ on GPT-4o, Haiku, and Sonnet, while GPT-5.4 costs more at L1 than under the neutral prompt (Appendix H).

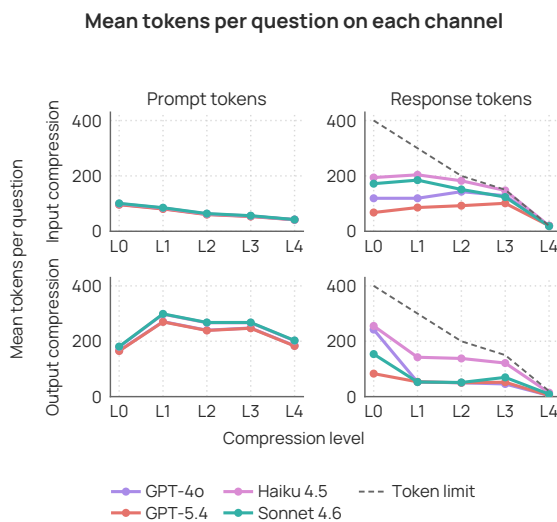


Figure 5: Mean tokens per question on each channel for the four API models, averaged over the five benchmarks, with prompt tokens on the left and response tokens on the right. The dashed line is the per-level token limit (Appendix A.4).

4.2 Finding 2: Under output compression, correct answers often no longer say what the model would have written

Finding 2. Under output compression, half of responses on the six non-reasoning models are correct but no longer say what the model would otherwise have written.

At L1 under output compression, 50.1% of responses on the six non-reasoning models are correct yet do not entail the model’s own L0-neutral response (C_2 in Table 1; 50.5% on judge-corrected labels). Figure 6 splits each model’s L1 responses into the four cells, and on every non-reasoning model C_2 is the largest cell. The two reasoning models invert the pattern because their uncon-

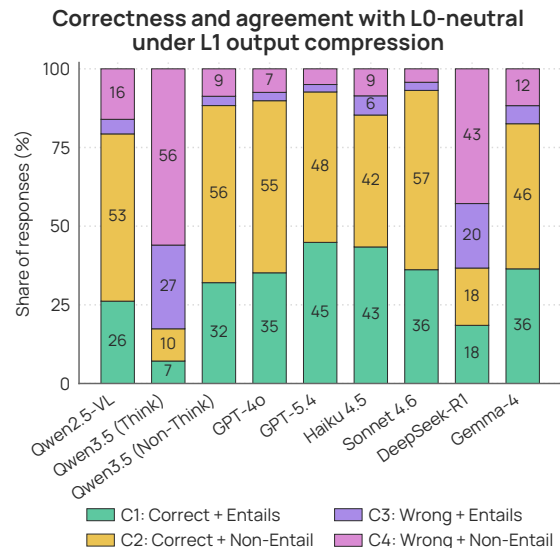


Figure 6: The four outcomes per model at L1 output compression, summed across the five benchmarks, as defined in Table 1, with the model’s L0-neutral response as the reference. Qwen3.5-9B (Non-Think) uses its 500 questions per benchmark. The same chart against the step-by-step L0 is in Appendix H.

strained generations run into the token limit, so the L0 reference is itself cut off (Appendices F and G).

The NLI judge (DeBERTa-v3, §3.7) is strict about wording as well as content. Two unconstrained responses from the same model fail to entail each other on more than half of questions (Appendix D), and compression still moves responses further from the original than resampling does on 14 of the 20 open-weight model and benchmark pairs. To separate content from wording, we also use an LLM judge, Qwen3.5-9B with the pair prompt of Appendix B.5, which compares each correct L1 response with the unconstrained response while ignoring wording and register. It finds that 70% reach the answer by the same reasoning and that one in four change or lose the reasoning (Appendix H).

Compression also changes which questions a model answers correctly. At L1 on the six non-reasoning models, 5.6% of questions change from wrong to correct and 3.9% from correct to wrong.

The choice of unconstrained reference does not cause the gap between accuracy and agreement. Scoring the compressed levels against the step-by-step L0 instead of L0-neutral gives 51.9% at L1 and the same pattern at every level (Appendix H).

Length alone does not explain the gap. When

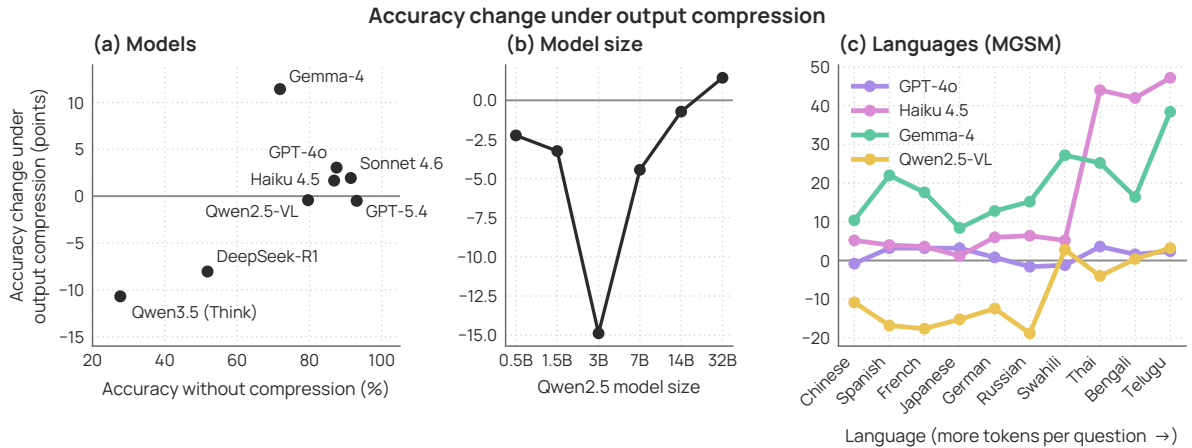


Figure 7: Accuracy change under output compression, in points: accuracy at L1 minus accuracy without the constraint (L0). (a) Each of the eight main models, averaged over the five benchmarks, plotted against its accuracy without the constraint. (b) Qwen2.5 from 0.5B to 32B on the same 500 questions per benchmark. (c) MGSM in ten languages for two API and two open-weight models, with languages ordered by how many tokens the same question costs relative to English.

each step-by-step L0 response is cut to the length of its L1 response and re-scored, the share of correct responses that fail to entail it rises from 51.9% to 80.4% (Appendix D).

All twelve semantic measures show the gap against the step-by-step L0 (Figure 12, Appendix D). It also appears under input compression with the learned LLMingua-2 compressor (Appendix I), is smaller on GSM8K than on the other four benchmarks, and persists across model sizes within Qwen2.5 (Appendix J). On long-form summarization it is larger still, since compressed summaries almost never entail the unconstrained one (Appendix L). Example responses at each level are in Appendix E.

4.3 Finding 3: Robustness to output compression varies across models and languages

Finding 3. How much accuracy a model keeps under output compression varies by model and language, and across the eight models we find no significant relation to unconstrained accuracy.

For each model we take its accuracy under telegraphic output compression minus its accuracy without the constraint, averaged over the five benchmarks (Figure 7a). Four of the six non-reasoning models change by less than two points in accuracy, GPT-4o gains 3.1 points, and Gemma-4-E4B gains 11.4, mostly on MMLU-STEM. Part of Gemma-

4’s gain comes from its unconstrained responses often running into the token limit, which its shorter compressed responses avoid (Appendix H). The two reasoning models lose 8 to 11 points in accuracy because the constraint shortens their reasoning trace. With thinking off, the same Qwen3.5-9B checkpoint gains 10.7 points in accuracy instead, so the loss comes from the shared token limit and not from the model (Appendices F and G). Across the eight models, how much accuracy a model keeps is not predicted by how accurate it is without the constraint (Spearman $\rho = 0.38$, $p = 0.35$). GPT-5.4, the most accurate model without the constraint, changes by 0.5 points in accuracy, while Gemma-4, the least accurate non-reasoning model, gains the most accuracy.

Within one family, size helps. Across Qwen2.5 from 3B to 32B on the same questions, the change rises from a fifteen-point loss to a small gain (Figure 7b, Appendix J). Below 3B the trend breaks down. The 0.5B model has little accuracy to lose, and the 1.5B model’s loss on GSM8K is offset by gains on BoolQ and CommonsenseQA. Across families, size does not predict the order, since Gemma-4-E4B, the smallest open-weight model in the main panel, gains the most accuracy.

On MGSM in ten languages (Figure 7c), the two API models keep or gain accuracy under L1 output compression, and their cost falls in every language, by a third to a half (Appendix K). Haiku’s large gains in accuracy on Thai, Bengali, and Telugu come mostly from its unconstrained answers

in those languages running into the token limit, which its compressed answers avoid, and part of Gemma-4’s gain in every language has the same cause. Qwen2.5-VL-7B loses accuracy in seven of the ten languages, most on Russian, French, and Spanish. The cost reductions are smallest in the languages whose text costs the most tokens, such as Telugu, because their compressed responses remain longer. Chinese, the cheapest language in tokens, also shows one of the smallest reductions on GPT-4o, because its unconstrained answers are already the shortest of the ten languages, so the constraint has less to remove.

5 Discussion

Output tokens are the expensive side of inference, so output compression lowers the cost of answering short-answer questions, and dropping the step-by-step instruction alone recovers much of the saving (§4.1). The saving shrinks on models whose unconstrained responses already reach the token limit and on models that bill hidden reasoning, and on long-form output it requires a numeric token limit (Appendix L).

