COMPUTE-OPTIMAL AGENTIC RAG

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-context language models can read whole documents, but paying for every token on every question is costly. An agent can instead bring document information into its context through a question-independent summary, written once and reused, or through retrieval at question time. How should inference compute be split between summary length, retrieval and model size? We vary five levers—summary length, number of retrieval calls, passages per call, passage length and model size (Qwen3.5, 0.8B to 27B parameters)—on LongBench-v2, over about 860k question–configuration pairs, and count each question's theoretical FLOPs exactly. Summaries and retrieval supply one resource, document information, and substitute for each other: the longer the summary, the less retrieval pays. Model size is a separate axis, with a threshold near 4B parameters below which the levers help little. A 15-parameter law built on this structure predicts lever combinations it was not fit on close to the limit set by sampling noise. Its compute-optimal allocation is simple: past the threshold, and for documents queried often enough to amortize the summary, spend about half of each question's compute on the summary and keep retrieval to a few thousand tokens; for a document queried once, retrieval and a larger model are best. An agent left to decide how often to search stays about 7 accuracy points below this allocation.

1 INTRODUCTION

Long-context language models can read hundreds of thousands of tokens at once, but reading a whole document for every question pays for every token, however little of the document the question needs. An agent can instead keep the document in an external memory and bring in only part of it, in one of two ways. *Retrieval* fetches passages chosen for the question, one tool call at a time. A *summary* compresses the whole document once, independently of any question, and is reused for every question asked about it, as in systems that build global representations of long inputs (Sarthi et al., 2024; Lee et al., 2024; Chen et al., 2023; Qian et al., 2024). Together with the choice of model, these compete for the same inference budget: a longer summary, more retrieval calls, more or longer passages per call, or a larger model. Where should the next unit of compute go?

Two things make the answer non-obvious. First, a summary could either *complement* retrieval or *substitute* for it. It could make retrieval more productive, by telling the agent what the document contains and which words to search for; or it could make retrieval less necessary, by already supplying what retrieval would have found. The two intuitions imply opposite allocations. Second, the levers' costs scale differently: a summary that keeps a fixed fraction of the document costs more for longer documents, whereas a retrieved passage costs the same whatever the document's length.

We study this question at scale. A retrieval agent answers LongBench-v2 questions (Bai et al., 2024) with Qwen3.5 models from 0.8B to 27B parameters (Qwen Team, 2026), and we vary five levers—summary length, number of retrieval calls, passages per call, passage length and model size—over about 860k question–configuration pairs, counting each question's compute in FLOPs, generated tokens included.

Our central finding is that the design space has two axes, not five levers: *document information*, which summaries and retrieval both supply and which substitute for each other, and *model capability*, which sets how much the model gets out of that information and which the other levers leave unchanged. Figure 1 shows what this means for allocation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

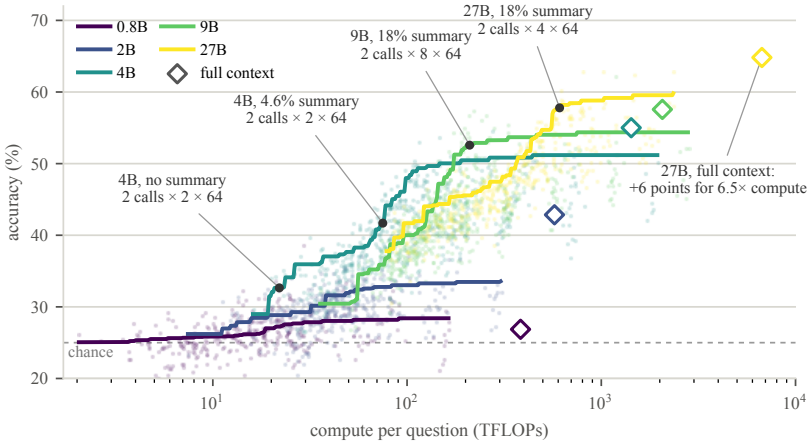

Figure 1: **Compute-optimal allocation.** Each line is the law's best predicted accuracy for one model size at each compute budget, over the configurations we ran; points are all configurations with at least 100 questions, colored by model size; diamonds are full context (the whole document, no retrieval). Labels give the best configuration at four budgets (calls × passages per call × passage length in tokens). The 0.8B model stays within about 3 points of chance; the 4B, 9B and 27B models take over from the next smaller one for good at about 16, 173 and 469 TFLOPs; along the 4B curve, accuracy is bought by a longer summary. The summary's one-time cost is excluded (Section 5 charges it).

Summaries and retrieval substitute; model size is a threshold. Accuracy grows with the logarithm of each retrieval lever, and with that of summary length once a summary passes about 0.2 to 0.8% of the document. The longer the summary, the less retrieval pays: on net, summaries replace retrieval rather than make it more productive. Within retrieval, more passages beat longer ones. Below about 4B parameters no amount of document information helps much; from 4B up the levers act the same way at every size.

A simple law predicts how the levers combine. Accuracy depends on what the model can do and on how much document information it receives, to which summaries and retrieval contribute as near-interchangeable sources: a summary through the fraction of the document it keeps, retrieval through the *distinct* passages the agent receives (agents re-retrieve: with eight calls of eight passages, nearly half of what they fetch is a repeat). With 15 parameters, the law predicts lever combinations it was not fit on close to the limit set by sampling noise.

