Not All Tokens Are Equally Important: Auditing and Compressing Redundancy in LLM Reasoning

Nguyen-Khang Le[1*], Dieu-Hien Nguyen[1*], Dinh-Truong Do[1], Nguyen Le Minh[1]

[1]Japan Advanced Institute of Science and Technology

Correspondence: {lnkhang, ndhien, nguyenml}@jaist.ac.jp

Abstract

Chain-of-thought traces drive strong LLM accuracy but dominate inference cost: output length sets both monetary cost and decoding latency. We ask which tokens in a reasoning trace actually carry the answer. A token-level analysis of the model's own next-token distribution identifies three forms of redundancy and their corresponding edits: discourse-element removal, tokenizer-aware abbreviation, and substitution of prose causal connectives with an arrow "$X \rightarrow Y$". All three leave the predictive distribution close to the original under Jensen–Shannon divergence and preserve the greedy next token in the vast majority of prefixes. We instantiate the analysis as CONCISON, a single training-free system prompt that operationalises the three rules and protects numeric, equation, and code tokens verbatim. Across 18 models from six families (2B–120B) and five reasoning benchmarks, CONCISON cuts output tokens by 24–62% on strong models at -1.1pp average accuracy, outperforms token-budget prompting in a single decoding call, and matches trained routers on the accuracy–cost frontier without any training, model pool, or extra infrastructure.

1 Introduction

Chain-of-thought (CoT) reasoning (Wei et al., 2022; Kojima et al., 2022) drives strong LLM performance on math, science, and code (OpenAI, 2024; Guo et al., 2025; Muennighoff et al., 2025), and test-time scaling laws (Snell et al., 2025) tie decoding compute directly to accuracy. The gains come at a practical cost: reasoning models routinely emit 1,000–3,000 intermediate tokens to reach a single final answer, and because API cost and decoding latency both scale linearly with output length – and output tokens are typically 3–5× more expensive than input – this verbosity is the dominant deployment cost, and the dominant contributor to user-visible latency.

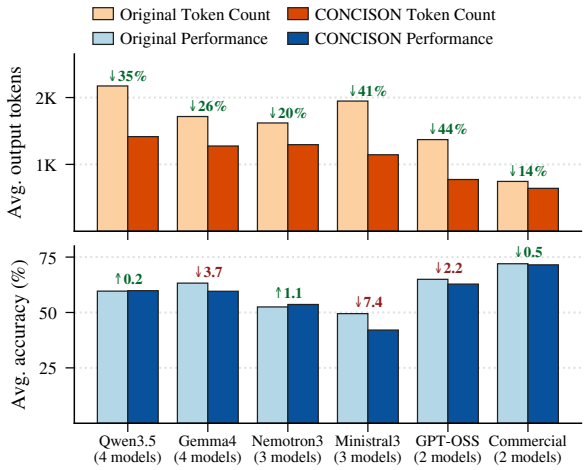


Figure 1: CONCISON cuts per-family average output tokens by 16–44% (and by 24–62% on individual strong models) while average accuracy stays within ±1 point. Numbers above bars are the headline % token change and accuracy delta vs. baseline; averaged over 18 models and 5 reasoning benchmarks.

Reasoning models often "overthink" (Chen et al., 2025; Sui et al., 2025), restating problems, hedging, or revisiting steps. Existing remedies fall into three families: training-based methods fine-tune or apply RL to reward brevity (Arora and Zanette, 2025; Aggarwal and Welleck, 2025; Kang et al., 2025; Xia et al., 2025); routing methods dispatch each query to a model of appropriate size (Chen et al., 2024; Ong et al., 2025; Ding et al., 2024; Madaan et al., 2024); and budget prompting estimates a token budget and passes it as guidance (Han et al., 2025; Nayab et al., 2024). Each pays a real cost – training data, a model pool, or a second decoding call – and treats the trace as a black box, shortening the answer without asking which parts of it were necessary.

We instead ask the more basic question: *which tokens in a reasoning trace actually carry the an-*

[*]These authors contributed equally to this work

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 5942–5962

November 6-10, 2026 ©2026 Association for Computational Linguistics

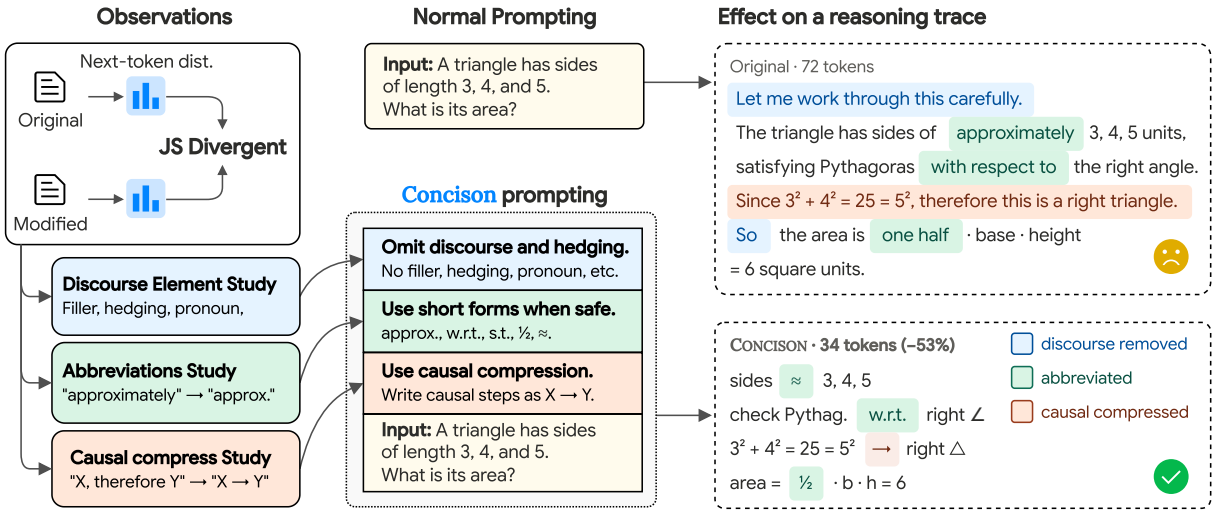


Figure 2: Overview. A token-level analysis (bottom left) identifies three forms of redundancy in reasoning traces; the Jensen–Shannon next-token framework (top left) confirms each transformation is distributionally safe; CONCISON (middle) operationalises them in a single training-free system prompt.

swer? A token-level analysis on the model's own predictive distribution surfaces three observations. (i) Discourse positions – fillers, hedges, articles, non-load-bearing connectives – sit at the high-entropy end of the distribution; counter-intuitively, high entropy marks where the model is least committed, not where the token carries information. (ii) Abbreviation is token-efficient only on the per-tokenizer *useful subset* $A_{\mathcal{M}}$ of a candidate dictionary, since the same word costs three tokens under one tokenizer and one under another. (iii) Prose causal connectives (*so*, *therefore*, *because*) appear in sentences accounting for 20–30% of trace tokens, and compact to "$X \rightarrow Y$" (a single token under every tokenizer we test). To promote each observation into a method, we measure the corresponding transformation against a model-agnostic safety criterion – Jensen–Shannon divergence between the model's next-token distributions on the original and edited prefix – and find all three within the noise floor.

Because each form of redundancy lives inside the base model's own distribution, no fine-tuning is needed: we ask the model to omit the redundancy via its system prompt. The resulting CONCISON prompt operationalises the three rules and protects numbers, equations, code, identifiers, and negation as a hard content invariant. The same template is sent to every model; only the spliced-in abbreviation table $A_{\mathcal{M}}$ varies. Across 18 models (2B–120B, six families) and five reasoning benchmarks, CONCISON cuts output tokens by 24–62% on strong models at -1.1pp average accuracy, dominates token-budget prompting in a single decoding call, and matches trained routers on the accuracy–cost frontier (Figure 1).

Contributions. (1) A token-level redundancy analysis of reasoning traces tied to Jensen–Shannon next-token divergence as a measurable, model-agnostic safety criterion (§3–§4). (2) CONCISON: a training-free, single-call, model-agnostic system prompt with a hard content invariant and three nested operating points (§5). (3) Across 18 models and 5 benchmarks (§7), CONCISON attains the highest token efficiency on every strong model (+23% vs. the next-best baseline on average) and matches the accuracy–cost frontier of four trained routers without any training.

2 Problem Formulation

Let $\mathcal{M}$ be an autoregressive language model that defines $p_{\mathcal{M}}(y \mid x) = \prod_{t=1}^{|y|} p_{\mathcal{M}}(y_t \mid x, y_{<t})$ for a response $y = (y_1, \ldots, y_{|y|})$ given context x. Given a query q and a system prompt s, the model produces $y \sim \mathcal{M}(\cdot \mid s \oplus q)$ (i.e. $x = s \oplus q$).

Cost. The serving cost of a response is governed by its output length $|y|$: monetary cost is $r_{\text{out}}|y| + r_{\text{in}}|x|$ with output rate $r_{\text{out}} \approx 3$–$5\, r_{\text{in}}$, and decode latency is $\approx |y| \cdot \tau$ for per-token decode time τ. As the query is fixed, $|y|$ is the dominant controllable cost; we report it as the primary metric.

Objective. Let y^{base} be the response under a standard chain-of-thought system prompt s^{base}. A token-reduction method supplies an alternative s

inducing y. We define the token reduction

$$\rho \;=\; 1 - \frac{\mathbb{E}_q[\,|y|\,]}{\mathbb{E}_q[\,|y^{\text{base}}|\,]}, \qquad (1)$$

and seek to maximize ρ subject to an accuracy constraint $\mathbb{E}_q\big[\mathbb{1}[\hat{a}(y) = a^\star]\big] \geq \mathbb{E}_q\big[\mathbb{1}[\hat{a}(y^{\text{base}}) = a^\star]\big] - \epsilon$. The remainder of the paper first asks which tokens can be removed without violating this constraint (§3), then designs s accordingly (§5).

3 Are Reasoning Traces Redundant?

Length is not information. We ask whether the surface tokens of a trace carry uniform predictive weight along three axes: per-token entropy (§3.1), tokenizer-aware abbreviation (§3.2), and causal compaction (§3.3). The resulting candidates are validated in §4.

3.1 Discourse Element Redundancy

Predictive entropy signals how constrained a position is: low entropy means the model is committed; high entropy means many tokens are roughly equiprobable. At every position t of a baseline trace we estimate the Shannon entropy (Shannon, 1948) $H_t = -\sum_v p_{\mathcal{M}}(v \mid q, y_{<t}) \log p_{\mathcal{M}}(v \mid q, y_{<t})$ from top-K logprobs over 16 self-hosted open-source models ($\approx$0.44M generated tokens at $T=0$) and tag each token with its lexical category.

Entropy marks interchangeability, not information. The intuitive reading "information at this position" is inverted by the data (Figure 3): high-entropy positions concentrate on discourse alternatives ("Since", "Because", "As"), where many surface forms yield the same continuation; low-entropy positions are numbers, operators, and formula tokens forced by the math on the page. Interchangeable tokens are the candidates for removal.

Cross-category confirmation. Binning tokens by entropy and plotting per-bin lexical composition (Figure 4) confirms the pattern: math symbols and numeric values dominate low-entropy bins, while the share of hedges, fillers, and conjunctions grows monotonically with entropy. The ordering holds on every model (Appendix B).

3.2 Word Redundancy and Abbreviations

For words that cannot be removed, abbreviation is the simplest compression. We curate a dictionary of 155 word/phrase pairs (e.g., `function`→`fn`) and scan 5,000 FineWeb-Edu documents (Penedo et al., 2024): 1.19% of words are dictionary-convertible on average, with a long right tail (Figure 5a).