Where the final answer is used as a decision, output compression is safe to adopt for non-reasoning models. Where the reasoning is read, the compressed response often no longer says what the model would have said. We recommend scoring compressed responses against the model’s own unconstrained responses with this protocol and an LLM judge, so that the accuracy and agreement given up for the cost saving are known.

Models gain or lose accuracy under the same constraint, so a reduction level chosen from these results cannot be assumed to carry over to another task, model, or language. The protocol is a quick

test, and one task and model cost a small fraction of the full evaluation in Figure 8, so we recommend running it on the particular setting before choosing a reduction level. A few hundred questions suffice, since subsets of 500 questions give the same share of correct but changed responses as the full benchmarks.

6 Conclusion

We presented CAVEWOMAN, a protocol that scores compression on cost, accuracy, and agreement with the model’s unconstrained response, and applied it to nine models, five benchmarks, ten languages, and two summarization datasets. Compressing the prompt and compressing the response have different effects. Output compression lowers the cost of a question by up to $3\times$ without losing accuracy on the non-reasoning models, while input compression raises it by up to 96% because the model answers at greater length. Under output compression, half of the responses are correct but no longer say what the model would have written, which accuracy alone cannot show. How much accuracy survives varies by model and language in ways that unconstrained accuracy does not predict. Before compressing, practitioners should run this evaluation on their own task and model to find the reduction level that lowers cost without losing the accuracy the task needs.

Limitations

Our NLI judge is sensitive to wording as well as content (§4.2), and the LLM judge used for answer correction, Qwen3.5-9B, is also one of the evaluated models. The per-level token limit applies to both conditions, so Condition A at L2 and stricter measures input compression under a shrinking limit (Appendix A.4); at L1 the limit binds on few output-compression responses but on 44% of Gemma-4’s and 23% of Qwen2.5-VL’s input-compression responses. Condition A reduces prompts by a fixed part-of-speech rule, so its results apply to that filter, not to natural variation in written English. The five main benchmarks have short, structured answers, and long-form generation is a two-model probe (Appendix L). We did not test with human readers whether compressed responses remain acceptable. Decoding is greedy throughout, and with two models from each of two API vendors we draw no vendor-level conclusion.

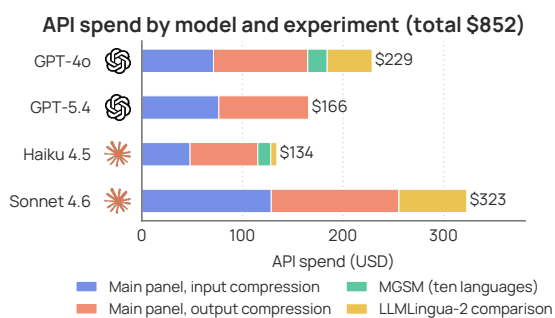


Figure 8: What the four API models cost to evaluate in this paper, by experiment, from the token counts of every billed call at the prices of Table 2. The per-level breakdown of the main panel is in Figure 9.

Acknowledgments

We thank the four colleagues who carried out the human answer-checking audit of Appendix C, and MIT Engaging for providing the GPU compute used for the local-model runs.

References

- Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David R Mortensen, Noah A. Smith, and Yulia Tsvetkov. 2023. [Do all languages cost the same? tokenization in the era of commercial language models](#). In *The 2023 Conference on Empirical Methods in Natural Language Processing*.
- Anthropic. 2025. [System card: Claude haiku 4.5](#). Technical report, Anthropic. System card documenting improvements and model safety testing for Claude Haiku 4.5.
- Anthropic. 2026. [System card: Claude sonnet 4.6](#). Technical report, Anthropic. Technical system card documenting model safeguards, model characteristics, and deployment of Claude Sonnet 4.6.
- Simon A Aytes, Jinheon Baek, and Sung Ju Hwang. 2025. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 24307–24331.
- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*.
- Vadim Borisov, Michael Gröger, Mina Mikhael, and Richard H Schreiber. 2026. Do chatbot llms talk too much? the yapbench benchmark. *arXiv preprint arXiv:2601.00624*.
- Samuel Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855.
- Lingjiao Chen, Matei Zaharia, and James Zou. 2024. [FrugalGPT: How to use large language models while reducing cost and improving performance](#). *Transactions on Machine Learning Research*. Featured Certification.
- Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846.
- Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2924–2936.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks Lakshmanan, and Ahmed H Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations*, volume 2024, pages 41348–41366.
- Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024. [In-context autoencoder for context compression in a large language model](#). In *The Twelfth International Conference on Learning Representations*.
- Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Anuoluwapo Aremu, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh Dhole, and 1 others. 2021. The gem benchmark: Natural language generation, its evaluation and metrics. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 96–120.
- Google. 2026. Gemma 4: Our most capable open models to date. <https://blog.google/innovation-and-ai/technology/developers-tools/gemma-4/>. Google Blog.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. [{DEBERTA}: {DECODING}-{enhanced} {bert} {with} {disentangled} {attention}](#).

- In *International Conference on Learning Representations*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS’15, page 1693–1701, Cambridge, MA, USA. MIT Press.
- Xijie Huang, Li Lyna Zhang, Kwang-Ting Cheng, Fan Yang, and Mao Yang. 2024. Fewer is more: Boosting math reasoning with reinforced context pruning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13674–13695.
- Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.
- Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. [LLMLingua: Compressing prompts for accelerated inference of large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.
- Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677.
- Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenye Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. 2024. The impact of reasoning step length on large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1830–1842.
- Warren Johnson and Charles Lee. 2026. Prompt compression in production task orchestration: A pre-registered randomized trial. *arXiv preprint arXiv:2603.23525*.
- juliusbrussee. 2026. Caveman. <https://github.com/juliusbrussee/caveman>. GitHub repository.
- Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338.
- Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. [Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation](#). In *The Eleventh International Conference on Learning Representations*.
- Philippe Laban, Tobias Schnabel, Paul N Bennett, and Marti A Hearst. 2022. Summac: Re-visiting nli-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.
- Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, and 1 others. 2023. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*.
- Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. Same task, more tokens: the impact of input length on the reasoning performance of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353.
- Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 6342–6353.
- Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. 2025. Prompt compression for large language models: A survey. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7182–7195.
- Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. [Holistic evaluation of language models](#). *Transactions on Machine Learning Research*. Featured Certification, Expert Certification, Outstanding Certification.
- Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 1906–1919.
- Jesse Mu, Xiang Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36:19327–19352.
- Arijit Nag, Animesh Mukherjee, Niloy Ganguly, and Soumen Chakrabarti. 2024. [Cost-performance optimization for processing low-resource language tasks](#)

- using commercial LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15681–15701, Miami, Florida, USA. Association for Computational Linguistics.
- Alliot Nagle, Adway Girish, Marco Bondaschi, Michael Gastpar, Ashok Vardhan Makkuva, and Hyeji Kim. 2024. Fundamental limits of prompt compression: A rate-distortion framework for black-box language models. *Advances in Neural Information Processing Systems*, 37:94934–94970.
- Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.
- Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2025. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*.
- OpenAI. 2026. Introducing GPT-5.4: Designed for professional work. <https://openai.com/index/introducing-gpt-5-4/>. OpenAI Blog.
- Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, and 1 others. 2024. LlmLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981.
- Qwen Team. 2026. Qwen3.5: Towards native multi-modal agents. <https://qwen.ai/blog?id=qwen3.5>. Qwen Blog.
- Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.
- Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR.
- Seoha Song, Junhyun Lee, and Hyeonmok Ko. 2025. Hansel: Output length controlling framework for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 23, pages 25146–25154.
- Yi Sun, Han Wang, Jiaqiang Li, Jiacheng Liu, Xiangyu Li, Hao Wen, Yizhen Yuan, Huiwen Zheng, Yan Liang, Yuanchun Li, and 1 others. 2025. An empirical study of llm reasoning ability under strict output length constraint. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7663–7682.
- Sho Takase and Naoaki Okazaki. 2019. Positional encoding to control output sequence length. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3999–4004, Minneapolis, Minnesota. Association for Computational Linguistics.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.
- Kimi Team. 2026. Kimi k2.6 technical report.
- Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965.
- Junlin Wang, Siddhartha Jain, Dejiao Zhang, Baishakhi Ray, Varun Kumar, and Ben Athiwaratkun. 2024. Reasoning in token economies: budget-aware evaluation of llm reasoning strategies. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19916–19939.
- wilpel. 2026. Caveman compression. <https://github.com/wilpel/caveman-compression>. GitHub repository.
- Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3351–3363.
- Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024. Re-comp: Improving retrieval-augmented lms with context compression and selective augmentation. In *International Conference on Learning Representations*, volume 2024, pages 43478–43502.
- Yusen Zhang, Sarkar Snigdha Sarathi Das, and Rui Zhang. 2025. Demystify verbosity compensation behavior of large language models. In *Proceedings of the 2nd Workshop on Uncertainty-Aware NLP (UncertainNLP 2025)*, pages 160–178.

A Pricing, Datasets, and Token Limit

This appendix gives the API pricing and model snapshots, the dataset statistics, the open-weight cost projection, and how often the token limit rather than the model ended a generation.

A.1 Pricing Assumptions and Model Snapshots

Table 2 gives the May 2026 per-token API prices and the model snapshots. The cost of a question is its input and output token counts, as counted by each model’s own tokenizer, multiplied by these prices (§4.1). The open-weight models run on local GPUs and have no metered price; Appendix A.3 projects their token counts onto public rates. Figure 9 gives what the main panel cost to run on the four API models, from every billed call in the released records at these prices, split by channel and level: \$702 in total. The output-compression bar is higher on three of the four models because it includes a sixth run, the L0-neutral prompt. The ten-language MGSM study on GPT-4o and Haiku added \$32 and the LLMingua-2 comparison \$118, for \$852 over the whole paper. Sonnet 4.6 is the most expensive model to evaluate and Haiku 4.5 the cheapest.