The compute-optimal allocation follows a compact rule. From about 60 TFLOPs per question, a longer summary is the lever that pays most per doubling of compute; once it enters, it should take about half of each question's compute, while retrieval stays at a few thousand tokens. Two conditions shape the rule. A summary costs more for a longer document, so short documents should be summarized more. And a summary is written once, so it pays for documents queried tens of times or more; for a document queried once, retrieval and a larger model are best. Reading the whole document is the most accurate option we measured with at least 100 questions, but only at budgets where little else is left to buy.

Agents do not find this allocation on their own. Left to decide how often to search, a 9B agent retrieving one passage per call spends compute about as well as one told how often, but every instruction we tried leaves it about 7 points below the compute-optimal allocation: the choices that matter, the model size and the summary, are made before the agent runs.

2 Setup: Levers, Conditions and Compute

Task and agent. We use LongBench-v2 (Bai et al., 2024), a four-way multiple-choice benchmark (chance is 25%) of questions about long documents, keeping the 469 questions whose document fits our summarizer's 1M-token budget; they cover 434 documents (a few are shared by up to seven questions), with a median length of 99k tokens. The agent is a tool-calling language model in

Table 1: **Experimental grids.** Every grid includes no summary and every summary length unless stated; every grid covers all five model sizes unless stated. Units are question–configuration pairs (860,762 in total). The retrieval grid runs 27B up to $c = 2048$. The calls $\times$ retrieval grid uses $(k, c) = (8, 512)$ at 4B, 9B and 27B, and $(1, 2048)$ and $(8, 2048)$ at 4B and 9B.

grid	varies	fixed	units
main	summary length $\times$ $f \in \{0, 1, 2, 4, 8\}$	$k = 1$, $c = 512$	127,860
retrieval	$k \in \{1, 2, 4, 8\} \times c \in \{64, \ldots, 4096\}$	$f = 2$	695,490
calls $\times$ retrieval	$f \in \{1, 4, 8\} \times (k, c)$	none + 3 summary lengths	35,067
full context	model size		2,345

reasoning mode, so it writes a reasoning trace before each call and before its answer, with one tool: a search over the document by hybrid retrieval that fuses dense (BGE-large-en-v1.5 (Xiao et al., 2023)) and BM25 (Robertson & Zaragoza, 2009) rankings by reciprocal rank fusion (Cormack et al., 2009), over passages from a recursive splitter with 10% overlap. To make the number of calls a lever we control, the agent must make exactly f calls before it answers ($f = 0$: no retrieval); it still writes every query itself. We compare three conditions: *no summary*, *summary* (the summary is placed in the prompt ahead of the question) and *full context* (the whole document in the prompt, no retrieval, one turn), which serves as a reference.

Levers. We vary five levers (Table 1). *Summary length* is the summary's length as a fraction of its document's (the fraction retained). The summarizer writes one summary per document and target length, from a 17-value octave grid of target fractions (0.0011% to 73.9% of the document); targets outside 10 to 10,000 tokens are dropped for that document rather than clipped, and every analysis uses the realized length, not the target. *Retrieval* has three levers: the number of calls f, the passages per call k and the passage length c in tokens; we call $k \times c$ the per-call retrieval. We call summary length and the three retrieval levers the *supply levers*, since each puts document content into the context. *Model size* ranges over Qwen3.5 0.8B, 2B, 4B, 9B and 27B (Qwen Team, 2026), as a pipeline: the same model writes the summary and answers. A *configuration* is one setting of every lever and the condition; each question is run once per grid cell and target summary length, sampling at temperature 1.0 (Section C gives the sampling and serving details).

Compute. We count theoretical inference FLOPs per question under ideal key–value caching, generated tokens included, for Qwen3.5's hybrid attention and Gated DeltaNet architecture (Yang et al., 2024; Qwen Team, 2026) (Section A). We exclude the one-time cost of writing the summary, which falls below 10% of each question's own compute after a median of 129 to 234 questions per document; Section 5 charges it.

Estimation. Not every configuration covers every question (a target summary length is skipped for a document if it falls outside 10 to 10,000 tokens), so raw accuracies would partly reflect which questions a configuration happens to cover. We therefore model each question's outcome as a configuration effect plus a question effect, and report every configuration's accuracy and compute as estimated over all 469 questions (*adjusted* values). Tests use per-question logistic regressions with standard errors clustered by question (clustering by document changes no conclusion; Section D). An adjusted accuracy averages at most 469 four-way questions, so it carries a median of about 2.4 points of sampling noise, the floor for any model's prediction error.

3 What the levers do

We first describe each lever on its own axis, then how the levers interact (Figure 2). Each observation fixes one part of the law in Section 4.