Savings depend on the tokenizer. "calculation" is one token under some BPE vocabularies and two or three under others; abbreviating to "calc" is only useful in the latter case (Figure 5b). We therefore restrict abbreviation to the per-tokenizer *useful subset* $A_{\mathcal{M}} = \{(w, a) : |\text{tok}_{\mathcal{M}}(w)| > |\text{tok}_{\mathcal{M}}(a)|\}$, which retains 32–71 pairs depending on the model.

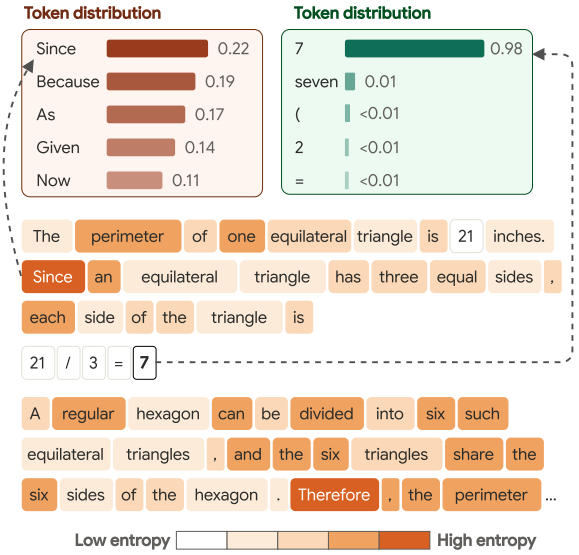


Figure 3: Per-token predictive entropy on a real reasoning chain (darker = higher).

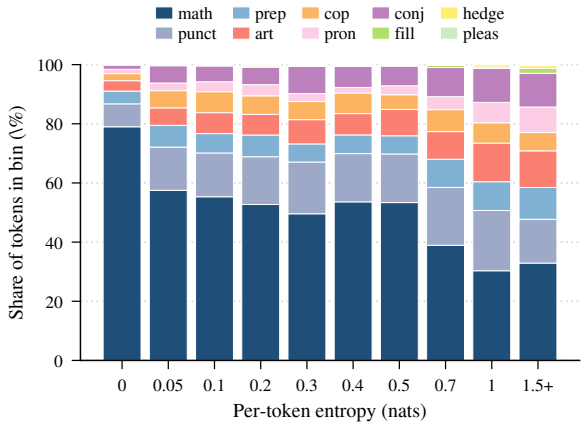
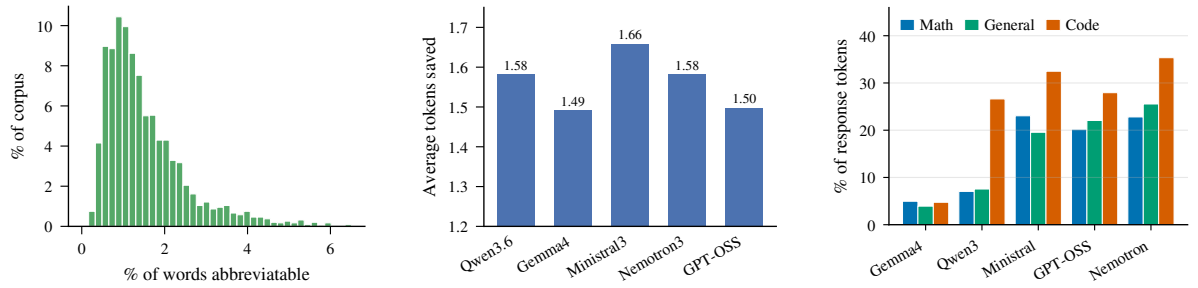


Figure 4: Lexical composition of generated tokens within predictive-entropy bin (each bar sums to 100%).

3.3 Causal Compression

A reasoning trace strings intermediate conclusions with causal connectives (*so*, *therefore*, *because*, *hence*, *thus*). Each expresses a fixed two-place relation but consumes several tokens, and prose frames ("This gives us…") add overhead. Across math (GSM8K, MATH-500, AIME), code (Live-



(a) % words convertible to abbreviation per FineWeb-Edu doc.

(b) Mean tokens saved per dictionary entry by tokenizer.

(c) Response-token share in sentences with a causal connective.

Figure 5: Abbreviation and causal-connective addressability. (a, b) quantify the abbreviation opportunity in raw text; (c) shows the trace-token share targeted by the arrow-form rewrite.

CodeBench), and science (GPQA-Diamond), 20–30% of trace tokens lie in sentences with such a connective (Figure 5c). A natural compaction rewrites the construction as "$X \rightarrow Y$" ("$X \leftarrow Y$" backward), collapsing several tokens into one arrow while preserving dependency structure.

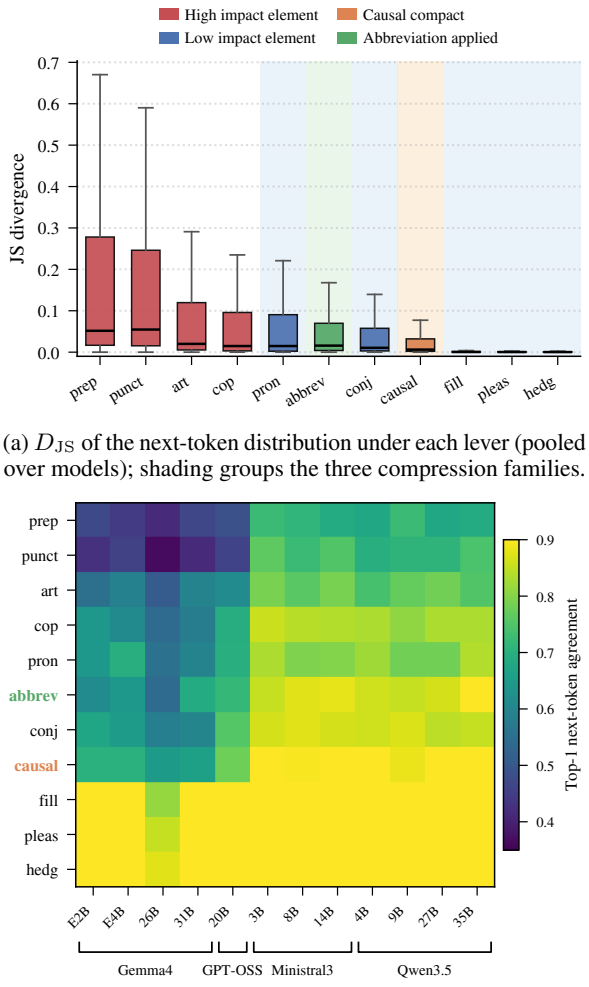
4 Retaining the Next-Token Distribution

The levers in §3 are only useful if applying them is *safe*: the model's next-token distribution on the modified prefix should match its distribution on the original. We measure this with a single framework applied to all three levers.

Framework. For a text transformation f and a prefix x, let $P = p_{\mathcal{M}}(\cdot \mid x)$ and $P_f = p_{\mathcal{M}}(\cdot \mid f(x))$. We sample 200 prefixes (≤ 500 chars) from FineWeb (Penedo et al., 2024), trimmed at the last whitespace, and report the Jensen–Shannon divergence (Lin, 1991) $D_{\mathrm{JS}}(P\|P_f) = \frac{1}{2}D_{\mathrm{KL}}(P\|\bar{P}) + \frac{1}{2}D_{\mathrm{KL}}(P_f\|\bar{P})$ with $\bar{P} = \frac{1}{2}(P + P_f)$, alongside *top-1 agreement* – the fraction of prefixes with $\arg\max P = \arg\max P_f$.

Levers. $f_{\mathrm{disc},e}$ deletes every token of category e, swept over removable categories (pronoun, conjunction, article, copula, filler, pleasantry, hedging, preposition, punctuation) and load-bearing controls (number, noun, verb); f_{abbrev} substitutes dictionary words; f_{causal} locates sentences with a causal connective via regex, extracts X/Y, and rewrites them as "$X \rightarrow Y$".

Results (Figure 6). All three safe lever families sit at the low end of the D_{JS} spectrum and preserve the greedy next token in 82–90% of prefixes across every model family, while the load-bearing controls shift D_{JS} an order of magnitude more and



(a) D_{JS} of the next-token distribution under each lever (pooled over models); shading groups the three compression families.

(b) Top-1 next-token agreement per (lever, model); bright = greedy token preserved.

Figure 6: Compression levers on FineWeb prefixes: distributional cost (a) and per-cell top-1 next-token agreement across model families (b).

flip the argmax on most prefixes. Abbreviation matches discourse removal; the arrow substitution costs slightly more D_{JS} than discourse removal –

comparable to dropping a single function word – but stays an order of magnitude below the load-bearing band.

From distributional safety to trace-level safety. The FineWeb framework above is observational and uses general text, which leaves open whether the same edits survive on a reasoning trace under autoregressive continuation. We replicate the discourse-removal claim causally with a strip-truncate-continue experiment (Appendix D): for a baseline trace, we delete the tokens in each candidate discourse category and force the model to continue from the edited prefix. Accuracy stays within 5–8pp of the unedited baseline across nested strip sets and truncation depths on three models; a numeric-strip control collapses 15–47pp. Together with the FineWeb D_{JS} evidence, the three observations are promoted from structural to measurable, model-agnostic safety properties.

5 CONCISON Prompting

§3–§4 characterise a *safe compression* of a reasoning trace: drop high-entropy discourse, shorten substitution-robust surface forms, never touch answer-bearing tokens. Existing methods turn such characterisations into trained rewriters (Arora and Zanette, 2025; Kang et al., 2025; Xia et al., 2025), which add data, weights, and a per-model derivation. Because the redundancy lives inside the base model's own distribution, we instead ask the model to omit it via its system prompt. We call the resulting prompt CONCISON. Distributional safety holds uniformly across the 16 open-source models whose APIs expose top-K logprobs (§4), so a single template suffices; only the abbreviation table $A_{\mathcal{M}}$ varies per model.

Design principles. We distil the analysis into three principles that the prompt operationalises.

(P1) Drop high-entropy discourse. Fillers, hedges, pleasantries, articles, subject pronouns, copulas, and non-load-bearing connectives produce near-noise-floor D_{JS} under removal and are the dominant source of length reduction.

(P2) Substitute verbose surface forms. Long words map to their $A_{\mathcal{M}}$ entries; prose causal constructions ("X, so Y") become "$X \rightarrow Y$" ("$X \leftarrow Y$" backward). Both preserve next-token prediction while collapsing several tokens into one.

(P3) Preserve content verbatim. Numbers, equations, code, identifiers, units, and negation are load-bearing – removing them flipped the argmax in most prefixes – and form the prompt's hard invariant.

The CONCISON prompt. The prompt has three parts (verbatim in Appendix A; sketched below):

> **Preamble** (P3 + style): solve step by step; numbers, equations, code, technical terms must remain exact; style is maximum compression.
>
> **Rules** (P1 + P2, apply whenever applicable): drop articles / fillers / pleasantries / subject pronouns / copulas / non-load-bearing connectives; apply $A_{\mathcal{M}}$ inline (`function`→`fn`, `equation`→`eq`, ...; 32–71 pairs depending on the model's tokenizer); write derivations as "[fact] $\rightarrow$ [step] $\rightarrow \ldots \rightarrow$ ans $= X$", using "$\leftarrow$" for backward dependencies; sentence fragments are permitted.
>
> **Exemplar:** one worked problem with a baseline rewrite and a CONCISON rewrite, both ending in $\boxed{\text{ans}}$.

The exemplar is not a few-shot demonstration; it disambiguates "drop", "shorten", and "preserve verbatim" more concretely than imperative rules alone. Figure 9 shows the resulting compression.