A.2 Dataset Statistics

Table 3 lists the number of questions in each benchmark. Each model answers all 11,465 questions at every level under both conditions.

A.3 Open-Weight Cost Projection

The four open-weight models have no metered API price, so we price their mean input and output tokens per question, pooled over the five benchmarks, at six public API rates (DSv4 Flash, DSv4 Pro, Haiku 4.5, GPT-4o, GPT-5.4, Sonnet 4.6) and take the median across the six. At L1, output compression lowers the projected cost of a question by $2.7\times$ on Qwen2.5-VL-7B and $2.1\times$ on Gemma-4, and by $1.2\times$ on the two reasoning models, whose unconstrained responses already run into the token limit.

A.4 Token-Limit Audit

The per-level token limits $\{400, 300, 200, 150, 20\}$ apply to both conditions, so a response can end because the model stopped or because it ran out of tokens. Figure 10 shows how often each model ran out. Under output compression the non-reasoning

models stop well inside the limit at L1, so the L1 cost saving is not caused by truncation, while the two reasoning models run into it on most or all of their responses at every level. Under input compression most models run into the limit on most responses from L2 on, and every model does at L4, so Condition A at L2 and stricter measures input compression under a shrinking token limit. Every input-compression claim in the main text rests on L1.

B Constraint-Level Specifications

The two conditions differ only in where the reduction is applied, so both prompts are given in full below.

B.1 Condition A: Input-Compression Filter

The system prompt is the same at all five levels, and only the question changes, through the spaCy part-of-speech filter of §3.1. The surviving words are rejoined with single spaces, and a question that the filter would empty is left unchanged. Figure 11 shows one GSM8K question at all five levels.

Neutral system prompt (identical at all levels).

Condition A: System Prompt (L0–L4)

You are a helpful assistant. Answer the following question accurately and completely.

B.2 Condition B: Per-Level System Prompts

Each prompt has a named constraint type, a rule list, a task-neutral example, and the answer-format convention. Token limits: $\{L_0:400, L_1:300, L_2:200, L_3:150, L_4:20\}$ max_new_tokens.

Condition B: L0 (Unconstrained)

Answer the following question accurately. Reason step by step in full, grammatical English sentences. Conclude with the final answer on its own line in the form 'Answer: <answer>'.

The final-line answer matches what the question asks for: a number for numeric questions, 'yes' or 'no' for yes/no questions, or a single letter (A, B, C, ...) for multiple-choice questions.

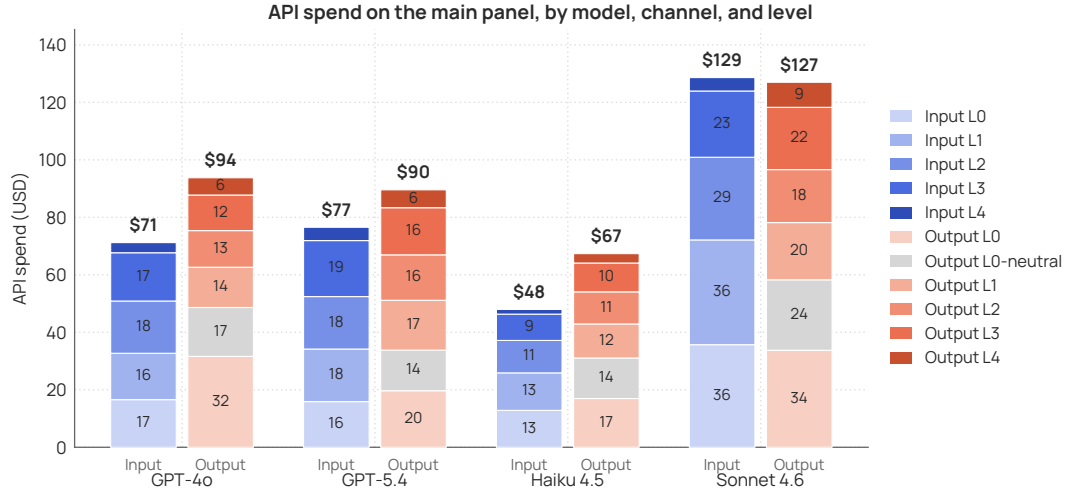


Figure 9: API spend on the main panel for each model, one bar per channel, stacked by level, computed from the token counts of every billed call at the prices of Table 2. Segment labels are dollars, and the bar total is printed on top. The output-compression bar includes both unconstrained prompts.

Model	Input (\$/M tokens)	Output (\$/M tokens)	Snapshot / endpoint
GPT-4o	2.50	10.00	gpt-4o-2024-08-06
GPT-5.4	2.50	15.00	gpt-5.4-2026-03-05
Claude Haiku 4.5	1.00	5.00	claude-haiku-4-5-20251001
Claude Sonnet 4.6	3.00	15.00	claude-sonnet-4-6
Qwen 2.5-VL-7B	local GPU		Qwen/Qwen2.5-VL-7B-Instruct
Qwen 3.5-9B	local GPU		Qwen/Qwen3.5-9B
DeepSeek-R1-Distill 7B	local GPU		deepseek-ai/DeepSeek-R1-Distill-Qwen-7B
Gemma-4-E4B	local GPU		google/gemma-4-E4B-it

Table 2: Per-token API pricing (USD/M tokens, May 2026) and model snapshots. The four open-weight models run at bf16 in single-GPU jobs on NVIDIA L40S GPUs, at most two at a time. The main-panel open-weight evaluation across all five benchmarks and both channels took approximately 610 GPU-hours.

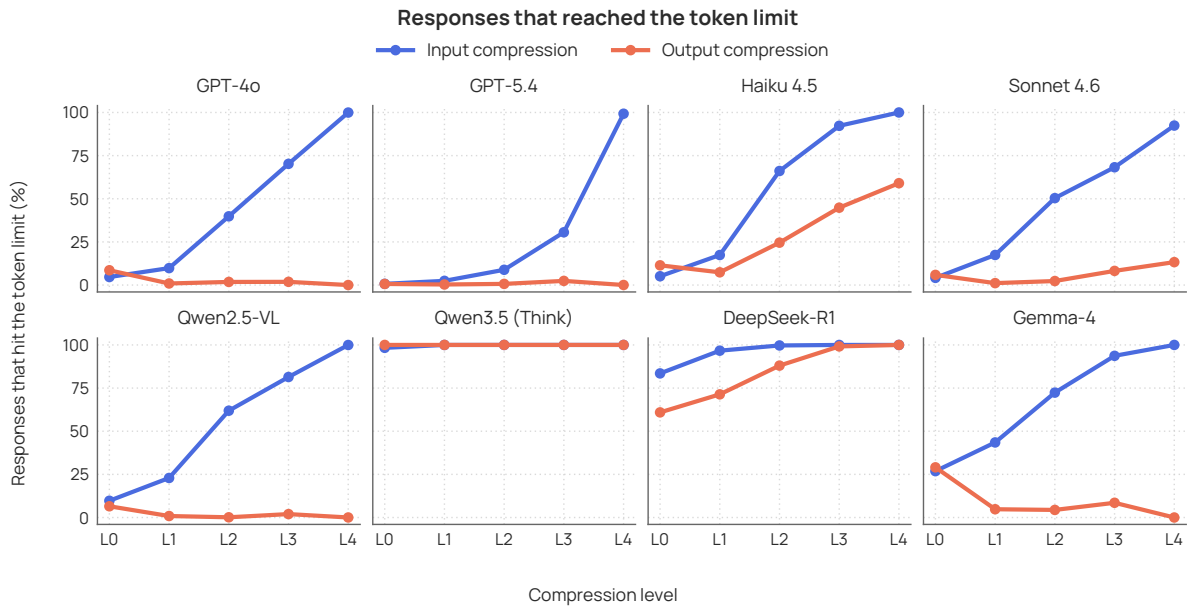


Figure 10: Share of responses that reached the per-level token limit for each model, under input compression (blue) and output compression (orange), pooled over the five benchmarks.

Input compression of one GSM8K question

L0 60 tokens	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?
L1 44 tokens	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?
L2 28 tokens	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?
L3 22 tokens	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?
L4 15 tokens	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Figure 11: Worked example of the input-compression filter applied to one GSM8K question at each of the five reduction levels. Gray, struck-through words are removed at that level, and the count is the number of spaCy tokens the model receives (Appendix I counts the same question with LLMingua-2's tokenizer, which gives 64).

Benchmark	Questions
GSM8K	1,319
BoolQ	3,270
ARC-Easy	2,376
CommonsenseQA	1,221
MMLU-STEM	3,279
Total	11,465

Table 3: The five short-answer benchmarks. MMLU-STEM is the 20 STEM subjects of MMLU.

Condition B: L0-neutral (Reasoning-Instruction Control)

Answer the following question accurately. Conclude with the final answer on its own line in the form 'Answer: <answer>'.

The final-line answer matches what the question asks for: a number for numeric questions, 'yes' or 'no' for yes/no questions, or a single letter (A, B, C, ...) for multiple-choice questions.

The L0-neutral prompt is the L0 prompt with the sentence “Reason step by step in full, grammatical English sentences.” deleted and nothing put in its place. The answer convention and the 400-token limit are the same, so the two prompts differ by exactly one instruction. The main text reports agreement against L0-neutral; Appendix H reports it against L0.

Condition B: L1 (Telegraphic)

Answer the question under a TELEGRAPHIC constraint.