Model size sets a capability threshold. The 0.8B model stays within about 3 points of chance whatever it is given, and the 2B model gains about 5 points from any document information and then plateaus. From 4B up, summary length and calls act the same way at every size: their curves are parallel across sizes. What grows with size is how much document information is worth: a long

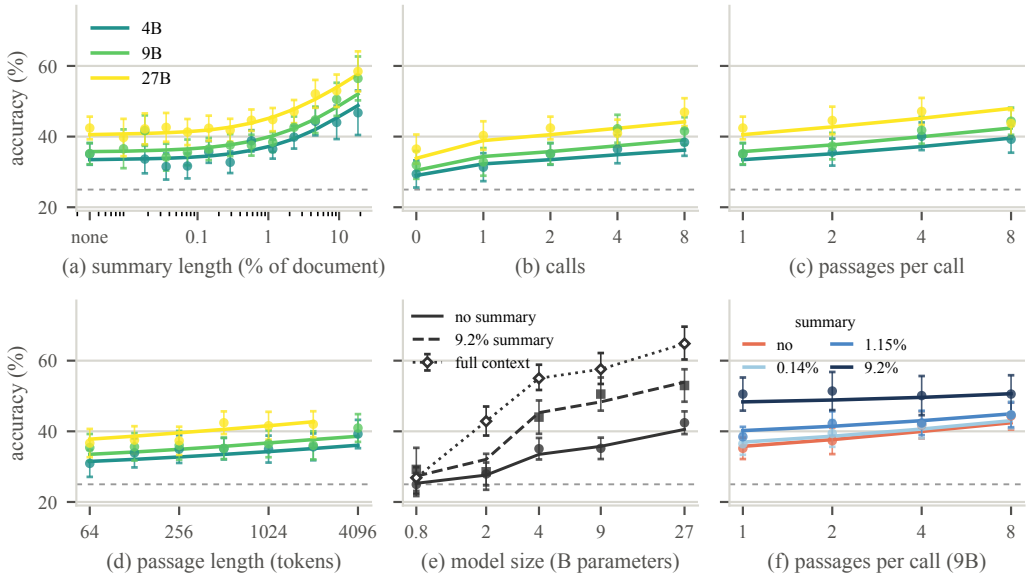

Figure 2: **What the levers do.** Points: one configuration each, with its 95% interval; lines: the law of Section 4 for the same configurations. (a) Summary length at two calls of one 512-token passage. (b) Number of calls, (c) passages per call and (d) passage length, without a summary, the other retrieval levers at one 512-token passage per call and two calls. (e) Model size at two calls of one 512-token passage, without a summary, with a 9.2% summary, and with the whole document (full context, not part of the law). (f) Passages per call at 9B (two calls of 512-token passages) without a summary and with three summary lengths: the longer the summary, the flatter the line (every size: Figure 8). Dashed line: chance.

summary's advantage over none is +3 points at 0.8B and +15 to +17 points from 4B up. The law captures this with a per-size capability and a guessing floor of 25%.

Every supply lever has diminishing returns. Accuracy rises linearly in the logarithm of the number of calls, of the passages per call and of their length, over the range we ran for each (up to eight calls, eight passages per call and 4,096-token passages). Summary length behaves differently: accuracy is flat up to a threshold of 0.2 to 0.8% of the document, then rises by 3 to 4.4 points per e-fold of summary length; the data identify "flat, then rising", not a particular functional form. Each lever therefore enters the law through the logarithm of the information it supplies.

Summaries and retrieval substitute. The longer the summary, the less retrieval pays (Figure 2f): at 9B, a doubling of the passages per call is worth +2.8 points without a summary and +0.8 with a long one, and the same holds for calls and passage length, at 27B as strongly and at 4B more weakly. Seen the other way, at 9B a long summary's advantage shrinks as retrieval grows, from +14 points to +3, but never becomes negative. On net, then, a summary replaces retrieval rather than making it more productive; this model-free test does not rule out that a summary also improves some queries, only that any such gain is smaller than what it replaces. The law combines the two kinds of information in an aggregate whose degree of substitution is estimated.

Retrieval breadth beats depth. For the same number of retrieved tokens per call, more, shorter passages beat fewer, longer ones: the effect is positive at every size and significant at most. More calls behave like more passages per call, while longer passages pay less, which suggests that calls and passages add distinct pieces of evidence and longer passages add context around the same evidence. The law gives distinct passages and passage length separate exponents.

Summaries count as a fraction, passages as passages. The gain from a longer summary does not depend on the document's length, only on the fraction it keeps, whereas each retrieved passage is

chosen for the question, so what retrieval yields depends on how many passages there are. The law measures the two accordingly. Sections D and E give the tests behind each statement.

4 A LAW FOR HOW THE LEVERS COMBINE

The form. The observations of Section 3 suggest a law with two parts: what the model can do, and how much document information it receives. We model accuracy as

$$\text{acc} = 0.25 + 0.75\,\sigma\big(\theta_N + \beta_N \log(\epsilon + I)\big), \qquad I = \big(w\,S^{\rho} + (1-w)\,R^{\rho}\big)^{1/\rho}, \tag{1}$$

where σ is the logistic function, 0.25 is the guessing floor, and θ_N and β_N are a model size's capability and how strongly document information moves its accuracy. Document information I is a constant-elasticity-of-substitution (CES) aggregate (Arrow et al., 1961) of summary information S, the fraction of the document the summary retains, and retrieval information

$$R = D^{\gamma_d}\left(\frac{c}{512}\right)^{\gamma_c} \Big/ 1000, \tag{2}$$

where D is the number of distinct passages the agent received and c their length in tokens (the constants fix the units; other values would be absorbed by w). The substitution parameter ρ interpolates between perfect substitutes ($\rho = 1$) and a multiplicative combination ($\rho \to 0$). We restrict it to $(0, 1]$ ($[0.05, 1]$ in the fit), since at $\rho \le 0$ either source would be worthless without the other, while retrieval alone and a summary alone each raise accuracy; the evidence that the levers substitute is therefore the model-free test of Section 3, not the fitted ρ. The floor ϵ is the information in the question and its answer choices alone; it keeps accuracy flat in $\log I$ until I exceeds it. The law has 15 parameters: θ_N and β_N for each of the five sizes, and w, ρ, γ_d, γ_c and ϵ. We fit it by weighted least squares to the adjusted accuracies of 2,456 configurations, weighting each by its number of questions; full context, which has no retrieval and a single turn, is not part of the fit.