Properties. The prompt template is identical across all 18 models – only the spliced-in $A_{\mathcal{M}}$ varies, and adapting to a new model is one pass over the 155-pair dictionary. A single decoding call suffices, unlike TALE-EP (Han et al., 2025) or rewriter pipelines. The prompt is $\sim$300 tokens vs. $\sim$ 30 for CoT, but this overhead is paid on input (3–5$\times$ cheaper than output), constant per query, and absorbed by prefix caching; output savings exceed it by an order of magnitude. Three nested variants (L1 $\subset$ L2 $\subset$ CONCISON) expose a smooth compression–accuracy trade-off used in §7.2.

6 Experimental Setup

Models. 18 models from 2B to 120B: Qwen3.5 (4/9/27B) (Yang et al., 2025) and Qwen3.6 (35B-A3B), Gemma4 (2/4/26/31B) (Gemma Team, 2025), Nemotron3 (4/30/120B) (NVIDIA et al., 2025), Ministral3 (3/8/14B) (Liu et al., 2026), gpt-oss (20/120B) (OpenAI, 2025), and two commercial models (GPT-4.1, GPT-5-mini).

Benchmarks. Five reasoning benchmarks – AIME (Veeraboina, 2023), MATH-500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), GPQA-Diamond (Rein et al., 2024), and LiveCodeBench (Jain et al., 2025) – run in full (30/500/1,319/198/175 = 2,222 problems per model per setting).

Baselines. Our reference is **Baseline (CoT)**, which asks the model to reason step by step.

Against it we compare two prompting baselines: **Plan-and-Solve** (Wang et al., 2023), which extracts variables then solves, and **TALE-EP** (Han et al., 2025), which estimates a token budget in a first call and passes it as guidance in a second. Four routing baselines, each routing between the smallest and largest model within a family: **RouteLLM** with the pretrained BERT-classifier (Ong et al., 2025), **AutoMix** (Madaan et al., 2024), **MFRouter** (matrix factorisation), and **SW-Ranking** (similarity-weighted ranking) following Ong et al. (2025).

Metrics. We report task accuracy (**Acc.**) and **Token Efficiency** (**Eff.**), our primary cost/quality metric:

$$\text{Eff.} = \frac{\text{Acc. (\%)}}{(\text{mean output tokens}/1000) \times N(\mathcal{M})},$$

where $N(\mathcal{M})$ is the model's total parameter count in billions (approximating per-token decode FLOPs; higher is better). For MoE models N overstates compute by the inactive ratio, so the *Eff.* numbers we report for MoE models (Qwen3.6-35B-A3B, GPT-OSS-120B) are conservative; using active parameters would inflate them. Critically, this choice does not affect any within-model comparison between methods, since the same N enters the numerator and denominator for every row of a given model block. We also report raw token counts throughout. For routing, we use a size-weighted token count (mean tokens $\times N$) for the same reason. Per-benchmark and per-(model, dataset) breakdowns are in Appendix I.

Implementation details. Greedy decoding ($T = 0$, $\text{top}_p = 1$, max 4,096 tokens); deltas computed on matched-sample intersections. Self-hosted models use vLLM/SGLang on H100/A6000; commercial models use the official API. Code is scored via the LiveCodeBench sandbox; math/MCQ via the official extractor. Full details in Appendix H.

7 Results and Analysis

7.1 Main Results

Comparison with prompting baselines. Across all four strong models, CONCISON attains the highest Token Efficiency and the largest token reduction (-24% to -62% vs. baseline; Table 1). Pooled, its mean Eff. of 1.28 leads the next-best baseline (TALE-EP, 1.04) by 23% and Plan-and-Solve (0.84) by 52%, in a single decoding call. The per-(model×benchmark) breakdown (Appendix I) shows CONCISON winning the Eff. entry in 16/20 cells, with the largest gains where baseline traces are most padded (GSM8K). Plan-and-Solve does modestly improve accuracy on three of four models but at a token-cost premium of up to $+28\%$, so it loses on Eff.; TALE-EP reduces tokens but pays for a second decoding call. Accuracy is not uniformly preserved: AIME and LiveCodeBench are essentially unchanged on the strong-model average and GSM8K stays within -2pp, but GPQA-Diamond can drop by up to 11pp on individual models (GPT-OSS-120B). The accuracy cost concentrates on GPQA-Diamond and on small/mid-size models, both analysed in §7.2.

Method	Acc.	Tokens	Reduced	Eff.
Qwen3.6-35B-A3B				
Baseline (CoT)	61.2	2082	–	0.84
Plan-and-Solve	62.9	2189	↑5%	0.82
TALE-EP†	**64.3**	1666	↓20%	1.10
CONCISON	62.7	**1590**	↓24%	**1.13**
Gemma4-31B-it				
Baseline (CoT)	**75.8**	1110	–	2.20
Plan-and-Solve	77.1	1419	↑28%	1.75
TALE-EP†	69.3	999	↓10%	2.24
CONCISON	73.4	**839**	↓24%	**2.82**
Nemotron-3-Super-120B				
Baseline (CoT)	55.8	2275	–	0.21
Plan-and-Solve	**60.4**	2148	↓6%	0.23
TALE-EP†	59.3	1761	↓23%	0.28
CONCISON	59.7	**1598**	↓30%	**0.31**
GPT-OSS-120B				
Baseline (CoT)	**68.8**	1675	–	0.34
Plan-and-Solve	67.6	1015	↓39%	0.55
TALE-EP†	62.8	965	↓42%	0.55
CONCISON	65.8	**644**	↓62%	**0.85**

Table 1: Main results, averaged over the five benchmarks (AIME, MATH-500, GSM8K, GPQA-Diamond, LiveCodeBench). *Reduced*: signed % token change vs. baseline (blue/red = down/up). *Eff.* = Acc. (%) / (KTok×params B); higher is better. Best per column per block in **bold**. Per-benchmark breakdown and per-(model, dataset) token counts: Appendix I. †TALE-EP spends a separate budget-estimation call.

Comparison with routing baselines. Figure 7 averages accuracy and size-weighted token cost (mean tokens × params) across families and benchmarks. CONCISON reaches the always-large baseline's accuracy at a fraction of its weighted cost and sits above-left of every router's swept curve, matching the best router operating point without any train-

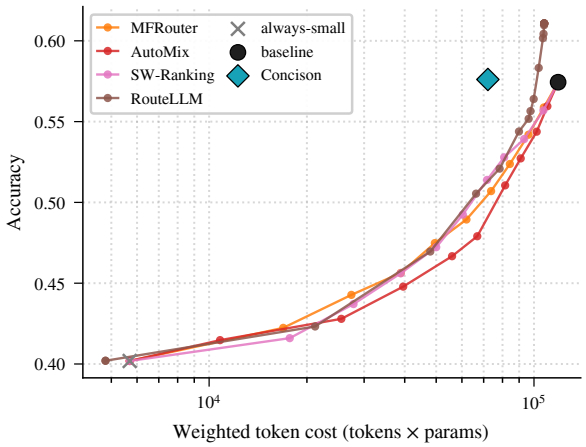


Figure 7: Accuracy vs. size-weighted token cost, averaged across all model families and benchmarks. Routers sweep a threshold (curves); CONCISON and the always-small/large baselines are single points.

Variant	Token reduct.	Acc.	ΔAcc.
Baseline (CoT)	n/a	58.0	n/a
CONCISON-L1	18.1%	58.0	−0.0
CONCISON-L2	37.9%	58.3	+0.2
CONCISON	42.4%	57.1	−1.1

Table 2: Ablation on 6 strong models, 5 benchmarks. The variants nest in aggressiveness (L1 ⊂ L2 ⊂ CONCISON).

ing, router, or model pool. The per-(family, benchmark) breakdown (Appendix K) shows the above-left placement holds in 17 of 20 cells; the three exceptions (Gemma-GPQA, Gemma-LCB, GPT-OSS-GSM8K) are benchmarks where the small model already approaches the large one. The two ideas are complementary rather than competing: a router can dispatch to models that themselves run CONCISON.

7.2 Analysis

Ablation: concision level. The three nested variants (Table 2) trace a smooth compression–accuracy frontier: token reduction grows monotonically from 18% to 42% while accuracy drops by at most 1.1pp. CONCISON-L2 is accuracy-neutral on average (+0.2pp) at 38% reduction – already exceeding TALE-EP in a single decoding pass.

Effect of model capability. Table 3 reports per-model token reduction and accuracy change for all 18 models grouped by family. Tokens drop on 17 of 18 models – typically by 20–45%, and up to 62% on GPT-OSS-120B (Nemotron-Nano-30B is the lone exception, where CONCISON expands rather

Family	Model	ΔTok	ΔAcc
Qwen3.5	4B	−37%	−2.0
	9B	−34%	−0.6
	27B	−44%	+1.9
	35B-A3B	−24%	+1.5
Gemma4	E2B	−20%	−5.5
	E4B	−39%	−4.8
	26B-A4B	−18%	−2.0
	31B	−24%	−2.4
Nemotron3	Nano-4B	−18%	+1.0
	Nano-30B	+16%	−1.4
	Super-120B	−30%	+3.8
Ministral3	3B-R	−35%	−15.3
	8B-R	−40%	−2.9
	14B-R	−47%	−4.1
GPT-OSS	20B	−15%	−1.3
	120B	−62%	−3.0
Commercial	GPT-4.1	−24%	−3.0
	GPT-5-mini	−1%	+2.0

Table 3: Per-model token reduction and accuracy change (pp) for CONCISON vs. baseline, averaged over the five benchmarks (matched samples). Blue/red in ΔTok mark reductions/increases. ΔAcc cells are colour-shaded: green = gain, red = loss, darker = larger magnitude. The accuracy/size relationship is plotted in Figure 8.

than compresses the trace); the accuracy cost of the full rule set scales inversely with model capability. Within every family, larger models absorb the dense instruction better: Qwen3.5 swings from −2.0pp at 4B to +1.5pp at 35B, Ministral from −15.3pp at 3B to −3 to −4pp at 8B/14B, and Nemotron-120B and GPT-5-mini gain (+3.8, +2.0pp). Gemma4 stays slightly negative but its worst drops are at 2B/4B. Aggressive compression is most reliable on models with $\gtrsim$30B parameters; smaller models can fall back to L1 or L2.

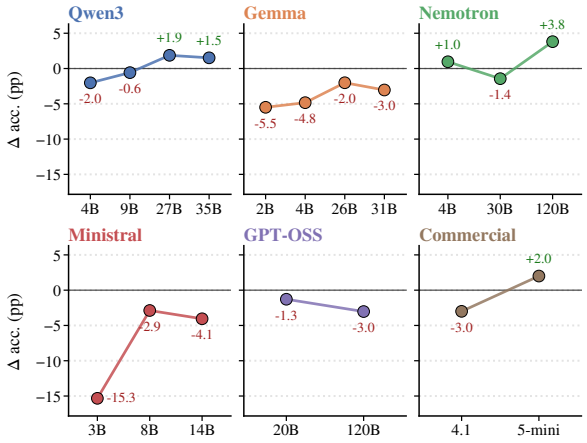


Figure 8: Per-model ΔAcc. (CONCISON − baseline) by family and size; markers above $y=0$ are gains.

Qualitative example. Figure 9 illustrates the mechanism on a GSM8K problem both methods solve. The baseline restates the question and narrates in full sentences; CONCISON keeps every number, operation, and the boxed answer but drops scaffolding, reaching the same answer in $\sim 4\times$ fewer tokens. The same pattern recurs across benchmarks.

GSM8K problem. Paul needs 63 cupcakes for a party; he already has 8 chocolate and 40 toffee cupcakes. How many more must he buy?