Rules:

- DO NOT use any function words. No articles (the, a, an). No conjunctions (and, but, or, so). No prepositions (of, in, to, for, at, with, from, by, on, per).
- DO use nouns, main verbs, numbers, and standard symbols (+, -, *, /, =).
- Show each reasoning step.
- End with a line: 'Answer: <answer>'.

Example of the telegraphic style (task-neutral demonstration):

Premise mentions item X. Property Y holds X. Match: yes.
Answer: <answer>

The final-line answer matches what the question asks for: a number for numeric questions, 'yes' or 'no' for yes/no questions, or a single letter (A, B, C, ...) for multiple-choice questions.

Condition B: L2 (Keyword-Only)

Answer the question under a KEYWORD-ONLY constraint.

Rules:

- Use ONLY nouns and main verbs. No grammar, no full sentences.
- Output as fragments, short labels, or list items.

- Numbers and standard symbols (+, -, *, /, =) are allowed.
- Each reasoning step appears as a fragment.
- End with a line: 'Answer: <answer>'.

Example of the keyword-only style:

Item: X
Property Y: holds
Match: yes
Answer: <answer>

The final-line answer matches what the question asks for: a number for numeric questions, 'yes' or 'no' for yes/no questions, or a single letter (A, B, C, ...) for multiple-choice questions.

Condition B: L3 (Noun-Phrase Skeleton)

Answer the question under a NOUN-PHRASE SKELETON constraint.

Rules:

- NO verbs of any kind. None.
- Use only nominal fragments: nouns, noun compounds, numbers, and standard symbols (+, -, *, /, =).
- Each step is a noun phrase labelling a quantity, claim, or property.
- End with a line: 'Answer: <answer>'.

Example of the noun-phrase skeleton style:

Item: X
Property in question: Y
Match status: positive
Answer: <answer>

The final-line answer matches what the question asks for: a number for numeric questions, 'yes' or 'no' for yes/no questions, or a single letter (A, B, C, ...) for multiple-choice questions.

Condition B: L4 (Hard 15-Token Budget)

Answer the question under a HARD TOKEN BUDGET.

Rules:

- Your ENTIRE response must be 15 tokens or fewer.
- The response MUST include the final answer.
- Prefer the raw answer over prose.

Example: 'Answer: <answer>'

The final-line answer matches what the question asks for: a number for numeric questions, 'yes' or 'no' for yes/no questions, or a single letter (A, B, C, ...) for multiple-choice questions.

B.3 Long-Form Summarization Prompts

The summarization probe of Appendix L uses its own register ladder, reworded for summarization and without the Answer: convention, which has no meaning for a summary. Two settings also differ from the short-answer panel. Source documents are truncated to 6,000 characters, and the token limits are halved to $\{L_0 : 200, L_1 : 150, L_2 : 100, L_3 : 75, L_4 : 20\}$, except at L4, where the 15-token target fixes the ceiling at 20 as in the short-answer benchmarks.

Long-form summarization: system prompts (L0–L4)

L0 Summarize the article below in one to three full, grammatical sentences. Write fluent English prose.

L1 Summarize the article below under a TELEGRAPHIC constraint.

Rules:

- DO NOT use any function words. No articles (the, a, an). No conjunctions (and, but, or, so). No prepositions (of, in, to, for, at, with, from, by, on, per).
- DO use nouns, main verbs, and numbers.
- One to three telegraphic lines.

L2 Summarize the article below under a KEYWORD-ONLY constraint.

Rules:

- Use ONLY nouns and main verbs. No grammar, no full sentences.
- Output fragments, short labels, or list items.
- Numbers are allowed.

L3 Summarize the article below under a NOUN-PHRASE SKELETON constraint.

Rules:

- NO verbs of any kind. None.
- Use only nominal fragments: nouns, noun compounds, and numbers.
- Each line is a noun phrase naming an entity, quantity, or claim.

L4 Summarize the article below under a HARD TOKEN BUDGET.

Rules:

- Your ENTIRE response must be 15 tokens or fewer.
- Prefer the raw content over prose.

B.4 Multilingual (MGSM) Prompts

The ten-language study of Appendix K keeps the rule set of the Condition B ladder but is not the

same string in every language. Two things vary. Every level appends a line requiring the response to be in the language of the question, since models otherwise often reason in English, and the L1 function-word examples are replaced by the target language’s own closed-class words so that the rule is meaningful in-language. The rules, the worked example, and the answer convention are identical across languages, and the prompts stay in English, so no translation-quality confound is introduced. MGSM is numeric-answer arithmetic, so the answer-format line is numeric-only.

MGSM L1 (Telegraphic), per-language template

Answer the question under a TELEGRAPHIC constraint.

Rules:

- DO NOT use any function words. No articles ({ARTICLES}). No conjunctions ({CONJUNCTIONS}). No adpositions or case particles ({ADPOSITIONS}).
- DO use nouns, main verbs, numbers, and standard symbols (+, -, *, /, =).
- Show each reasoning step.
- End with a line: 'Answer: <answer>'.

Example (task-neutral):

Eggs 16. Breakfast 3. Muffins 4. Rest
16-3-4=9. 9*2=18.
Answer: 18

The final-line answer is a number (digits only, no units, no thousands separators).

Respond in {LANGUAGE}, the language of the question. Only the final 'Answer:' line keeps the English word 'Answer'.

The three braced slots are filled per language from Table 4, and {LANGUAGE} is the language name in English. The worked example in the prompt is the solution of MGSM item 0 (the same problem in every language), so the prompt contains the answer to one of the 250 test items; it can move a cell by at most 0.4 points.

B.5 LLM-Judge Prompts

Two judge prompts are used. The *recovery* judge of Appendix C decides whether a trace the regex could not parse reaches the gold answer, and every judge-corrected accuracy in this paper uses it. The *pair* judge of Appendix H decides whether two traces from the same model reason the same way, and it produces the AGREE rates of Appendix H. Both run on Qwen3.5-9B, and the second-judge agreement check runs the recovery prompt unchanged

on Qwen2.5-32B.

Recovery judge (answer correction)

SYSTEM

You are a precise evaluator of language-model reasoning. You will read one reasoning trace and decide whether it arrives at the correct answer to a question, regardless of how the answer is phrased or whether it appears on a separate line.

CRITICAL: Your FIRST output line must be exactly one of:

Verdict: CORRECT
Verdict: INCORRECT
Verdict: NO_ANSWER
Verdict: AMBIGUOUS

Optional reasoning may follow on subsequent lines, but the verdict line MUST come first.

USER

Question:
{question}

Choices:
{choices}

Correct answer: {gold_answer}

Reasoning trace produced by a language model:

""
{trace}
""

Decide whether the reasoning trace ends with
- or unambiguously supports - the correct answer ({gold_answer}). The trace may not use the exact format "Answer: X"; it may name the choice in words, abbreviate it, or get cut off mid-sentence.

Decide between these labels:

CORRECT - trace arrives at the correct answer
INCORRECT - trace clearly arrives at a wrong answer
NO_ANSWER - trace stops before reaching any conclusion (e.g. cut off mid-reasoning, no decisive claim)
AMBIGUOUS - trace gives multiple answers or contradicts itself

Rules:

- Be strict. If the trace is fragmentary and never commits to an answer, choose NO_ANSWER. Do not give credit for a trace that merely starts in the right direction without finishing.

OUTPUT FORMAT (verdict FIRST, reasoning optional):

Language	Articles	Conjunctions	Adpositions
English	the, a, an	and, but, or, so	of, in, to, for, at, with, from, by, on, per
German	der, die, das, ein, eine	und, aber, oder, also	von, in, zu, fuer, bei, mit, aus, durch, auf, pro
Spanish	el, la, los, las, un, una	y, pero, o, asi que	de, en, a, para, con, por, desde, hasta
French	le, la, les, un, une, des	et, mais, ou, donc	de, a, en, pour, avec, dans, sur, par
Swahili	none, demonstratives dropped instead	na, lakini, au, kwa hivyo	ya, wa, katika, kwa, kutoka, hadi, kila
Chinese	3 items	4 items	7 items
Bengali	3 items	4 items	5 items
Russian	none, stated in the prompt	4 items	8 items
Japanese	none, case particles dropped instead	4 items	10 items
Thai	none, stated in the prompt	4 items	7 items
Telugu	none, stated in the prompt	4 items	6 items

Table 4: Closed-class words substituted into the MGSM L1 template, per language. The five Latin-script languages above the rule are given verbatim. The six below it are written in scripts this document does not typeset, so their lists are given by item count here and released verbatim in `prompts_mgsm.py`. Where a language has no articles the prompt says so rather than leaving the slot empty, and for Swahili and Japanese it names a different closed class to drop in their place.