Fitted values. The fit gives $\rho = 0.74$: summary and retrieval information are strong substitutes, as Section 3 found without the law. It gives $\gamma_d = 0.84$ and $\gamma_c = 0.44$, so a doubling of the distinct passages is worth about twice a doubling of their length, the breadth-over-depth result recovered from all grids at once. The 0.8B model's capability places it at the guessing floor, and the 2B model's close to it. Counting summary information as a fraction of the document and retrieval information in passages predicts held-out configurations better than counting both as fractions or both in tokens (Section F). The law also has no parameter for the summary-length threshold of Section 3, yet implies one: a summary starts to matter once its information is comparable to retrieval's, at $S^{*} = R\,((1-w)/w)^{1/\rho}$, so the threshold should move to longer summaries as retrieval grows. On the 9B retrieval grid it does, though more slowly than implied at the largest per-call retrieval (Section F).

Predicting unseen lever combinations. A law is useful to a practitioner if it predicts configurations nobody has run. We test this as a structured extrapolation: we fit the law on the main grid (summary length $\times$ calls) at every size plus the retrieval grid at 9B alone (770 configurations), and predict, without refitting, the retrieval grids at the other four sizes and the calls $\times$ retrieval cells, combinations of levers and sizes the fit never saw (the 1,272 configurations with a recorded prediction). This test used the form without re-retrieval ($D = f\,k$), and its predictions were recorded before those runs finished. Figure 3a shows the result (Table 10 gives the details): an RMSE of 2.33 accuracy points, matching the 2.29 points that the sampling noise in the observed accuracies alone would produce, with 97% of configurations within the predictions' uncertainty band widened by their own noise. The retrieval grids are predicted at the noise floor (2.28 against 2.28); the calls $\times$ retrieval cells are not (3.05 against 2.44), and are over-predicted by 1.3 points on average.

Re-retrieval. The test's misses were concentrated where both the calls and the passages per call were many (eight calls of eight 512-token passages: +1.3 to +4.2 points; of eight 2,048-token passages: +5.4 to +7.0). The cause is measurable: agents retrieve the same passage more than once, increasingly so with more calls and more passages per call; with eight calls of eight 512-token passages only about 34 of the 64 retrieved passages are distinct, and smaller models repeat themselves more. The law therefore counts distinct passages D; to predict a configuration before running it, a

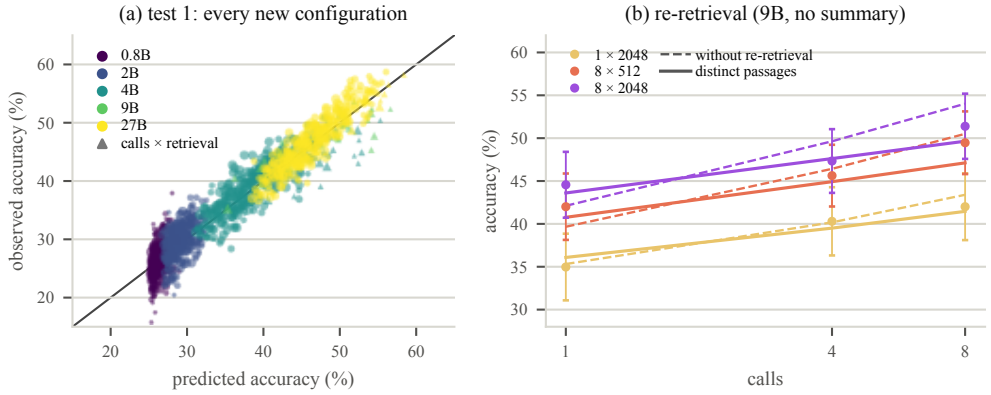

Figure 3: **Predicting unseen lever combinations.** (a) Test 1: the law's recorded predictions against the observed accuracy of every configuration run afterwards (circles: retrieval grids; triangles: calls × retrieval cells; size ∝ questions; calibration in Figure 10). (b) Re-retrieval at 9B without a summary, for three per-call retrieval settings (passages × tokens; 95% intervals). Dashed: test 1's recorded predictions ($D = f\,k$). Solid: the law with distinct passages, fit on the same 770 configurations, with distinct passages predicted from the levers (4B and 27B in Figure 11).

small model predicts D from the levers, to within 2.8% (median; 3.1% on configurations it was not fit on; Section F). We chose this retrieval term among four candidates by a rule fixed before comparing them, recorded all four candidates' predictions for configurations still running, and scored them once those runs were verified: on 152 configurations, the chosen law has an RMSE of 2.13 points against a noise of 2.37, where the form without re-retrieval scores 2.42; on the eight-call cells its bias is +0.54 points, against +2.80 (Figure 3b).