Baseline, 159 output tokens.
"To find out how many more cupcakes Paul needs to buy, we can follow these steps: (1) Determine the total number of cupcakes Paul already has: Paul has 8 chocolate cupcakes and 40 toffee cupcakes. $8 + 40 = 48$ cupcakes. (2) Determine the total number of cupcakes needed for the party: Paul needs a total of 63 cupcakes. (3) Calculate the difference between the number needed and the number he already has: $63 - 48 = 15$. Paul needs to buy 15 more cupcakes." $\boxed{15}$

CONCISON, 40 output tokens.
"Total req $= 63 \rightarrow$ Current $= 8 + 40 = 48 \rightarrow$ Need $= 63 - 48 = 15$. ans $= 15$." $\boxed{15}$

Figure 9: Baseline vs. CONCISON response (Gemma-31B, GSM8K); both correct.

Further analyses. Verbatim prompts (App. A), per-benchmark tables (App. I), per-family bar plots (App. J), and per-lever cross-model breakdowns (App. B–G, including a strip-truncate-continue causal control) are deferred to the appendix.

8 Related Work

Reasoning and its cost. Chain-of-thought prompting (Wei et al., 2022; Kojima et al., 2022) and, more recently, models trained to reason at length (OpenAI, 2024; Guo et al., 2025) improve accuracy on hard tasks, and test-time scaling laws relate compute to accuracy (Snell et al., 2025; Muennighoff et al., 2025). The same length, however, is the cost: Sprague et al. (2025) show chain-of-thought helps mainly on mathematical and symbolic tasks, and Chen et al. (2025) document systematic "overthinking" in o1-style models.

Efficient reasoning. Sui et al. (2025) group efforts to shorten reasoning into training-based and inference-based families. Training-based methods fine-tune or apply RL to reward brevity (Arora and Zanette, 2025; Aggarwal and Welleck, 2025; Kang et al., 2025; Xia et al., 2025); they are effective but require data and per-model training. Inference-based methods change the prompt or decoding: Nayab et al. (2024) and Renze and Guven (2024) show generic "be concise" instructions help, while Han et al. (2025) (TALE-EP) estimate a token budget in a first call and pass it as soft guidance in a second. Two things distinguish CONCISON from this prior work. First, its rules are derived from a measurable token-level redundancy analysis with a model-agnostic safety criterion (§3–§4), rather than introspective heuristics; the same analysis tells us *what* to drop and *what to protect*. Second, it composes three independently-validated levers in a single decoding call – the arrow rewrite and the tokenizer-aware abbreviation table $A_{\mathcal{M}}$ have, to our knowledge, no prior use in CoT compression – and is validated at the scale of 18 models.

Routing and cascades. An orthogonal line reduces cost by sending each query to a model of appropriate size: FrugalGPT (Chen et al., 2024), RouteLLM (Ong et al., 2025), Hybrid LLM (Ding et al., 2024), and AutoMix (Madaan et al., 2024). These need a model pool, a trained router, and extra infrastructure. CONCISON is complementary: it reduces the cost of whichever model is used, and we show it is competitive with trained routers on the accuracy-cost frontier without any training (§7.1).

Token importance. That tokens contribute unequally is implicit in work on pruning, KV-cache eviction, and saliency. We make the point directly for reasoning traces, tying redundancy to next-token distribution shift and to predictive entropy (Shannon, 1948; Lin, 1991), and we use the analysis to design a method.

9 Conclusion

We asked which tokens in a reasoning trace carry the answer. Three token-level observations – discourse redundancy, tokenizer-aware abbreviation, and compactable causal connectives – ground a measurable Jensen–Shannon safety criterion and motivate CONCISON, a training-free single-call system prompt that cuts output tokens by 24–62% on strong models at -1.1pp average accuracy and matches trained routers without any router, model pool, or training. The observations themselves give a concrete account of where reasoning traces are compressible – a property of the base model's own distribution, accessible at inference time.

Limitations

First, the next-token-divergence experiments (§4) use FineWeb prefixes rather than reasoning traces because they require controlled text edits; the strip-truncate-continue experiment (Appendix D) is the trace-level causal complement on three models. The entropy analysis covers 16 open-source models – commercial APIs lack top-K logprobs. Second, our benchmarks are math, science, and code, where answers are verifiable; CONCISON may behave differently on open-ended generation, dialogue, or safety-sensitive tasks where discourse tokens carry pragmatic content. Third, the full rule set costs accuracy on small ($\leq$4B) and some mid-size models; L1/L2 mitigate this but do not eliminate it. Fourth, evaluation uses greedy decoding; interactions with sampling and self-consistency are untested. Finally, the method is prompt-based and inherits the usual prompt sensitivity, though the underlying redundancy observations should be more stable across model generations than the absolute numbers.

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling how long a reasoning model thinks with reinforcement learning. In *Conference on Language Modeling (COLM)*.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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*.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 9487–9499. PMLR.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations (ICLR)*.

Gemma Team. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638.

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.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations (ICLR)*.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3oT: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Jianhua Lin. 1991. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, and 101 others. 2026. Ministral 3. *Preprint*, arXiv:2601.08584.

Aman Madaan, Pranjal Aggarwal, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Mausam, and Manaal Faruqui. 2024. AutoMix: Automatically mixing language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 20275–20321.

Sania Nayab, Giulio Rossolini, Marco Simoni, Andrea Saracino, Giorgio Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli. 2024. Concise thoughts: Impact of output length on LLM reasoning and cost. *arXiv preprint arXiv:2407.19825*.

NVIDIA, :, Aaron Blakeman, Aaron Grattafiori, Aarti Basant, Abhibha Gupta, Abhinav Khattar, Adi Renduchintala, Aditya Vavre, Akanksha Shukla, Akhiad Bercovich, Aleksander Ficek, Aleksandr Shaposhnikov, Alex Kondratenko, Alexander Bukharin, Alexandre Milesi, Ali Taghibakhshi, Alisa Liu, Amelia Barton, and 340 others. 2025. Nvidia nemotron 3: Efficient and open intelligence. *Preprint*, arXiv:2512.20856.

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 with preference data. In *International Conference on Learning Representations (ICLR)*.

OpenAI. 2024. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*.

OpenAI. 2025. gpt-oss-120b and gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *Conference on Language Modeling (COLM)*.

Matthew Renze and Erhan Guven. 2024. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pages 476–483. IEEE.

Claude E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations (ICLR)*.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2025. To CoT or not to CoT? chain-of-thought helps mainly on math and symbolic reasoning. In *International Conference on Learning Representations (ICLR)*.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*.

Hemish Veeraboina. 2023. AIME Problem Set 1983–2024. https://www.kaggle.com/datasets/hemishveeraboina/aime-problem-set-1983-2024. Kaggle dataset.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2609–2634.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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 (EMNLP)*, pages 3351–3363.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Appendix Overview

The appendix is organised as follows. **§A** gives the verbatim CONCISON system prompt and the three nested ablation variants used in §7.2. **§B** extends the entropy analysis of §3.1 to all sixteen open-source models in our sweep, both as a small-multiples grid and as a pooled boxplot. **§C** reports the per-(lever, model) top-1 next-token agreement that backs the pooled JS-divergence claim in §4. **§D** presents the strip-truncate-continue experiment that converts the observational JS-divergence claim

into a causal one: the model continues autoregressively from a trace whose discourse tokens have been deleted, and still reaches the right answer. **§E** reports per-tokenizer abbreviation savings and the filtering rule that constructs $A_{\mathcal{M}}$. **§F** provides per-model statistics for the arrow-substitution lever. **§G** documents the prefix word-boundary preprocessing step common to all FineWeb-prefix experiments. **§H** lists serving and decoding details for reproducibility. **§I** expands the averaged main-body Table 1 into the per-benchmark accuracy / efficiency table (Table 6) and the per-(model, dataset) output-token table (Table 7). **§J** reports per-family output-token and accuracy bar plots (one figure per model family) in the style of Figure 1. **§K** gives the per-(family, benchmark) accuracy–cost Pareto grid that backs the averaged Figure 7.

A CONCISON Prompt Family

Prompt structure. Every prompt in the family shares the same skeleton: a preamble stating the task and content invariant (**P3**), a rule block operationalising **P1** (discourse removal) and **P2** (surface compression), and a single good/bad exemplar. Three variants nest in aggressiveness (§7.2).

Variants. Table 4 summarises which rules each variant includes. CONCISON-L1 retains only the safest, lowest-D_{JS} subset of **P1**; CONCISON-L2 adds the remaining **P1** rules and the causal-arrow part of **P2**; the full CONCISON additionally splices in the per-tokenizer useful subset $A_{\mathcal{M}}$. The content invariant (**P3**) and exemplar are identical across all three. Figure 10 shows the verbatim prompt sent to the model for the full CONCISON setting. The Baseline, Plan-and-Solve, and TALE-EP prompts follow their original formulations and are not reproduced here.

B Cross-Model Predictive Entropy

Figure 11 reports the mean per-token Shannon entropy by lexical category for each of the 16 self-hosted open-source models used in the analysis of §3.1. The pooled pattern of Figure 4 holds without exception: math-symbol tokens are near-deterministic ($\bar{H} \lesssim 0.1$ nats) while discourse tokens (filler, hedging, conjunction) sit an order of magnitude higher. The inversion holds across all five open-source families and across a $> 30\times$ range in parameter count (Gemma-E2B $\approx$ 2B to Nemotron-120B), ruling out the worry that the entropy/interchangeability finding is specific to one model or scale.

Rule (principle)	L1	L2	CONCISON
Content invariant (P3)	✓	✓	✓
Drop fillers, hedges, pleasantries (P1)	✓	✓	✓
Drop articles / subject pronouns (P1)		✓	✓
Drop copulas / non-load conjunctions (P1)		✓	✓
Sentence fragments permitted (P1)		✓	✓
Causal-arrow substitution (P2)		✓	✓
Tokenizer-aware abbreviation $A_{\mathcal{M}}$ (P2)			✓
Good/bad exemplar	✓	✓	✓

Table 4: Rule composition of the three CONCISON variants. The variants nest: L1 $\subset$ L2 $\subset$ CONCISON.

```
You are a problem solver. Reason step by step, then give
the final answer. Numbers, equations, code, and technical
terms must remain exact.

Style: MAXIMUM COMPRESSION. Talk ultra-terse.
MUST do all of (apply EVERY rule when applicable):
  - DROP articles (a/an/the)
  - DROP fillers (just/really/basically/so/then/
    therefore/because/hence)
  - DROP subject pronouns and copulas when droppable
  - APPLY abbreviations: function→fn, equation→eq,
    value→val, answer→ans, maximum→max, . . .
  - Strip conjunctions
  - ARROWS for causality: X → Y forward, X ← Y back
  - ONE word when one word enough; fragments OK

Pattern: [fact] → [step] → [step] → ans = X.

Example, “Area of square, perim 28”:
  Bad: “The perimeter of a square is 4 times the side
    length, so 4s = 28, hence s = 7. Area is 49.”
  Good: “P=4s → 4s=28 → s=7 → A=s^2=49. ans=49.”

After reasoning, give the final answer in the required
format.
```

Figure 10: Verbatim CONCISON system prompt sent to every model in the broad evaluation. The content-invariant line and the good/bad exemplar are the two ingredients that make aggressive compression safe; the abbreviation list is spliced in from $A_{\mathcal{M}}$ at request time so that the model sees only tokenizer-useful pairs. Causal connectives (*so/therefore/because/hence*) appear in both the “drop fillers” and “arrows for causality” rules: the model uses the arrow form when the connective expresses an explicit $X \rightarrow Y$ relation, and drops the word otherwise.

The two commercial models in the broad evaluation (GPT-4.1, GPT-5-mini) are excluded because their public APIs do not return the top-K logprobs the analysis requires; the remaining 16 open-source models cover every family used in the main token-reduction results.