Verdict: <one of CORRECT INCORRECT NO_ANSWER AMBIGUOUS> Reasoning: <one or two short sentences explaining your verdict> Begin your response with the verdict line.	<pre> fragmented): """ {trace_b} """ Decide between these labels: AGREE - same final answer AND substantively consistent reasoning SAME_ANSWER - same final answer BUT the reasoning differs in substance (different key facts, different intermediate steps) DISAGREE - different final answer UNCLEAR - one or both traces are too fragmentary to judge Rules: - Be strict. If Trace B is so compressed that you cannot follow its reasoning, choose UNCLEAR rather than AGREE. - If the two traces appeal to different facts or different intermediate steps to reach the same answer, choose SAME_ANSWER, not AGREE. OUTPUT FORMAT (verdict FIRST, reasoning optional): Verdict: <one of AGREE SAME_ANSWER DISAGREE UNCLEAR> Reasoning: <one or two short sentences explaining your verdict> Begin your response with the verdict line. </pre>
Pair judge (reasoning sameness) SYSTEM You are a precise evaluator of language-model reasoning. You will compare two reasoning traces produced by the SAME model on the SAME question under different output-format constraints. Your job is to decide whether the two traces reach the same final conclusion using semantically consistent reasoning, ignoring surface differences in wording, length, register, or formatting. CRITICAL: Your FIRST output line must be exactly one of: Verdict: AGREE Verdict: SAME_ANSWER Verdict: DISAGREE Verdict: UNCLEAR Optional reasoning may follow on subsequent lines, but the verdict line MUST come first. USER Question: {question} Trace A (unconstrained, written in full English sentences): """ {trace_a} """ Trace B (under an output-format constraint, may be terse / telegraphic /	C Answer Checking C.1 Answer Correction Accuracy is the share of questions whose answer matches the gold answer, so it depends on reading an answer out of free-form text. If the questions the regex cannot parse are counted as wrong, ac-

curacy depends on the extractor, and an apparent gain from L0 to L1 can come from the extractor working better on shorter responses. The problem is concentrated in Condition A, whose neutral system prompt imposes no answer format; at L0 the regex parses as few as 4% of Sonnet’s CommonsenseQA responses. Even after correction, the L0 accuracies of Haiku and Sonnet on GSM8K under input compression remain low in Figure 4.

Correction ladder. We therefore check answers in three steps. First, regex extraction as before. Second, an LLM judge (Qwen3.5-9B, prompt in Appendix B.5) reads the responses the regex cannot parse and rules each one CORRECT, INCORRECT, NO-ANSWER, or AMBIGUOUS. In 134 of the 400 cells (model, benchmark, condition, level) the judge reads every unparsed response; in 199 cells with many unparsed responses it reads a seeded sample of 200 and the sample’s recovery rate is applied to the rest of the cell; the remaining 67 cells have nothing to judge. This gives 90,695 judgments. Every cell accuracy in the paper, including Figure 4 and the Finding 3 numbers, is this judge-corrected accuracy. The significance tests use raw per-question correctness, and the question-level statistics of §4.2 use per-question labels in which only judged responses are credited; on the output channel of the non-reasoning models these differ from the cell accuracies by at most 0.1 points. A second judge (Qwen2.5-32B) given the same prompt agrees with the first on 90.9% of the CORRECT-or-not decisions. Third, four human annotators audit a stratified sample of 180 responses each, 100 on which the two judges disagree and 80 of the first judge’s CORRECT verdicts.

Human audit. The four annotators reviewed the sample independently, judging whether each response reached the correct answer under the same criteria as the LLM judge and without seeing its verdicts. They were unpaid volunteer colleagues who took no part in the experiments they audited and were told how their annotations would be used. Each received the same worksheet, with the instructions below exactly as written, one worked example, and a four-way dropdown.

Annotator instructions (verbatim)

Answer-Checking Task (about 30 minutes)

Thank you for helping. You will read short answers written by AI models and decide

whether each one reached the correct answer. No background needed.

For each row: read the "Model's answer" and decide if it arrives at the "Correct answer". Put ONE of these in the "Your verdict" column (a drop-down is provided):
CORRECT = it reaches the correct answer (in any wording or format)
INCORRECT = it clearly gives a different, wrong answer
NO_ANSWER = it stops or is cut off without giving any answer
AMBIGUOUS = it gives more than one answer, or you truly cannot tell

Only judge whether it reached the right answer. Ignore whether the reasoning is good or well written. Some answers are very short or terse; that is normal.

One row is filled in below as an example. Please do as many of the rest as you can (even a portion is helpful), then save and send the file back.

Human-measured correction. Table 5 summarizes the audit. The annotators agree with the LLM judge about as often as they agree with each other, and they dispute about one in nine of the judge’s CORRECT verdicts on the rows where the sheet showed the model’s final answer (about one in four on all rows, including those the sheet truncated). Judge-corrected accuracies are therefore not a strict lower bound. We do not apply this human correction; applying it would move the mean cell accuracy by about one point, almost all of it on the input channel. The per-cell audit is released with the artifact.

Human audit (4 annotators, 180 rows each)	
Agreement with the LLM judge (correct or not)	71.9%
Agreement between annotators	71.3%
CORRECT verdicts disputed, answer shown	11.7%
CORRECT verdicts disputed, all rows	26.4%
Mean cell accuracy change if applied	0.95 pp
output channel	0.2 pp
input channel	1.8 pp

Table 5: Human audit of the LLM judge. The disputed-verdict rate is the share of the judge’s CORRECT verdicts that the annotators overturned.

D Semantic Robustness

The judge calibration, the two-condition baseline comparison, and the bootstrap intervals cover the seven models other than Gemma-4. The other checks in this appendix cover all eight.

Judge reliability. We test the NLI judge on 70 synthetic pairs per level that say the same thing by construction, with 70 negatives. It misses 3% of the matches at L1, 29% at L2, and 50% at L3. We therefore treat L1 as reliable, L2 as supplementary, and L3 as descriptive only, and exclude L4.

Twelve semantic measures. We re-score each L1 output-compression response against its step-by-step L0 response under eleven other measures on the six non-reasoning models: two variants of our judge (forward only, and both directions above probability 0.5) on every question; four other NLI judges (DeBERTa-large, BART-large-MNLI, mDeBERTa-v3-XNLI, MiniCheck) and an STS cross-encoder on 500 questions per model and benchmark; and a question-answering check, ROUGE-L, METEOR, and BLEURT on 300. Figure 12 shows the share of responses that are correct but do not match L0 under each measure. Every measure shows the gap between accuracy and agreement, and our measure gives the same share on the subsets as on the full set.

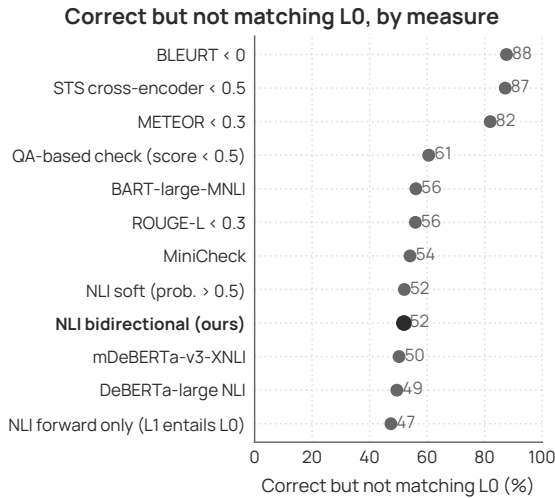


Figure 12: Share of L1 output-compression responses that are correct but do not match the L0 response, under each of the twelve semantic measures, on the six non-reasoning models (all items for the variants of our judge, 500 questions per model and benchmark for the other NLI judges and STS, 300 for the question-answering check and the word-overlap scores). Thresholds are given in the labels. Our main measure is highlighted.

Register compliance. Models follow the telegraphic instruction. Table 6 gives, for 500 questions per model and benchmark, the share of response tokens tagged with the six part-of-speech classes the L1 filter removes (DT, IN, CC, RP, TO, MD;

spaCy tags, punctuation excluded). It falls from about a quarter of the tokens in unconstrained responses to a few percent at L1 on every model, with Qwen2.5-VL the least compliant.

Model	L0	L1
GPT-4o	29.5%	4.7%
GPT-5.4	25.2%	3.9%
Haiku 4.5	26.9%	5.6%
Sonnet 4.6	25.6%	4.5%
Qwen2.5-VL-7B	29.7%	12.5%
Gemma-4-E4B	28.9%	2.1%
Mean	27.6%	5.5%

Table 6: Share of response tokens in the six part-of-speech classes the L1 filter removes, under output compression at L0 and L1.

Length and sampling controls. Table 7 collects the checks behind Finding 2. The length control cuts each step-by-step L0 response to the length of its L1 response and re-scores it. The share of correct but non-matching responses rises on every model and benchmark, so length does not explain the gap. Two floors measure how much responses differ without compression. The L0 responses of the two conditions, which differ only in the system prompt, entail each other on about a third of questions, and two temperature-0.7 draws of the unconstrained Condition B response, on the four open-weight models, on a little under half. Against the step-by-step L0, L1 entailment falls below the sampling floor on 15 of 20 model and benchmark pairs (14 against L0-neutral), and the exceptions are all DeepSeek-R1. The table also gives the per-cell significance counts and the bootstrap intervals on the C_2 share.

Check	Result
C_2 share at L1, six non-reasoning models, vs L0-neutral	50.1%
vs step-by-step L0	51.9%
with step-by-step L0 cut to the length of L1	80.4%
Condition A and Condition B L0 responses entail each other (7 models)	37.2%
Two $T=0.7$ L0 draws entail each other (4 models)	45.5%
L1 entailment below the sampling floor, vs step-by-step L0 (vs L0-neutral)	15 (14) of 20 cells
Wilcoxon tests significant after BH (320 cells)	278
input / output channel	154 / 124
Bootstrap half-width of C_2 , max (median)	2.9 (1.9) pp

Table 7: Controls for Finding 2. Wilcoxon tests use raw per-question correctness; bootstrap intervals (1,000 resamples) are on the five non-reasoning models other than Gemma-4.

E Qualitative Examples

Two items. The first, a BoolQ item (Qwen2.5-VL, Condition B) traced across all five levels, shows

C_2 by omission, where the compressed response keeps the answer and drops the reasoning. The second, a GSM8K item (Claude Haiku 4.5, L0 against L1), shows C_2 by a different route to the same answer. Box color:

green = C_1 (correct, entailment PASS) ,

yellow = C_2 (correct, entailment FAIL) , gray = L0.