How far to trust it. Refit on every configuration, the law predicts held-out blocks at or below their noise on random folds (2.35 against 2.47), when extrapolating to eight calls (2.31 against 2.64), in the region where a long summary meets a large per-call retrieval and on the calls × retrieval cells (2.41 against 2.92). A flexible 22-parameter surface with per-size shapes and interactions predicts about as well (Section F); we prefer the law because it does so with 15 parameters, each of which has a meaning. The law is 0.25 to 0.3 points above noise when extrapolating to eight passages per call or to 4,096-token passages. The form without re-retrieval fails exactly where re-retrieval is large (3.06 at eight calls, 3.12 on the calls × retrieval cells).

5 Compute-optimal allocation

Throughout this section, the summary's one-time cost is excluded, as if it were spread over many questions; we charge it at the end. We distinguish the *empirical frontier*, the best observed configuration at each budget, from the law's *compute-optimal path*, the configuration the law predicts is best at each budget among those we ran.

The compute-optimal path. Figure 1 shows the path. Below about 16 TFLOPs per question only the two smallest models are affordable, and little helps them. From there the 4B model takes over, first with retrieval alone, then with a growing summary (from about 0.02% to 18% of the document, with two calls of small passages); the 9B model takes over at about 173 TFLOPs and the 27B at about 469, both with long summaries and small retrieval. Reading the whole document costs more than 1,000 TFLOPs even at 4B; at 27B it buys about 6 points (0 to 10 within the law's 95% band) for 6.5 times the compute of the law's best configuration under 1,200 TFLOPs, about 2 points per doubling of compute: less than a longer summary pays, similar to the other levers. The empirical frontier follows the same path but sits about 3 points above the law. The gap is selection, not misfit: the frontier picks the best of about 2,000 configurations, each with 2.4 points of noise, and when we choose it on half of the questions and measure it on the other half, it sits -0.5 ± 1.1 points from the law (Section G, which also splits the frontier's compute by component).

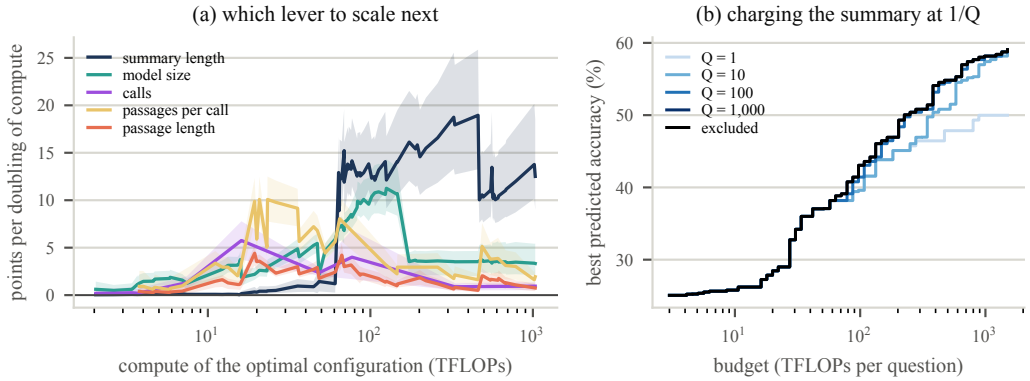

Figure 4: **Which lever to scale next, and charging the summary.** (a) Along the law's compute-optimal path (Figure 1), the accuracy gained per doubling of compute by scaling one lever from the optimal configuration (95% bands from question-bootstrap refits, conditional on the law's form); jumps mark where the optimal configuration changes; summary cost excluded. (b) The best predicted accuracy when the summary's one-time cost is charged at $1/Q$ per question, for Q questions per document (typical document length; the optimal summary length is in Figure 13).

Which lever to scale next. At each configuration on the compute-optimal path, we ask which single lever pays most for a doubling of compute (Figure 4a). Below about 16 TFLOPs, passages per call, model size and calls take turns at the top, for at most about 6 points. Around 20 to 50 TFLOPs, on the 4B model, more passages per call pay most (about +10 points per doubling at 30 TFLOPs). From about 60 TFLOPs a longer summary becomes the best lever and stays so: +12.5 points per doubling at 80 TFLOPs, where a larger model is a close second (+9.2, within the uncertainty), +14.6 at 200 TFLOPs, and +10.2 at 600 TFLOPs. Longer passages are never the best lever. Measured directly between neighbouring observed configurations, the payoffs scatter around the law's, mostly below it, as selection along the empirical frontier predicts (Section G).

Document length changes the allocation. A summary that keeps a fixed fraction of the document costs more for a longer document, whereas retrieval costs the same, and the gain from a summary depends on the fraction it keeps, not on the document's length (Section 3). The compute-optimal allocation therefore depends on the document (Figure 5; Table 11 gives examples). For a short document (19k tokens, the 10th percentile), summarize heavily: at about 150 TFLOPs the best allocation is the 4B model with a 37% summary. For a median document (99k tokens), retrieval comes first and a summary enters as the budget grows. For a long document (312k tokens, the 90th percentile), retrieval and a larger model dominate, and a summary barely enters even at 500 TFLOPs. The law's compute predictions match measured compute (median ratio 1.00, 80% of configurations within 0.93 to 1.12).