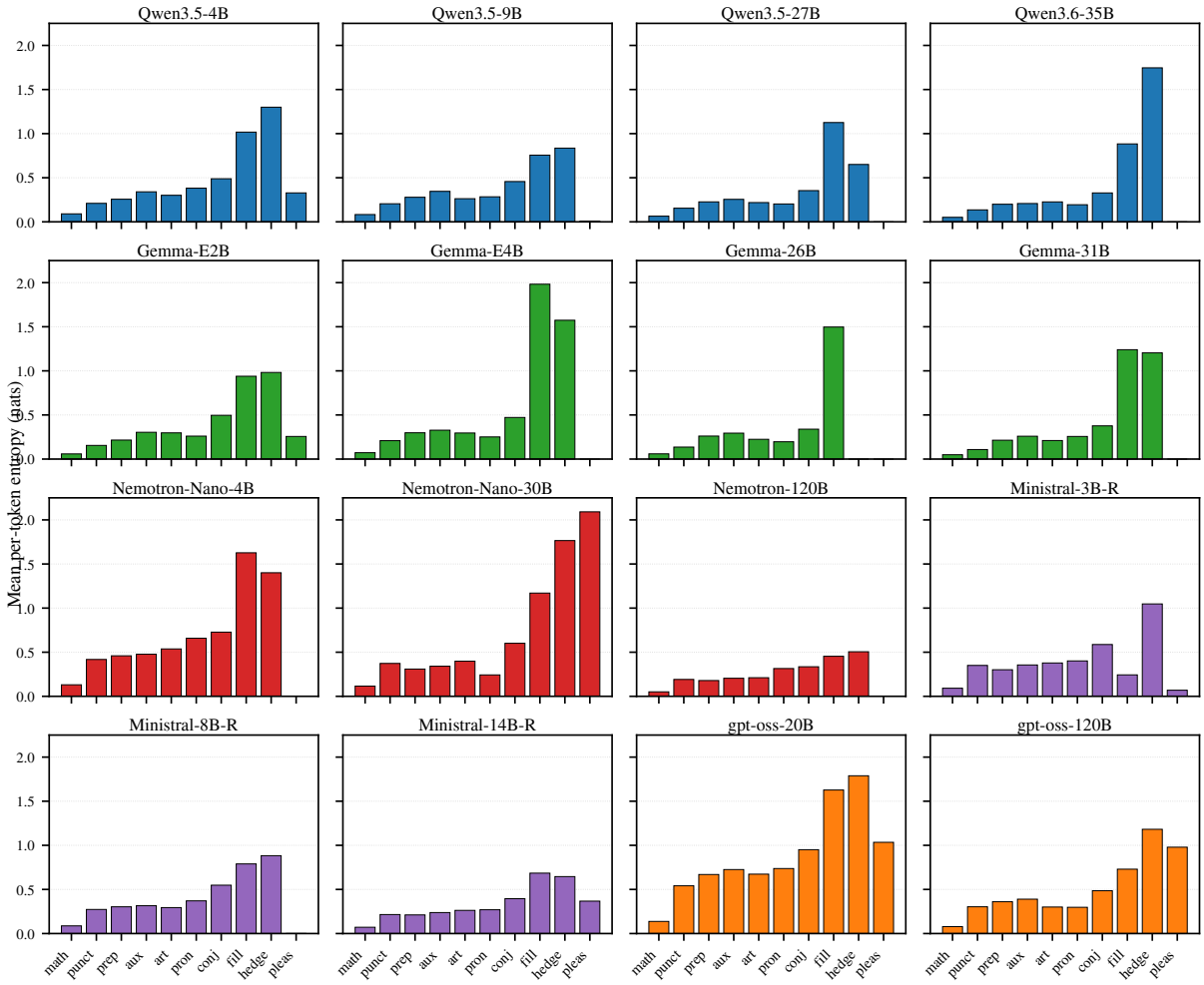


Figure 11: Mean per-token predictive entropy by lexical category, one panel per model (16 open-source models across all five open-source families). Bars are coloured by family. Low bars (math symbols, math keywords) mark near-deterministic load-bearing tokens; high bars (filler, hedging, conjunction) mark interchangeable discourse tokens. The ordering holds for every model and across a $> 30\times$ range in parameter count, establishing that the entropy/interchangeability finding of §3.1 is not a model- or scale-specific artefact.

C Per-(Lever, Model) Next-Token Agreement

Figure 6b in the main body shows the per-cell top-1 next-token agreement after each compression lever for every (lever, model) pair. We summarise the takeaway here:

- **Discourse** removal (filler, hedging, pleasantry): top-1 agreement of 82–90% across all open-source families and scales.
- **Function-word** removal (preposition, article, pronoun, copula, conjunction): top-1 agreement still high but consistently a few points lower, with greater family-level variation.
- **Content** removal (number, noun, verb): top-1 agreement collapses, confirming the unsafe regime in our framework.

The discourse-vs-content gap holds uniformly across model families and scales, so we use a single rule block for every model in §5.

D Strip-Truncate-Continue: Causal Validation

The next-token experiments of §4 are observational: they measure how an edit shifts the model's distribution. We now ask the stronger causal question: *does the answer survive the edit when the model is forced to continue from the edited trace?*

Setup. For a baseline reasoning trace y produced for query q, let $\sigma_V(y)$ delete every token whose lexical category lies in a set V, and let $\pi_K(\cdot)$ keep the first $K\%$ of characters. We feed the model the stripped, truncated prefix and have it continue *autoregressively* from that exact point, obtaining $\tilde{y} = [\pi_K(\sigma_V(y)); c]$ with $c \sim \mathcal{M}(\cdot \mid q, \pi_K(\sigma_V(y)))$,

and score $\hat{a}(\tilde{y})$. We use six nested strip sets $V_0 \subset V_1 \subset \cdots \subset V_5$, ordered by the divergence ranking of §4: V_0 strips nothing, and V_1 through V_5 cumulatively add hedges/pleasantries, fillers, conjunctions, pronouns, and articles. As a control, MATH-STRIP removes numeric and operator tokens instead. We sweep $K \in \{25, 50, 75, 100\}$ on three models – Qwen3.6-35B-A3B, Gemma-4-31B-it, Mixtral-8x22B – across all five benchmarks. We retain Mixtral here (despite excluding it from the broad evaluation in §6) because its NIM endpoint supports the `continue_final_message` flag required for autoregressive continuation.

True autoregressive continuation requires bypassing the chat template's turn boundary. We use SGLang's `/v1/completions` endpoint for Qwen3.6 and vLLM's `continue_final_message=True` for Gemma-4 and Mixtral; both make the model resume from the final token of the prefix rather than starting a new assistant turn. gpt-oss is excluded because its NIM endpoint rejects `/v1/completions` and ignores `continue_final_message`, likely due to harmony-format channel tokens.

Result. Figure 12 shows the outcome. The six strip variants V_0–V_5 stay bunched within 5–8 accuracy points of each other at every truncation level, and the gap closes toward the baseline as more of the trace is kept. The MATH-STRIP control sits 15–47 points below the discourse variants and, unlike them, degrades as more of the (corrupted) trace is supplied: with more numeric corruption to condition on, the model commits more confidently to the wrong answer. The conclusion is unambiguous: discourse and function tokens are not load-bearing for the answer; numeric and operator tokens are.

E Tokenizer-Aware Abbreviation: Useful-Subset Construction

Section 3.2 of the main body shows that the average tokens saved per abbreviation entry varies by a factor of ≈ 2 across tokenizer families. The variation is large enough that a fixed list applied blindly would *increase* token count for 29–35% of pairs on a given model. We therefore construct, for every target model $\mathcal{M}$, the useful subset

$$A_{\mathcal{M}} = \{\, (w, a) \;|\; |\text{tok}_{\mathcal{M}}(w)| > |\text{tok}_{\mathcal{M}}(a)| \,\}.$$

The cardinality $|A_{\mathcal{M}}|$ ranges from 32 (smallest-vocabulary tokenizer) to 71 (largest), and the resulting corpus savings on FineWeb-Edu range from 0.8% to 2.2% of tokens. Abbreviation alone is therefore a small additive effect, but it composes cleanly with the discourse-removal and arrow-substitution levers in the full CONCISON prompt and contributes a measurable share of the 42% output-token reduction (§7.1, Table 2).

The construction is the only per-model component of CONCISON: the rule block and exemplar are identical across all 18 evaluation models; only the spliced-in abbreviation list varies. Building $A_{\mathcal{M}}$ for a new model is one pass over the 155-pair candidate dictionary – a few seconds of CPU work.

F Arrow Substitution: Per-Model Distribution Shift

We replicate the word-type-removal framework of §4 with a substitution edit. The operator $e_{\rightarrow}$ regex-replaces, on each FineWeb prefix, occurrences of "*so/because/since*" (after a comma), "*Therefore/Hence/Thus/So/Consequently*" (sentence-initial, possibly followed by a comma), and the bare connectives "*therefore/hence/thus/consequently*" with the symbol "$\rightarrow$" (a single token under every tokenizer we test). We then measure $D_{\text{JS}}(P \| P_e)$ as before.

Table 5 reports per-model statistics across 200 FineWeb prefixes per model where at least one substitution fires. The median Jensen–Shannon divergence is 0.003–0.087 across the 15 open-source models with sufficient coverage (Nemotron-Super-120B from the broader 16-model entropy sweep is not included here); eleven of fifteen sit at or below $D_{\text{JS}} = 0.02$, comparable to or below the median for dropping a conjunction in §4 and an order of magnitude below dropping a content word. Top-1 next-token agreement stays at 65–93%. Figure 13 compares the arrow-substitution box (rightmost, dark blue) to the eleven word-type removals on the same prefixes for each model: the arrow box sits at or below the conjunction-removal level on every model.

The result aligns directly with the entropy analysis (§3.1): causal connectives sit at the high-entropy end of the model's predictive distribution, so the model already treats them as substitutable, and substituting with "$\rightarrow$" (which has substantial probability mass at exactly those positions) moves the prediction only marginally. Arrow substitution is therefore distributionally safe at the same level as deleting discourse markers, and gains $\approx$2–4 tokens per connective on most tokenizers.

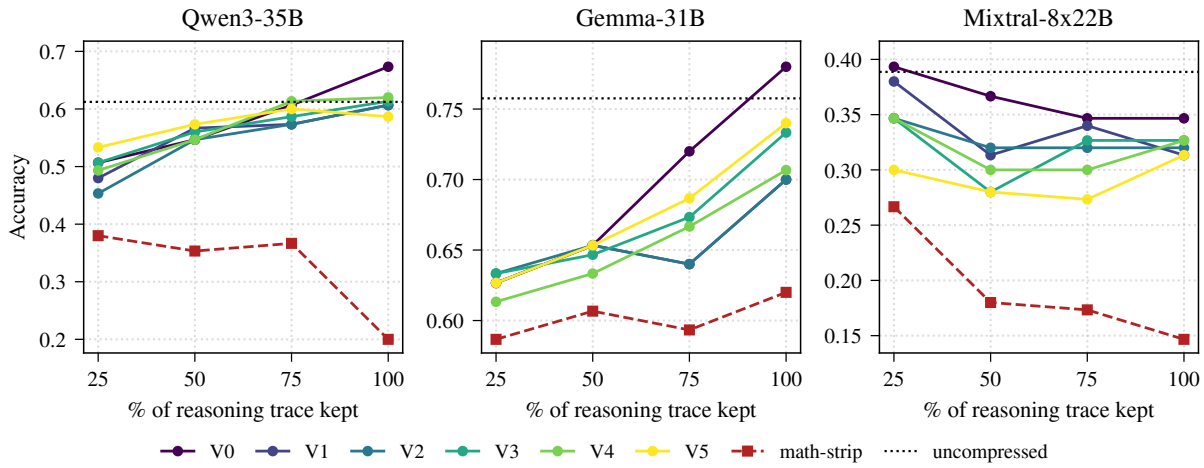


Figure 12: Strip-truncate-continue. Accuracy when the model continues autoregressively from a reasoning trace whose first $K\%$ is kept, after stripping nested categories of discourse and function tokens (V_0: none, up to V_5: hedges + fillers + conjunctions + pronouns + articles). Discourse stripping (V_0 to V_5) stays close to the uncompressed baseline (dotted); the MATH-STRIP control (red dashed) collapses.