BoolQ: Passage Yes/No (Qwen2.5-VL, Condition B)

Question: is harry potter and the escape from gringotts a roller coaster ride

Ground truth answer: yes

(BoolQ provides a supporting passage with each question. The passage is omitted here for space.)

L0 (Unconstrained baseline) Pred: yes GT: yes Correct

To determine if this attraction is a roller coaster ride, I analyzed the passage. The passage states it is an “indoor steel roller coaster” at Universal Studios Florida, which directly confirms it is a roller coaster. Based on this evidence. . .

Answer: yes

L1 (Telegraphic) Pred: yes GT: yes Correct NLI: FAIL [C_2]

Yes.

Answer: yes

L2 (Keyword-Only) Pred: yes GT: yes Correct NLI: FAIL [C_2]

Yes

L3 (Noun-Phrase Skeleton) Pred: yes GT: yes Correct NLI: PASS [C_1]

Item: Harry Potter and the Escape from Gringotts
Property in question: Roller coaster ride
Match status: positive

Answer: yes

L4 (Hard 15-Token Budget) Pred: yes GT: yes Correct NLI: FAIL [C_2]

Answer: Yes

Takeaway. The answer is correct at every level, but the reasoning is missing at L1, L2, and L4. The L0 output makes several inferential steps citing the passage, whereas “Yes” makes none, so “Yes” does not entail L0 and the bidirectional check fails. L3 passes the entailment check because the noun-phrase skeleton names the entity and the property,

which repeats enough of the L0 response. Accuracy alone cannot tell C_1 from C_2 .

GSM8K: Arithmetic (Claude Haiku 4.5, Condition B, L0 and L1)

Question: Bob spends \$27000 distributed between 3 schools to buy books. He can buy 100 books for \$500. How many books can he buy per school?

Ground truth answer: 1800

L0 (Unconstrained baseline) Pred: 1800 GT: 1800 Correct

I need to find how many books Bob can buy per school. Let me work through this step by step. First, I'll find the total number of books Bob can buy with \$27,000. The rate is 100 books for \$500, which means each book costs \$500 / 100 = \$5 per book. With \$27,000, Bob can buy: \$27,000 / \$5 per book = 5,400 books total. Next, I need to distribute these books equally among 3 schools. Books per school: 5,400 books / 3 schools = 1,800 books per school.

Answer: 1,800

L1 (Telegraphic) Pred: 1800 GT: 1800 Correct NLI: FAIL [C_2]

Step 1: Calculate money per school
\$27000 / 3 schools = \$9000 per school
Step 2: Calculate books per dollar
100 books costs \$500
Rate: 100 / 500 = 0.2 books per dollar
Step 3: Calculate books per school
\$9000 * 0.2 books/dollar = 1800 books per school

Answer: 1800

Takeaway. Both responses are correct, and the compressed one still shows its steps. L0 converts the whole budget to a number of books and then divides by the number of schools; L1 divides the budget by the number of schools and then converts, through a books-per-dollar rate that L0 never states. Neither response entails the other, so this is C_2 with changed content rather than omitted content. A judge told to ignore register rates this pair as the same answer reached by different steps.

F Reasoning-Token Accounting (DeepSeek-R1, GPT-5.4, Kimi-K2.6)

These three models spend one token budget on a reasoning trace (visible for DeepSeek-R1, hidden for GPT-5.4 and Kimi-K2.6) and on the visible answer, so output compression shrinks the budget the trace uses, and the visible answer is cut.

DeepSeek-R1-Distill-Qwen-7B writes <think> traces that count against max_new_tokens. At L4 (20 tokens) the budget runs out before the visible

answer starts, so its accuracy falls to near zero on every dataset (Figure 4). At L1 its C_3 cell is slightly larger than its C_2 cell, the reverse of the non-reasoning models (Qwen3.5-9B inverts more strongly, Appendix G), so its accuracy change is not comparable with the non-reasoning models.

GPT-5.4 shows the same pattern through the API. Its billed output tokens fall by about a third from L0 to L1, against a half or more on the other three API models (Figure 5), because its length is set by hidden reasoning rather than by the visible answer. The API does not expose the split between reasoning and visible tokens, so this is an inference. GPT-5.4’s cost is its billed output tokens, which include the hidden trace, at the Table 2 rate.

Kimi-K2.6 returns empty visible text on up to a third of questions already at L0 and on nearly every question at L1 and stricter under output compression, because its reasoning uses the whole budget. A judge can only score these as no answer, so we report Kimi on raw extraction and do not run the judge on it. Figure 13 shows its accuracy on raw extraction at every level, in the layout of Figure 4: under output compression it answers the unconstrained prompt well and then returns nothing visible from L1 on, and under input compression it falls from L1.

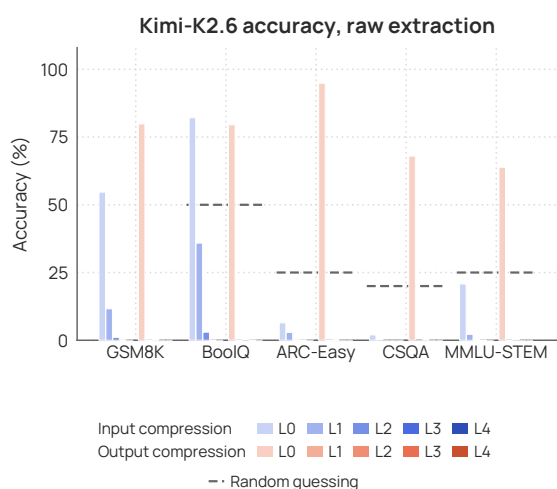


Figure 13: Kimi-K2.6 accuracy on raw regex extraction across the five benchmarks and five reduction levels, blue for input compression and orange for output compression, darker for stricter levels, with the dashed line at random guessing. Empty visible responses count as wrong, so the bars at zero are cells in which nearly every response was empty.

G Thinking-Mode Contrast

Qwen3.5-9B’s chat template turns thinking on unless it is disabled, and our main panel uses this default (§3.7). We compare the same checkpoint with thinking on and off at the same token budgets, on the same 500 questions per benchmark, with raw regex accuracy (Figure 4, Think and Non-Think). With thinking on, every output-compression response runs into the token limit at every level and accuracy stays low. With thinking off, the model stops inside the limit and reproduces Finding 2: accuracy rises from 77.6% to 88.3% under telegraphic output compression, while entailment with its own L0 response falls to 37.8%. Because the thinking-off run covers 500 questions per benchmark and is scored on raw regex extraction, its input-compression bars in Figure 4 are limited by answer extraction.

H The Two Unconstrained References (L0 and L0-neutral)

Condition B has two unconstrained prompts (Appendix B.2). L0 asks the model to “reason step by step in full, grammatical English sentences”, which is how deployed prompts for short-answer tasks are usually written, and every compressed prompt through L3 keeps that request (“Show each reasoning step” at L1, one fragment or one noun phrase per step at L2 and L3), so L0 and the compressed levels ask for the same reasoning and differ only in the grammar the response may use. L0-neutral drops the step-by-step sentence and asks only for the answer. Comparing against L0 alone would mix compression with the removal of that instruction, so the main text reports agreement against L0-neutral and this appendix gives the same results against L0. All eight models answer all five benchmarks under both prompts.

Removing the instruction never lowers accuracy.

Raw accuracy rises on all eight models: by less than five points on the API models, Qwen2.5-VL, and DeepSeek-R1, and by more on the two models whose L0 responses often run into the token limit, Qwen3.5-9B (16.5 points) and Gemma-4 (6.6). Responses get shorter, GPT-4o’s by more than half, and GPT-4o’s cost for the five benchmarks falls by 46%.

The results are the same against either reference.

Scoring each level against L0 instead of L0-neutral

moves entailment by a few points in either direction and leaves the pattern across levels unchanged (Figure 14). Figure 15 gives the four outcomes at L1 against both references, for the eight main models and for Qwen3.5-9B with thinking off. On every non-reasoning model C_2 is the largest cell under either reference, the pooled C_2 share on the six non-reasoning models is 50.1% against L0-neutral and 51.9% against L0 (50.5% and 52.3% on judge-corrected labels), no model moves by more than six points, and the two reasoning models invert in the same way. Against L0 the question-level changes are 7.1% from wrong to correct and 3.4% from correct to wrong (against L0-neutral, 5.6% and 3.9%), and L1 entailment falls below the sampling floor on 15 of the 20 open-weight cells (against L0-neutral, 14). Qwen3.5-9B with thinking off, on its 500 questions per benchmark, has C_2 at 56% against L0-neutral and 54% against L0, so once the reasoning trace no longer consumes the token limit the same checkpoint behaves like the non-reasoning models.

Pair judge. The pair judge (Appendix B.5) rates the two baselines as reasoning the same way on 60% of judged questions (500 per model), and the compressed levels against L0-neutral at 58%, 52%, 48%, and 17% from L1 to L4, pooled over all eight models. Restricted to the six non-reasoning models and to correct L1 responses (2,646), the verdicts are AGREE 70%, SAME_ANSWER 15%, UNCLEAR 10%, DISAGREE 2%, and no verdict 3%, the split reported in §4.2.

I Comparison with LLMingua-2

LLMLingua-2 (Pan et al., 2024) was built to compress few-shot demonstrations and long contexts, not single short questions, so we apply it outside its intended use, as a standard learned input-compression baseline. We run it on three models (GPT-4o, Sonnet 4.6, Qwen2.5-VL-7B) and three datasets (GSM8K, BoolQ, ARC-Easy) at its default rate $\tau=0.5$, and at $\tau=0.8$ on Qwen2.5 to match the share of tokens our telegraphic level keeps (Figure 16).