A compact rule. Wherever the optimum uses a summary of a length we observed, it follows a simple pattern. A summary enters at about 60 to 80 TFLOPs per question for short and long documents alike (58, 68 and 79 TFLOPs, median over document lengths of 10k to 50k, 50k to 250k and over 250k tokens). From there, the summary takes about half of the compute (median 53%, interquartile range 34 to 66%): about 60% for documents up to 250k tokens and 30% beyond, while retrieval stays near 2k tokens per question (4k for the longest documents). For the longest documents, the optimum drops the summary again once the 27B model is affordable (Figure 23).

How often a document is queried. So far the summary's one-time cost has been excluded. Almost every document in our benchmark has a single question, so spreading that cost over many questions is an assumption about deployment, and the allocation depends on it (Figure 4b). Charging the summary's cost at $1/Q$ per question, for Q questions per document: at $Q = 1$ a summary is never compute-optimal, and the best allocation is retrieval with a larger model, which at 500 and 1,500 TFLOPs reaches 47.8 and 50.0% against 54.8 and 59.1% when the summary is free. At $Q = 10$ a summary enters the optimum only from about 250 TFLOPs; from $Q = 100$ the best accuracy is

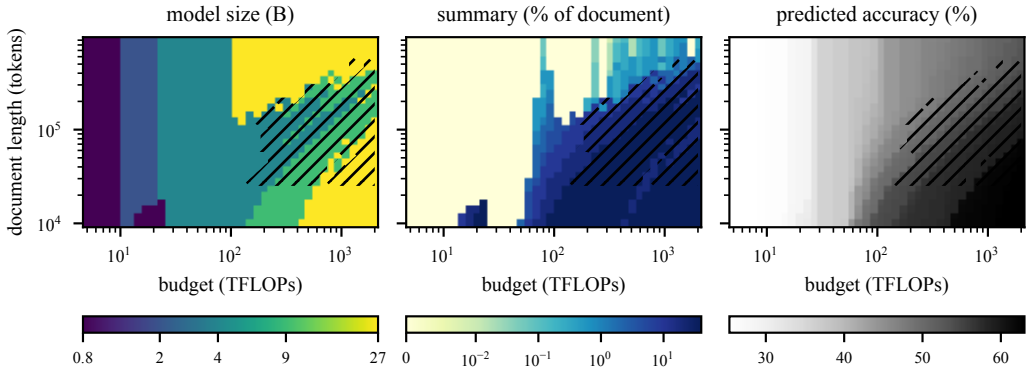

Figure 5: **Compute-optimal allocation over budget and document length** (the law, over lever combinations we ran; summary cost excluded). Left to right: the optimal model size, the optimal summary length, and the predicted accuracy. Hatched: the optimum would need a summary longer than any we observed (>10k tokens), so the law extrapolates there; Section G gives the best allocation within observed summary lengths, and the optimal retrieval.

within 2 points of the cost-excluded optimum (identical from $Q = 1{,}000$). Summaries are thus the compute-optimal choice for documents that are queried often, the setting they are built for, and not for one-off questions.

Allocation under constraints. A practitioner is often stuck with some choices, such as a retrieval service that returns two passages; the law handles this by fixing those levers and maximizing Equation (1) over the others. With the passages per call fixed at two, the best allocation loses a median of 0.7 points across budgets and at most 2: because the levers substitute, a constraint on one of them costs little as long as another supply lever is free.

Do agents allocate compute well on their own? We also ran the 9B agent with the number of calls left to it, under three instructions (minimize calls, stop when the information suffices, retrieve freely) at one 512-token passage per call, and under "stop when sufficient" over the whole retrieval grid (Figure 6). It makes a median of 3.5 to 4.6 calls with one passage per call and about 2 with larger per-call retrieval, re-retrieving a third of what it fetches. The law, never fit on these runs, predicts them at their measured distinct passages (RMSE 1.8 to 2.4 points against a noise of 2.1). With one passage per call the free agent is as efficient as the best forced-call 9B configuration at the same compute (within 0.3 points of the 9B law curve); with larger per-call retrieval it falls 2.1 points below. But every instruction leaves the agent about 7 points below the compute-optimal path at the same compute. Deciding how often to search is not where accuracy is lost; the choices that matter, the model size and the summary, are made before the agent runs.

A recipe. In our setting: (1) below about 4B parameters, spend compute on the model, since document information helps little; (2) past that, if documents are queried tens of times or more, grow the summary until it takes about half of each question's compute, the more so the shorter the documents; if they are queried once, grow retrieval and the model instead; (3) keep retrieval small and broad, a few thousand tokens in a few calls of several short passages; (4) read the whole document only when the budget is large enough, where it buys the last few points.

6 Related work

Scaling laws tell how to split *training* compute between parameters and tokens (Kaplan et al., 2020; Hoffmann et al., 2022); for retrieval, they concern training compute or data (Zhou et al., 2026; Singh et al., 2026), and data-constrained laws discount repeated training tokens (Muennighoff et al., 2023) much as ours discounts re-retrieved passages. Closest to ours, Yue et al. (2024) allocate *inference* compute for RAG with a single model, with a law additive in the logarithm of each lever; we add a