Model	**med.** D_{JS}	**mean** D_{JS}	**top-1**
Qwen3.5-4B	0.0043	0.0153	93.3%
Qwen3.5-9B	0.0038	0.0149	88.5%
Qwen3.5-27B	0.0028	0.0124	92.3%
Qwen3.6-35B-A3B	0.0052	0.0158	92.8%
Gemma-4-E2B	0.0436	0.1250	70.2%
Gemma-4-E4B	0.0430	0.1046	70.2%
Gemma-4-26B-A4B	0.0745	0.1406	64.9%
Gemma-4-31B	0.0868	0.1433	66.3%
Nemotron-Nano-4B	0.0032	0.0113	90.9%
Nemotron-Nano-30B	0.0050	0.0182	89.4%
Ministral-3B-R	0.0031	0.0139	89.9%
Ministral-8B-R	0.0026	0.0139	89.4%
Ministral-14B-R	0.0035	0.0130	91.3%
gpt-oss-20B	0.0190	0.0418	77.9%
gpt-oss-120B	0.0173	0.0358	78.8%

Table 5: Arrow-substitution next-token divergence per model (15 open-source models, all five families used in the broad evaluation). Top-1 is next-token agreement; "Nemotron-Nano-30B" abbreviates Nemotron-Nano-30B-A3B. Word-boundary-trimmed prefixes (Appendix G).

G Prefix Word-Boundary Preprocessing

All FineWeb-prefix experiments in §4–§F measure $D_{\mathrm{JS}}\big(p_{\mathcal{M}}(\cdot \mid x) \,\big\|\, p_{\mathcal{M}}(\cdot \mid e(x))\big)$ for a prefix x and a text edit e. A naive implementation samples x as the first N characters of a document, which often lands mid-word (e.g. "calcula"). The model is then forced into a strong "complete-the-fragment" continuation that has nothing to do with the semantic edit, and the resulting spurious distribution shift inflates D_{JS} asymmetrically across word types: function-word edits change prefix length differently than discourse-word edits, so the mid-word boundary lands differently in each.

We mitigate this with a one-line preprocessing step: after the N-character cut, trim x back to the last whitespace boundary, ensuring the prefix never ends mid-word. We A/B-tested the effect on Qwen3.5-4B, 200 paired FineWeb prefixes, every word-type removal: median D_{JS} drops by 2.5–3.5$\times$ on function words and by $\approx$100$\times$ on discourse words (which were already near-zero). The qualitative ranking discourse $\ll$ function $\ll$ content is preserved in both regimes, but the word-boundary numbers are artefact-free, and we report only those in §4.

H Reproducibility and Implementation

Decoding. Every result in the main body uses greedy decoding (temperature $=$ 0, top$_p$ $=$ 1). Greedy outputs are deterministic up to GPU floating-point non-determinism, which is small enough that re-running a row of Table 1 reproduces the reported accuracy and mean token count to within rounding. The maximum generation length is 4,096 tokens for every model.

Serving infrastructure. Self-hosted open-source models are served via vLLM (for dense models) or SGLang (for MoE models that benefit from radix attention). NIM-hosted models (gpt-oss-20B / 120B, Nemotron-3-Super-120B) use the public OpenAI-compatible NIM endpoint with two API keys, round-robin dispatch, and a per-key 40-rpm sliding-window limiter to respect the published rate limits. Two commercial models (GPT-4.1, GPT-5-

mini) use the OpenAI API directly with the corresponding endpoints.

Per-model decoding flags. Three model families require non-default flags to disable their built-in long-reasoning channel, because we evaluate them at a fixed inference budget rather than their model-default budget. Qwen3 receives `chat_template_kwargs={"enable_thinking": False}`; Nemotron receives `reasoning_effort=none`; gpt-oss receives `reasoning_effort=low`. Gemma's chat template forbids a system role, so the system prompt is merged into the user message.

Resumability and sample matching. Every run writes a per-sample JSONL line with the $(\textit{model}, \textit{dataset}, \textit{setting}, \textit{sample_id})$ key. Reruns skip completed keys, so an interrupted experiment resumes deterministically. Per-(model, dataset) deltas in Table 1 are computed on the matched intersection of sample IDs across the baseline and each method, ensuring like-for-like comparisons even when a method occasionally errors out.

Known data issues. Three known sources of measurement noise affect our absolute numbers, but not the comparison between methods (because each method is affected identically):

- **gpt-oss harmony channel.** About 10–15% of gpt-oss responses place the final boxed answer in the analysis channel, where our regex extractor misses it. This depresses absolute accuracy uniformly across the Baseline, Plan-and-Solve, TALE-EP, and CONCISON settings for gpt-oss-20B/120B.
- **Nemotron stripped reasoning.** With `reasoning_effort=none`, $\sim$ 25% of Nemotron records have `final_text` equal to just "`\boxed{X}`" (the analysis channel is stripped). The boxed prediction is still extracted correctly, so accuracy is unaffected, but text-level analyses on these records (e.g. token-by-token entropy) are not informative for that subset.
- **NIM rate limits.** A small fraction of NIM requests returned 429 errors. Affected cells were re-run with an 8-attempt exponential-backoff retry and now have 100% completion.

I Per-Benchmark Results and Per-(Model, Dataset) Tokens

This appendix gives the per-benchmark detail behind the averaged main-body Table 1. Table 6 reports accuracy (*Acc.*, %) and Token Efficiency (*Eff.*, accuracy per 1K tokens per billion parameters) for every (model, method, benchmark) cell of the broad evaluation, plus the unweighted average per model. Table 7 reports the underlying mean output-token counts behind the efficiency numbers (the average column of which is what appears in the *Tokens* column of the main-body table).

J Per-Family Output Tokens and Accuracy

Figures 14–19 break down the broad evaluation by model family. Each figure follows the layout of Figure 1 but uses model size on the x-axis instead of model family: the top panel reports average output tokens (Original vs. CONCISON), the bottom panel reports average accuracy, both averaged across the five benchmarks. Token reduction is visible at every size on every family; accuracy stays close to baseline on the larger members, with the largest accuracy gaps concentrated on the smallest models in each family (Qwen3.5-4B, Gemma4-E2B, Ministral-3B-R) that struggle to follow the dense rule block; the lighter L1/L2 variants recover most of the lost accuracy there (§7.2). Annotations above each bar group give the headline percent token change and the accuracy point delta vs. that model's own baseline.

K Per-(Family, Benchmark) Pareto Grid

Figure 20 replicates the averaged Figure 7 at the per-(family, benchmark) granularity: each panel plots accuracy vs. size-weighted token cost (mean tokens $\times$ params) for a single family on a single benchmark, with trained routers (RouteLLM, AutoMix, MFRouter, SW-Ranking) as swept curves and the always-small / baseline / CONCISON references as discrete points. CONCISON sits above-left of every router curve in 17 of 20 cells; the three exceptions (Gemma-GPQA, Gemma-LCB, GPT-OSS-GSM8K) are benchmarks where the small model already approaches the large one.

Method	AIME		MATH-500		GSM8K		GPQA		LCB		Average	
	Acc.	Eff.	Acc.	Eff.	Acc.	Eff.	Acc.	Eff.	Acc.	Eff.	Acc.	Eff.
Qwen3.6-35B-A3B												
Baseline (CoT)	36.7	0.32	69.4	1.62	95.8	6.47	67.7	0.77	36.6	0.35	61.2	0.84
Plan-and-Solve	36.7$_{\uparrow 0.0}$	0.32$_{\uparrow 0.00}$	70.0$_{\uparrow 0.6}$	1.58$_{\downarrow 0.04}$	**97.0**$_{\uparrow 1.2}$	4.32$_{\downarrow 2.15}$	70.0$_{\uparrow 2.3}$	0.72$_{\downarrow 0.05}$	41.0$_{\uparrow 4.4}$	0.40$_{\uparrow 0.05}$	62.9$_{\uparrow 1.7}$	0.82$_{\downarrow 0.02}$
TALE-EP†	**42.7**$_{\uparrow 6.0}$	**0.41**$_{\uparrow 0.09}$	67.0$_{\downarrow 2.4}$	2.49$_{\uparrow 0.87}$	94.0$_{\downarrow 1.8}$	9.10$_{\uparrow 2.63}$	**75.0**$_{\uparrow 7.3}$	**1.26**$_{\uparrow 0.49}$	**43.0**$_{\uparrow 6.4}$	**0.48**$_{\uparrow 0.13}$	**64.3**$_{\uparrow 3.1}$	1.10$_{\uparrow 0.26}$
CONCISON (Ours)	40.0$_{\uparrow 3.3}$	0.41$_{\uparrow 0.09}$	**70.6**$_{\uparrow 1.2}$	**3.55**$_{\uparrow 1.93}$	93.9$_{\downarrow 1.9}$	**22.11**$_{\uparrow 15.64}$	69.2$_{\uparrow 1.5}$	1.08$_{\uparrow 0.31}$	40.0$_{\uparrow 3.4}$	0.44$_{\uparrow 0.09}$	62.7$_{\uparrow 1.5}$	**1.13**$_{\uparrow 0.29}$
Gemma4-31B-it												
Baseline (CoT)	60.0	0.91	**72.0**	3.56	96.4	10.58	**77.8**	2.82	**72.6**	1.48	75.8	2.20
Plan-and-Solve	**76.7**$_{\uparrow 16.7}$	1.01$_{\uparrow 0.10}$	71.0$_{\downarrow 1.0}$	2.54$_{\downarrow 1.02}$	**97.0**$_{\uparrow 0.6}$	6.18$_{\downarrow 4.40}$	77.0$_{\downarrow 0.8}$	1.79$_{\downarrow 1.03}$	64.0$_{\downarrow 8.6}$	1.12$_{\downarrow 0.36}$	**77.1**$_{\uparrow 1.3}$	1.75$_{\downarrow 0.45}$
TALE-EP†	52.7$_{\downarrow 7.3}$	0.86$_{\downarrow 0.05}$	67.0$_{\downarrow 5.0}$	4.81$_{\uparrow 1.25}$	94.0$_{\downarrow 2.4}$	14.66$_{\uparrow 4.08}$	69.0$_{\downarrow 8.8}$	3.36$_{\uparrow 0.54}$	64.0$_{\downarrow 8.6}$	1.23$_{\downarrow 0.25}$	69.3$_{\downarrow 6.5}$	2.24$_{\uparrow 0.04}$
CONCISON (Ours)	60.0$_{\uparrow 0.0}$	**1.04**$_{\uparrow 0.13}$	70.6$_{\downarrow 1.4}$	**7.98**$_{\uparrow 4.42}$	95.8$_{\downarrow 0.6}$	**36.77**$_{\uparrow 26.19}$	70.2$_{\downarrow 7.6}$	**3.83**$_{\uparrow 1.01}$	70.4$_{\downarrow 2.2}$	**1.66**$_{\uparrow 0.18}$	73.4$_{\downarrow 2.4}$	**2.82**$_{\uparrow 0.62}$
Nemotron-3-Super-120B												
Baseline (CoT)	30.0	0.07	65.0	0.40	94.8	2.08	54.0	0.15	35.4	0.10	55.8	0.21
Plan-and-Solve	**40.0**$_{\uparrow 10.0}$	0.10$_{\uparrow 0.03}$	**67.0**$_{\uparrow 2.0}$	0.49$_{\uparrow 0.09}$	**99.0**$_{\uparrow 4.2}$	2.08$_{\uparrow 0.00}$	55.0$_{\uparrow 1.0}$	0.16$_{\uparrow 0.01}$	**41.0**$_{\uparrow 5.6}$	0.12$_{\uparrow 0.02}$	**60.4**$_{\uparrow 4.6}$	0.23$_{\uparrow 0.02}$
TALE-EP†	39.3$_{\uparrow 9.3}$	0.11$_{\uparrow 0.04}$	65.0$_{\uparrow 0.0}$	0.74$_{\uparrow 0.34}$	94.0$_{\downarrow 0.8}$	3.49$_{\uparrow 1.41}$	57.0$_{\uparrow 3.0}$	0.21$_{\uparrow 0.06}$	41.0$_{\uparrow 5.6}$	**0.13**$_{\uparrow 0.03}$	59.3$_{\uparrow 3.5}$	0.28$_{\uparrow 0.07}$
CONCISON (Ours)	36.7$_{\uparrow 6.7}$	**0.11**$_{\uparrow 0.04}$	65.0$_{\uparrow 0.0}$	**1.11**$_{\uparrow 0.71}$	94.9$_{\uparrow 0.1}$	**9.71**$_{\uparrow 7.63}$	**61.1**$_{\uparrow 7.1}$	**0.28**$_{\uparrow 0.13}$	40.6$_{\uparrow 5.2}$	0.12$_{\uparrow 0.02}$	59.7$_{\uparrow 3.9}$	**0.31**$_{\uparrow 0.10}$
GPT-OSS-120B												
Baseline (CoT)	**53.3**	0.14	70.2	0.63	94.8	2.43	**69.2**	0.34	56.6	0.21	**68.8**	0.34
Plan-and-Solve	50.0$_{\downarrow 3.3}$	0.21$_{\uparrow 0.07}$	68.0$_{\downarrow 2.2}$	0.77$_{\uparrow 0.14}$	95.0$_{\uparrow 0.2}$	1.87$_{\downarrow 0.56}$	68.0$_{\downarrow 1.2}$	0.67$_{\uparrow 0.33}$	57.0$_{\uparrow 0.4}$	0.43$_{\uparrow 0.22}$	67.6$_{\downarrow 1.2}$	0.55$_{\uparrow 0.21}$
TALE-EP†	39.3$_{\downarrow 14.0}$	0.16$_{\uparrow 0.02}$	67.0$_{\downarrow 3.2}$	0.90$_{\uparrow 0.27}$	91.5$_{\downarrow 3.3}$	2.83$_{\uparrow 0.40}$	61.0$_{\downarrow 8.2}$	0.65$_{\uparrow 0.31}$	55.0$_{\downarrow 1.6}$	0.39$_{\uparrow 0.18}$	62.8$_{\downarrow 6.0}$	0.55$_{\uparrow 0.21}$
CONCISON (Ours)	43.3$_{\downarrow 10.0}$	**0.27**$_{\uparrow 0.13}$	**72.0**$_{\uparrow 1.8}$	**1.90**$_{\uparrow 1.27}$	**96.6**$_{\uparrow 1.8}$	**6.33**$_{\uparrow 3.90}$	58.0$_{\downarrow 11.2}$	**1.02**$_{\uparrow 0.68}$	**59.0**$_{\uparrow 2.4}$	**0.50**$_{\uparrow 0.29}$	65.8$_{\downarrow 3.0}$	**0.85**$_{\uparrow 0.51}$