The gap between accuracy and agreement appears under both methods, with $C_2 > 0$ on every model and benchmark. At the default rate GSM8K accuracy is 19–35%, against about 79% reported for demonstration compression (Pan et al., 2024). The drop comes from the rate, not the method. At $\tau=0.8$, LLMingua-2 on Qwen2.5 GSM8K

reaches 0.66 accuracy and 0.73 NLI, close to our telegraphic level.

Compressed prompt at $\tau=0.5$. Same GSM8K question (Janet’s ducks, gold answer 18) before and after LLMingua-2 compression. Articles, prepositions, and “how much” phrasing are removed. The $\tau=0.8$ check preserves 51 of 64 tokens. Per-cell numbers are in our repository.

Original prompt (64 tokens)

Janet’s ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers’ market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers’ market?

LLMLingua-2 at $\tau=0.5$ (30 tokens, 47% kept)

Janet ducks lay 16 eggs. eats three breakfast bakes muffins. sells remainder farmers’ market \$ 2 fresh duck egg. dollars?

Structural failure mode at the default rate.

LLMLingua-2 prunes single-letter MCQ labels (“A:”, “B:”, ...) as low-information tokens, which lowers ARC-Easy accuracy to 6–7% on all three models at $\tau=0.5$. CAVEWOMAN’s POS filter retains these labels in practice (all four letters survive in 99.7% of ARC-Easy L1 prompts), because spaCy tags them as nouns, which the L1 drop-list never removes.

J Within-Family Size Sweep

We run the full protocol on Qwen2.5-Instruct at 3B, 7B, 14B, and 32B, on the same 500 questions per benchmark, with one tokenizer and one set of prompts; Figure 7b adds the 0.5B and 1.5B models on the same questions. Accuracy is raw regex extraction, so every size is scored by the same extractor (Figure 18).

Under output compression, the L1 accuracy change rises from a 15-point loss at 3B to a small gain at 32B (Figure 18a), almost all of it on GSM8K, where the drop shrinks from 46 points to 4. Under input compression the drop stays between 10 and 16 points at every size. Below 3B the trend breaks down: the 0.5B model loses little because it has little accuracy to lose, and the 1.5B model’s small net change hides a 23-point GSM8K loss offset by gains elsewhere (Figure 7b). The trend

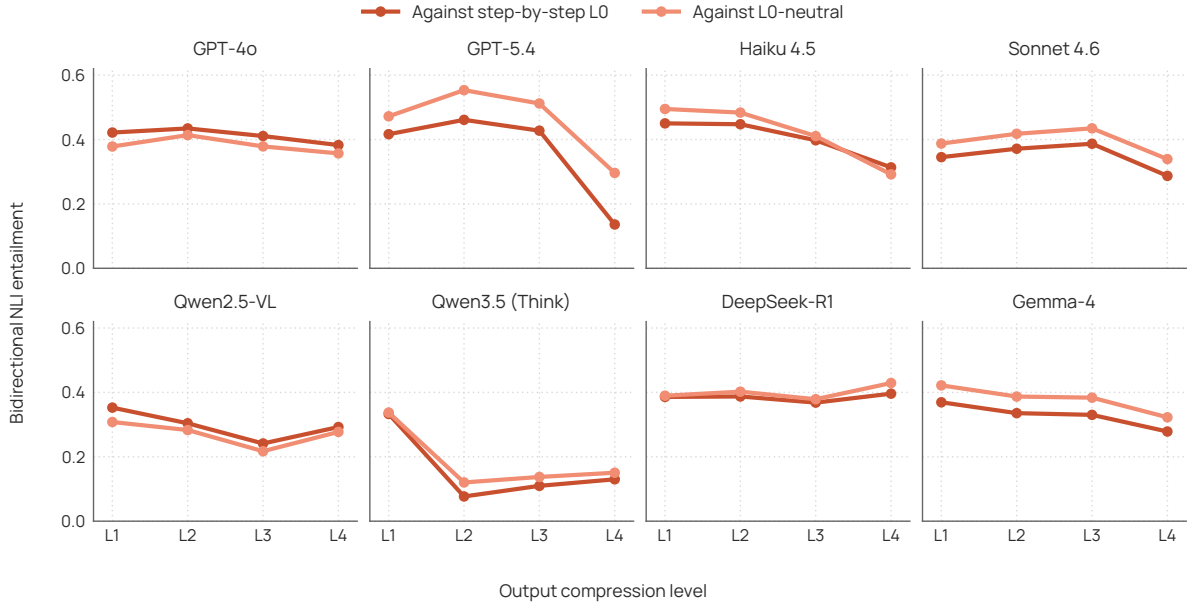


Figure 14: Each compression level scored against both the step-by-step L0 and the L0-neutral reference under the same bidirectional NLI criterion, n -weighted across the five benchmarks.

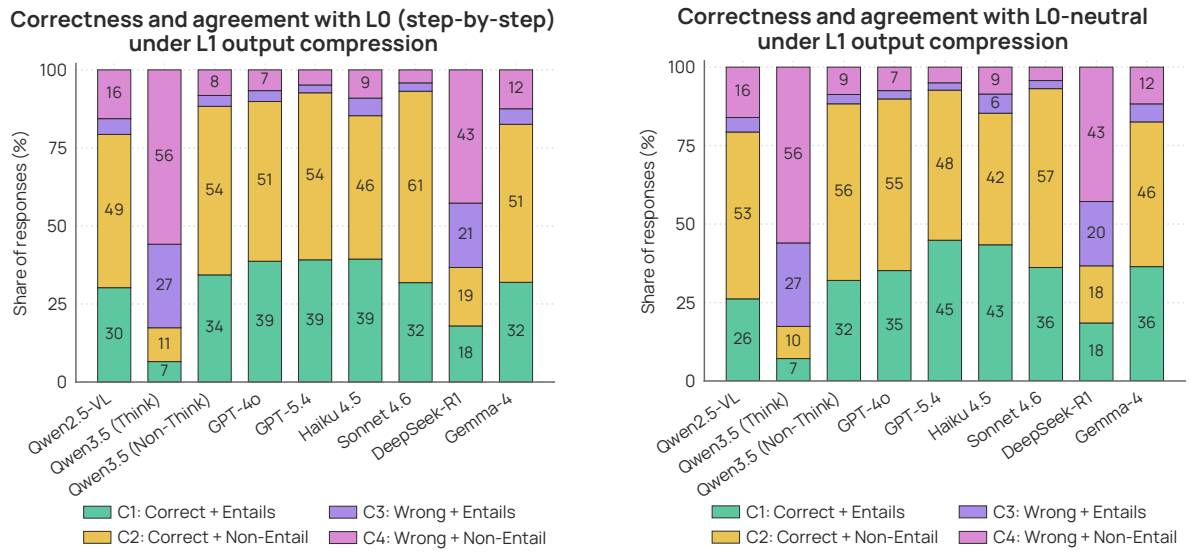


Figure 15: The four outcomes at L1 output compression against the two unconstrained references, L0 with the step-by-step instruction (left) and L0-neutral without it (right), for the eight main models and Qwen3.5-9B with thinking off. The main models use all 11,465 questions and raw extraction labels; the non-think bar uses its 500 questions per benchmark.

therefore holds only for models that are competent without compression. Entailment with L0 rises only from 28% at 3B to 41% at 32B (Figure 18b), so scale does not close the gap between accuracy and agreement. Output compression lowers cost at every size, by three to four times at the Haiku 4.5 price.

K Multilingual Generalization (Ten-Language MGSM)

We evaluate output compression (Condition B) on MGSM (Shi et al., 2023), using 250 parallel grade-school problems in each of ten languages spanning seven writing systems, on the two API models and two open-weight models, at all five reduction levels (Figure 19). Appendix B.4 gives the prompts

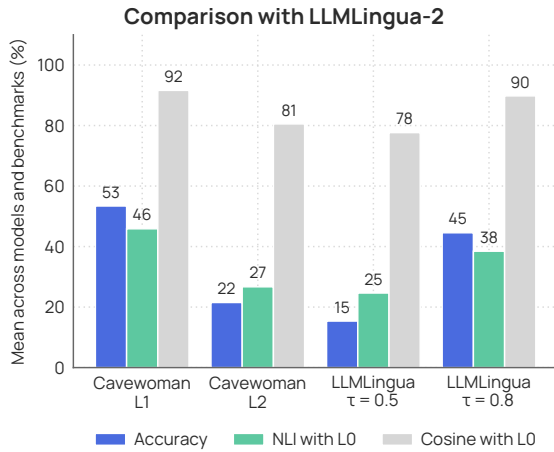


Figure 16: Mean accuracy, bidirectional NLI entailment against L0, and sentence-embedding cosine similarity with L0 for four input-compression configurations. The first three are averaged over the three models and three datasets the comparison was run on; the $\tau=0.8$ configuration was run on Qwen2.5-VL-7B only and is averaged over its three datasets.

and how they vary by language. The input channel is not run multilingually, since a faithful part-of-speech filter requires per-language taggers of uneven quality across these scripts.

Token cost by language. Figure 19 orders the languages by how many tokens the Qwen2.5 tokenizer assigns to a question relative to English on the same parallel items. Telugu costs more than six times the tokens of English and Bengali more than four, Thai about twice, Swahili and Russian half as much again, and the remaining five languages at most a third more. Per-token pricing therefore charges most for the first five.