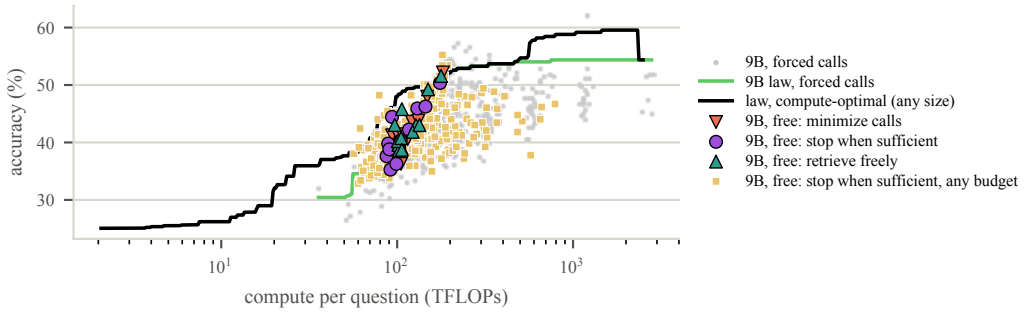

Figure 6: **Agents that choose their own number of calls** (9B; colored markers, one per configuration), against every forced-call 9B configuration (grey), the law's best 9B forced-call configuration at each compute (green), and the law's compute-optimal path over every size (black). Summary cost excluded.

question-independent summary and model size as levers, count generated tokens, which they exclude as negligible but which are the largest share of compute at most points in our setting, and couple the levers because they substitute. Test-time compute can substitute for model size on reasoning problems (Snell et al., 2024; Wu et al., 2024); we spend it on what enters the context. Long-context models and retrieval have been compared before (Xu et al., 2023; Li et al., 2024b; Bai et al., 2024); we place full context on the same compute axis as every summary and retrieval configuration. Systems that give a model a global view of a long document (Sarthi et al., 2024; Lee et al., 2024; Chen et al., 2023; Qian et al., 2024) show that summaries can help; we measure what one costs and buys against retrieval and model size. Section H discusses these and other related works in detail.

7 Limitations and Conclusion

Limitations, and what we expect to transfer. We studied one model family, Qwen3.5, on one benchmark, LongBench-v2, with one retriever and one summarizer (the pipeline model itself), and the law's parameters are specific to that setting. Its structure rests on properties that are not: a retrieved passage is chosen for the question while a summary is not, a summary's cost grows with document length while a passage's does not, and repeated passages add nothing. We therefore expect the qualitative findings to transfer—the supply levers substitute, and the shorter the documents, the more a summary pays—but not necessarily those that depend on how these agents behave: how many tokens each model size generates, which sets most of a trajectory's compute; how often an agent re-retrieves; and where the capability threshold falls (about 4B parameters here). Model size enters the law as a category, so it does not predict sizes we did not run. The law is validated within the lever ranges and combinations of Table 1; the optimum beyond them, hatched in Figure 5, is extrapolation, and the law is 0.25 to 0.3 points above noise when extrapolating to the largest per-call retrieval. Accuracy is on four-way multiple choice, whose guessing floor is built into the law's form, and open-ended answers may behave differently. We study agents that choose their own number of calls only at 9B. Almost every document in our benchmark has one question, so the number of questions a summary serves is modelled (Figure 4b), not observed. And our compute is theoretical FLOPs, which rank configurations by cost but are not wall-clock time or serving cost. Testing the law's structure on other model families and benchmarks, with the structural parameters fixed and only the per-size terms refit, is the next step.

Conclusion. Deciding how to spend inference compute on a long-document agent looks like a choice among many levers, but it reduces to two axes: document information, which summaries and retrieval both supply and which substitute for each other, and model capability, which sets how much the model gets out of that information. A 15-parameter law built on this structure predicts lever combinations it was not fit on close to the limit set by sampling noise, and its compute-optimal allocation is simple: past a capability threshold, and for documents queried often, give the summary about half of each question's compute and keep retrieval to a few thousand tokens.

AI USE STATEMENT

We used generative AI tools to help implement and run the experiments (writing and debugging code), to help with the analysis (including designing analyses, developing the predictive model and interpreting results), and to help write the paper (drafting and editing text, and creating figures). We have not used them to generate data sets, to translate, or for mathematical proofs. We have reviewed all AI-assisted work: we verified the code by rerunning the analyses and checking the reported numbers against their outputs, retrieved all references ourselves, and checked every statement attributed to a cited work against that work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses a public benchmark (LongBench-v2) and openly released models (Qwen3.5); it involves no human subjects and no personal data beyond what the public benchmark contains. Its aim is to make long-document agents cheaper to run for the same accuracy. We see no specific risk of harm beyond those of the language models themselves.

REPRODUCIBILITY STATEMENT

The task, agent, levers and grids are described in Section 2, with prompts, sampling settings, retrieval and summarizer details in Section C. The compute accounting, including its derivation for the hybrid architecture and its validation against the reference implementation, is in Section A; the checks that every trajectory was returned intact are in Section B. The law's fitting procedure, the candidate retrieval terms, the selection rule and the order in which predictions were recorded and scored are in Section F. We will release the code, the per-question results with their compute, the summaries, and the recorded prediction files.

REFERENCES

K. J. Arrow, H. B. Chenery, B. S. Minhas, and R. M. Solow. Capital-labor substitution and economic efficiency. *The Review of Economics and Statistics*, 43(3):225–250, 1961. ISSN 00346535, 15309142. URL http://www.jstor.org/stable/1927286.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench v2: Towards Deeper Understanding and Reasoning on Realistic Long-context Multitasks. *arXiv e-prints*, art. arXiv:2412.15204, December 2024. doi: 10.48550/arXiv.2412.15204.