Table 6: Per-(model, method, benchmark) accuracy (*Acc.*, %) and Token Efficiency (*Eff.*); plus the unweighted Average column over the five benchmarks. Subscripts on non-baseline rows are absolute deltas vs. that model's own baseline (pp for Acc., Acc/(KTok·B) units for Eff.). Best value per metric within each model block is in **bold**. Blue marks an improvement; red marks a regression. The compact average view appears as Table 1. †TALE-EP additionally spends a separate budget-estimation call.

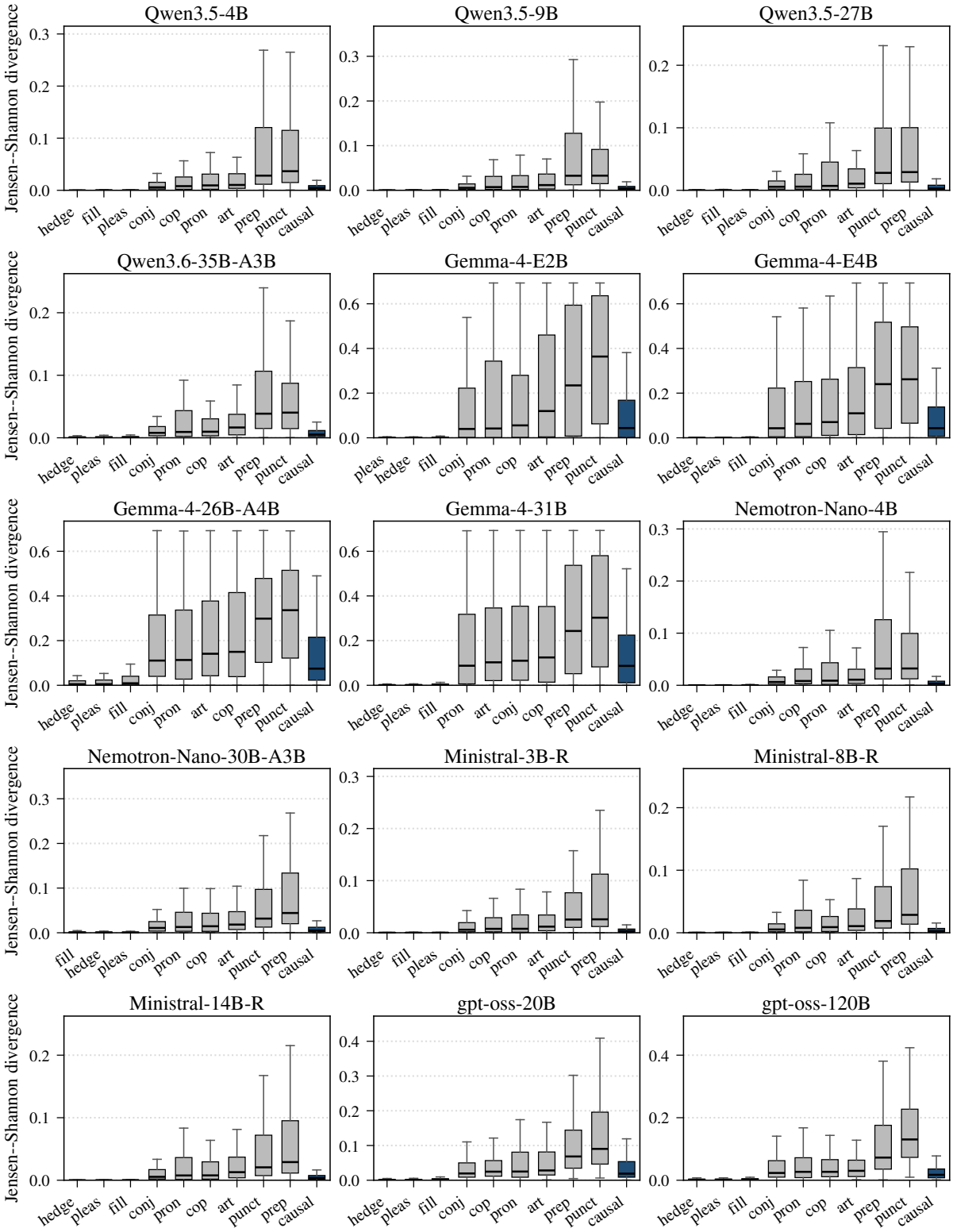


Figure 13: Arrow substitution (→) vs. word-type removal on the same FineWeb prefixes (next-token JS divergence; boxplots without outliers). The nine word-type removal categories are ordered left-to-right by their median divergence (cheapest first); the arrow-substitution box (dark blue) is placed at the right. Across fifteen models spanning all five open-source families, the arrow-substitution median is 0.003–0.087 nats and in every panel sits at or below the conjunction-removal level. The content-word removal control sits an order of magnitude higher (median $D_{\mathrm{JS}} \approx 0.35$–$0.69$), confirming the gap.

Method	AIME	MATH-500	GSM8K	GPQA	LCB	Average
Qwen3.6-35B-A3B						
Baseline (CoT)	3239	1222	423	2525	3003	2082
Plan-and-Solve	3316$_{\uparrow 2}$	1263$_{\uparrow 3}$	642$_{\uparrow 52}$	2772$_{\uparrow 10}$	2954$_{\downarrow 2}$	2189$_{\uparrow 5}$
TALE-EP†	2993$_{\downarrow 8}$	768$_{\downarrow 37}$	295$_{\downarrow 30}$	**1701**$_{\downarrow 33}$	**2576**$_{\downarrow 14}$	1666$_{\downarrow 20}$
CONCISON (Ours)	**2816**$_{\downarrow 13}$	**569**$_{\downarrow 53}$	**121**$_{\downarrow 71}$	1834$_{\downarrow 27}$	2612$_{\downarrow 13}$	**1590**$_{\downarrow 24}$
Gemma-4-31B-it						
Baseline (CoT)	2128	652	294	891	1584	1110
Plan-and-Solve	2459$_{\uparrow 16}$	903$_{\uparrow 39}$	507$_{\uparrow 72}$	1389$_{\uparrow 56}$	1837$_{\uparrow 16}$	1419$_{\uparrow 28}$
TALE-EP†	1995$_{\downarrow 6}$	449$_{\downarrow 31}$	207$_{\downarrow 30}$	663$_{\downarrow 26}$	1679$_{\uparrow 6}$	999$_{\downarrow 10}$
CONCISON (Ours)	**1870**$_{\downarrow 12}$	**285**$_{\downarrow 56}$	**84**$_{\downarrow 71}$	**591**$_{\downarrow 34}$	**1366**$_{\downarrow 14}$	**839**$_{\downarrow 24}$
Nemotron-3-Super-120B						
Baseline (CoT)	3523	1340	380	3075	3055	2275
Plan-and-Solve	3460$_{\downarrow 2}$	1152$_{\downarrow 14}$	396$_{\uparrow 4}$	2880$_{\downarrow 6}$	2851$_{\downarrow 7}$	2148$_{\downarrow 6}$
TALE-EP†	2975$_{\downarrow 16}$	726$_{\downarrow 46}$	224$_{\downarrow 41}$	2241$_{\downarrow 27}$	**2640**$_{\downarrow 14}$	1761$_{\downarrow 23}$
CONCISON (Ours)	**2889**$_{\downarrow 18}$	**488**$_{\downarrow 64}$	**81**$_{\downarrow 79}$	**1813**$_{\downarrow 41}$	2717$_{\downarrow 11}$	**1598**$_{\downarrow 30}$
GPT-OSS-120B						
Baseline (CoT)	3211	934	325	1687	2218	1675
Plan-and-Solve	1968$_{\downarrow 39}$	738$_{\downarrow 21}$	424$_{\uparrow 30}$	849$_{\downarrow 50}$	1096$_{\downarrow 51}$	1015$_{\downarrow 39}$
TALE-EP†	1968$_{\downarrow 39}$	623$_{\downarrow 33}$	270$_{\downarrow 17}$	787$_{\downarrow 53}$	1176$_{\downarrow 47}$	965$_{\downarrow 42}$
CONCISON (Ours)	**1325**$_{\downarrow 59}$	**317**$_{\downarrow 66}$	**127**$_{\downarrow 61}$	**473**$_{\downarrow 72}$	**976**$_{\downarrow 56}$	**644**$_{\downarrow 62}$

Table 7: Per-(model, dataset) output-token counts (mean completion tokens). The *Baseline (CoT)* row gives absolute values; non-baseline rows give absolute values with a coloured subscript for the signed percent change vs. that model's own baseline. Blue = reduction, red = increase. The smallest token count per (model, dataset) is in **bold**. This table is the per-dataset complement of the efficiency-only Table 1. †TALE-EP additionally spends a separate budget-estimation call not counted here.