Realized cost. On the two API models the cost of a question is below L0 at every level in every language, at L1 by 40–54% on GPT-4o and 35–44% on Claude Haiku. L2 is not always cheaper than L1, because L2 responses are longer than L1 responses in four of the ten languages on each model, and the reductions are smallest on Telugu and Bengali, whose compressed responses remain the longest. Total logged API spend for the multilingual evaluation was \$32.19.

Truncation. Under this prompt the unconstrained answers often run into the 400-token limit: Haiku on two thirds or more of the Thai, Bengali, and Telugu questions, Gemma-4 on a quarter to two thirds of the questions in every language, and Qwen2.5-VL on three quarters of the Telugu ques-

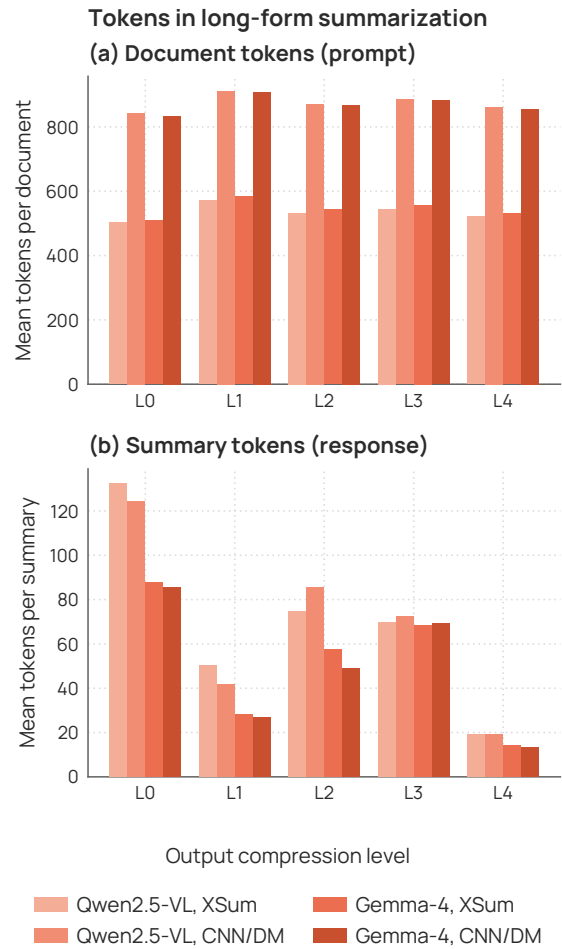


Figure 17: Token flows under register-constrained summarization at every reduction level, mean over 200 documents per model and task, with bar shade marking the model and task.

tions. Part of the L0 to L1 gain on these cells therefore comes from the limit no longer cutting the answer, not from better answers (hollow points in Figure 19).

Entailment. On the four languages scored for entailment (German, Swahili, Chinese, Bengali), L1 entailment with the unconstrained response averages 52% on Qwen2.5-VL and 63% on Gemma-4, below the sampling floor measured for the same models on the same questions on six of the eight model and language cells.

L Long-Form Generation (Summarization)

The main benchmarks have short, structured answers, so the Finding 2 gap between accuracy and agreement could be an artifact of terse outputs. We run register-constrained summarization

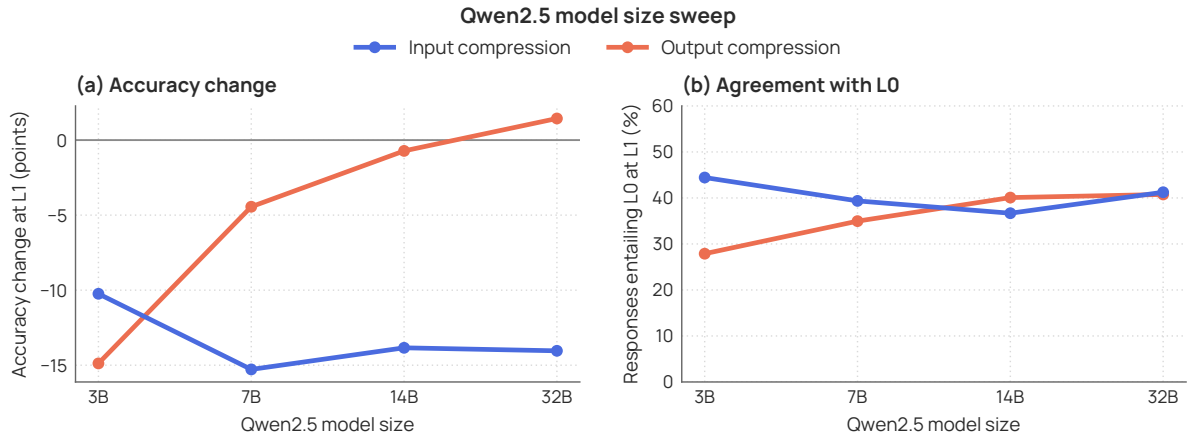


Figure 18: Qwen2.5-Instruct at four sizes, on the same 500 questions per benchmark, averaged over the five benchmarks. (a) Accuracy at L1 minus accuracy at L0, in points. (b) Share of L1 responses that entail the same size’s L0 response.

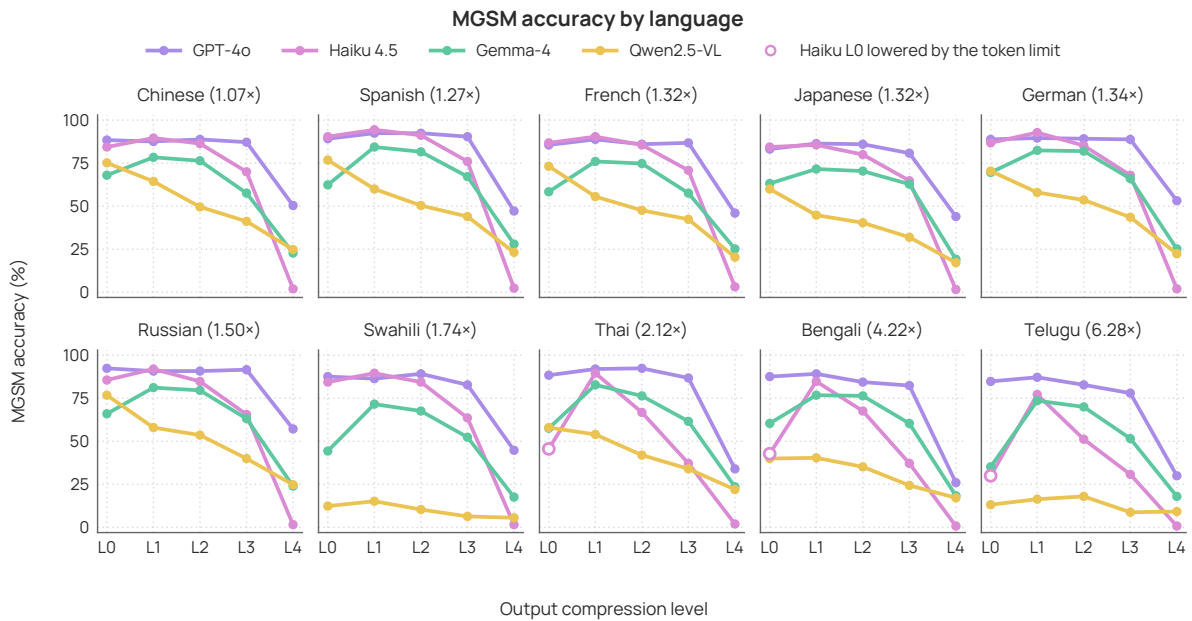


Figure 19: MGSM accuracy under output compression at every level, one panel per language, ordered by how many tokens a question costs relative to English (in parentheses). Accuracy is raw regex extraction on 250 problems per language. Hollow points mark Haiku’s L0 on Thai, Bengali, and Telugu, lowered by truncation at the 400-token limit (see text).

on XSum (Narayan et al., 2018) and CNN/DailyMail (Hermann et al., 2015), 200 documents each, on two open-weight models (Qwen2.5-VL-7B and Gemma-4-E4B), scoring each compressed summary for bidirectional NLI entailment against the model’s own unconstrained (L0) summary. Prompts, token limits, and source truncation are in Appendix B.3.

Entailment with the unconstrained summary never exceeds 4.5% at any level on any cell, while the summaries lose about two thirds of their tokens

at the telegraphic level (Figure 17). A sampling floor, from a second unconstrained summary drawn at temperature 0.7 and scored against the greedy one, is at least 2.3 times higher than every compressed level on every cell, so the gap is not a sampling artifact. Unlike the short-answer setting, where entailment falls level by level, it is already near zero at L1.

The register ladder is not monotone in length on long-form output. Every level is cheaper than

the unconstrained baseline, so the cost direction of Finding 1 carries over. But on all four cells the telegraphic L1 register gives the shortest summaries of the three intermediate levels, and the stricter keyword-only (L2) and noun-phrase (L3) registers give longer ones (Figure 17), which we read as list-style enumeration over a long source document. Only L4, which states a token budget rather than a register, is shorter than every other level on every cell. On these two models a style instruction did not control length; only the token limit did. A dedicated study of long-form and long-horizon agent settings is left to future work.

M Released Artifacts and Reproducibility

Code is at <https://github.com/danielle34/cavewoman> and data at <https://huggingface.co/datasets/rayascript/cavewoman-data>.

For every model, dataset, condition, and level we release the per-item records (extracted answer, gold label, token counts, cost, NLI and cosine scores) and the notebook and scripts that regenerate every table and figure.

N Use of AI Assistants

The authors acknowledge the use of ChatGPT and Claude solely for assistance with grammar, punctuation, and vocabulary corrections, as well as for supporting coding tasks. All scientific claims, experimental decisions, and analyses are the authors' own.