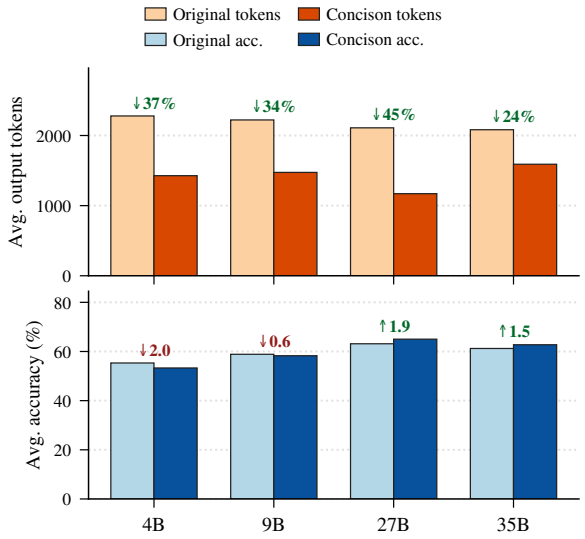


Figure 14: Qwen3.5 family (4B / 9B / 27B / 35B-A3B). Output tokens (top) and accuracy (bottom), averaged across the five benchmarks.

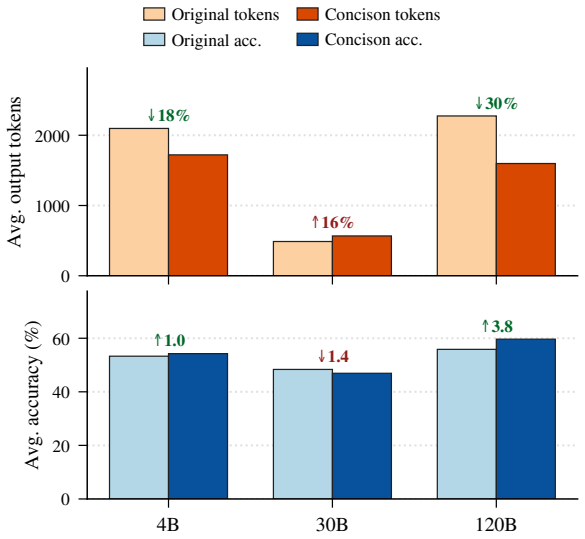


Figure 16: Nemotron family (Nano-4B / Nano-30B / Super-120B). Tokens (top) and accuracy (bottom), averaged across the five benchmarks.

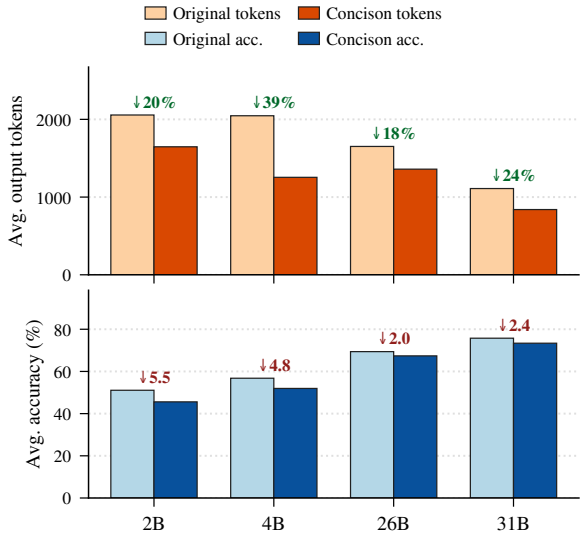


Figure 15: Gemma4 family (E2B / E4B / 26B-A4B / 31B-it). Tokens (top) and accuracy (bottom), averaged across the five benchmarks.

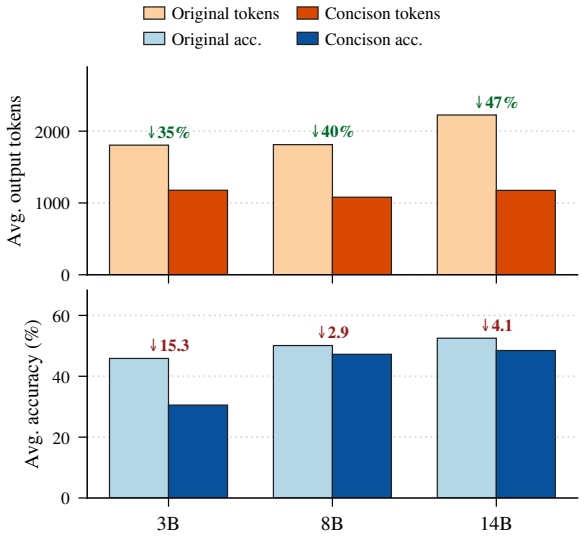


Figure 17: Ministral family (3B-R / 8B-R / 14B-R). Tokens (top) and accuracy (bottom), averaged across the five benchmarks.

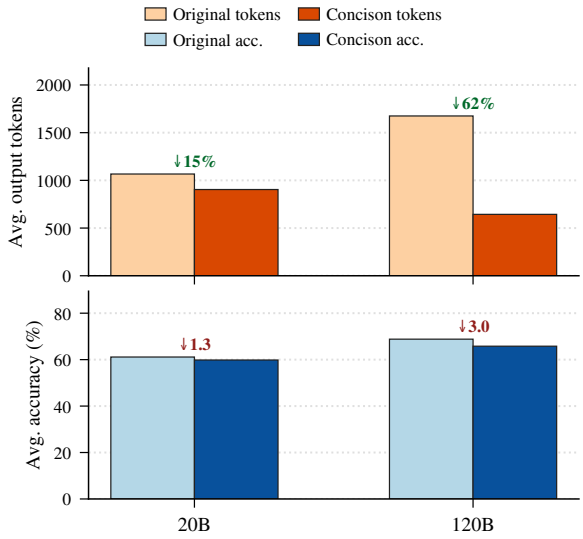


Figure 18: GPT-OSS family (20B / 120B). Tokens (top) and accuracy (bottom), averaged across the five benchmarks.

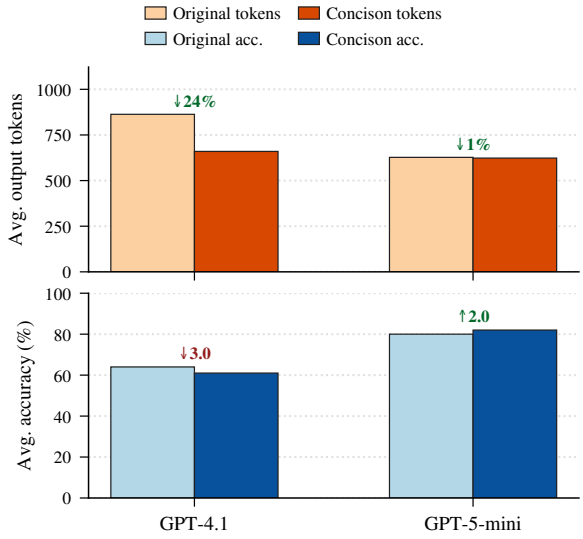


Figure 19: Commercial models (GPT-4.1, GPT-5-mini). Tokens (top) and accuracy (bottom), averaged across the five benchmarks.

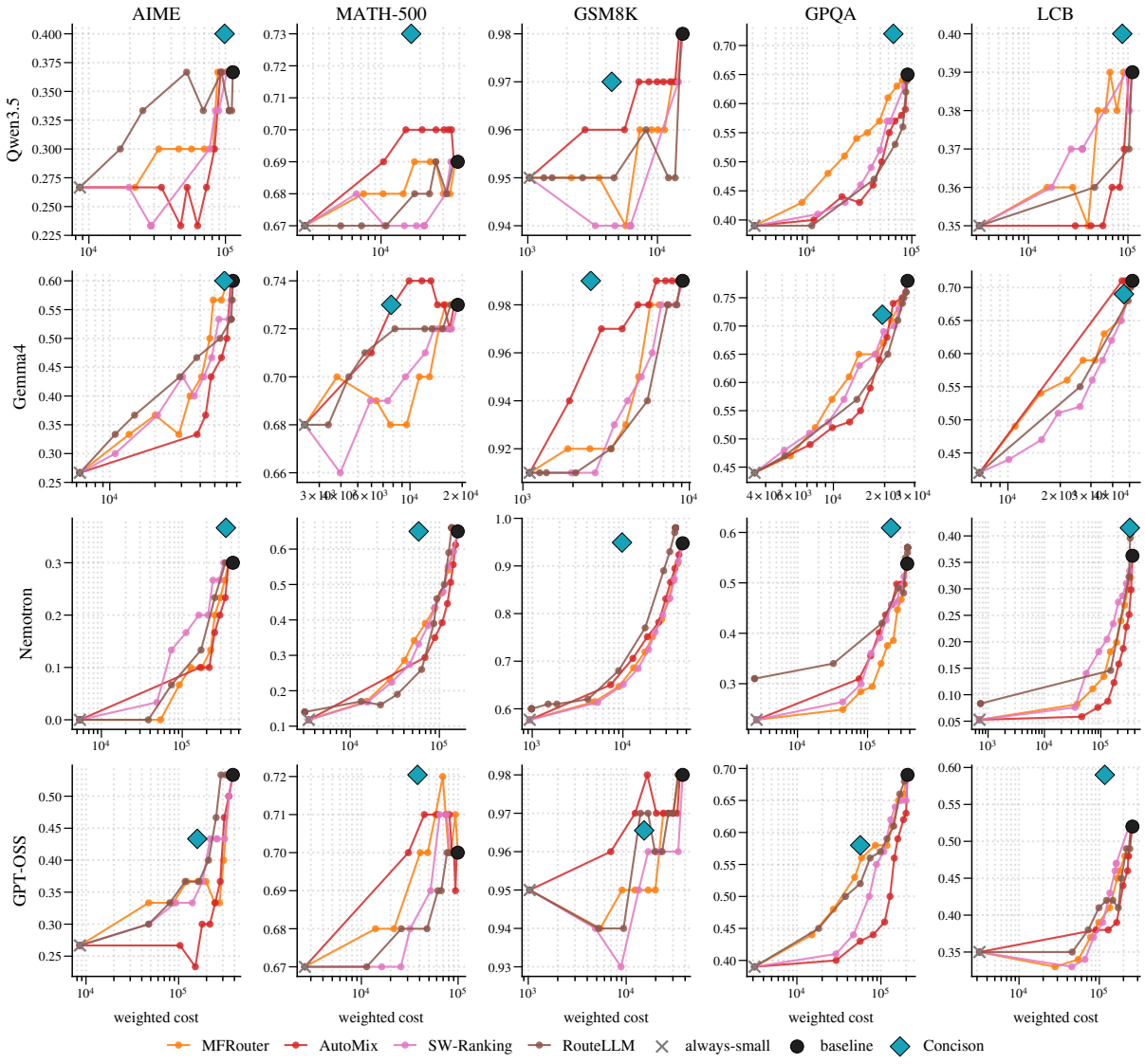


Figure 20: Per-(family, benchmark) accuracy vs. size-weighted token cost. Rows: families; columns: benchmarks. Same legend as Figure 7.