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. Walking Down the Memory Maze: Beyond Context Limit through Interactive Reading. *arXiv e-prints*, art. arXiv:2310.05029, October 2023. doi: 10.48550/arXiv.2310.05029.

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pp. 758–759, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584836. doi: 10.1145/1571941.1572114. URL https://doi.org/10.1145/1571941.1572114.

Yair Feldman and Yoav Artzi. No Mean Feat: Simple, Strong Baselines for Context Compression. *arXiv e-prints*, art. arXiv:2510.20797, October 2025. doi: 10.48550/arXiv.2510.20797.

Jia-Chen Gu, Junyi Zhang, Di Wu, Yuankai Li, Kai-Wei Chang, and Nanyun Peng. BRIEF-Pro: Universal Context Compression with Short-to-Long Synthesis for Fast and Accurate Multi-Hop Reasoning. *arXiv e-prints*, art. arXiv:2510.13799, October 2025. doi: 10.48550/arXiv.2510.13799.

Ruishan Guo, Yibing Liu, Guoxin Ma, Yan Wang, Yueyang Zhang, Long Xia, Kecheng Chen, Zhiyuan Sun, and Daiting Shi. When Less is More: The LLM Scaling Paradox in Context Compression. *arXiv e-prints*, art. arXiv:2602.09789, February 2026. doi: 10.48550/arXiv.2602.09789.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training Compute-Optimal Large Language Models. *arXiv e-prints*, art. arXiv:2203.15556, March 2022. doi: 10.48550/arXiv.2203.15556.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning. *arXiv e-prints*, art. arXiv:2503.09516, March 2025. doi: 10.48550/arXiv.2503.09516.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models. *arXiv e-prints*, art. arXiv:2001.08361, January 2020. doi: 10.48550/arXiv.2001.08361.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. *arXiv e-prints*, art. arXiv:2309.06180, September 2023. doi: 10.48550/arXiv.2309.06180.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A Human-Inspired Reading Agent with Gist Memory of Very Long Contexts. *arXiv e-prints*, art. arXiv:2402.09727, February 2024. doi: 10.48550/arXiv.2402.09727.

Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, Wenbo Su, and Bo Zheng. GraphReader: Building Graph-based Agent to Enhance Long-Context Abilities of Large Language Models. *arXiv e-prints*, art. arXiv:2406.14550, June 2024a. doi: 10.48550/arXiv.2406.14550.

Yuqing Li, Jiangnan Li, Zheng Lin, Ziyan Zhou, Junjie Wu, Weiping Wang, Jie Zhou, and Mo Yu. Mindscape-Aware Retrieval Augmented Generation for Improved Long Context Understanding. *arXiv e-prints*, art. arXiv:2512.17220, December 2025. doi: 10.48550/arXiv.2512.17220.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. Retrieval Augmented Generation or Long-Context LLMs? A Comprehensive Study and Hybrid Approach. *arXiv e-prints*, art. arXiv:2407.16833, July 2024b. doi: 10.48550/arXiv.2407.16833.

Kyle Montgomery, David Park, Jianhong Tu, Michael Bendersky, Beliz Gunel, Dawn Song, and Chenguang Wang. Predicting Task Performance with Context-aware Scaling Laws. *arXiv e-prints*, art. arXiv:2510.14919, October 2025. doi: 10.48550/arXiv.2510.14919.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling Data-Constrained Language Models. *arXiv e-prints*, art. arXiv:2305.16264, May 2023. doi: 10.48550/arXiv.2305.16264.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient Context Window Extension of Large Language Models. *arXiv e-prints*, art. arXiv:2309.00071, August 2023. doi: 10.48550/arXiv.2309.00071.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. MemoRAG: Boosting Long Context Processing with Global Memory-Enhanced Retrieval Augmentation. *arXiv e-prints*, art. arXiv:2409.05591, September 2024. doi: 10.48550/arXiv.2409.05591.

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

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, April 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL `https://doi.org/10.1561/1500000019`.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. *arXiv e-prints*, art. arXiv:2401.18059, January 2024. doi: 10.48550/arXiv.2401.18059.

Noam Shazeer. GLU Variants Improve Transformer. *arXiv e-prints*, art. arXiv:2002.05202, February 2020. doi: 10.48550/arXiv.2002.05202.

Karan Singh, Michael Yu, Varun Gangal, Zhuofu Tao, Sachin Kumar, Emmy Liu, and Steven Y. Feng. To Memorize or to Retrieve: Scaling the Interaction Between Pretraining and Retrieval. *arXiv e-prints*, art. arXiv:2604.00715, April 2026. doi: 10.48550/arXiv.2604.00715.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. *arXiv e-prints*, art. arXiv:2408.03314, August 2024. doi: 10.48550/arXiv.2408.03314.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference Scaling Laws: An Empirical Analysis of Compute-Optimal Inference for Problem-Solving with Language Models. *arXiv e-prints*, art. arXiv:2408.00724, August 2024. doi: 10.48550/arXiv.2408.00724.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-Pack: Packed Resources For General Chinese Embeddings. *arXiv e-prints*, art. arXiv:2309.07597, September 2023. doi: 10.48550/arXiv.2309.07597.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets Long Context Large Language Models. *arXiv e-prints*, art. arXiv:2310.03025, October 2023. doi: 10.48550/arXiv.2310.03025.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with Delta Rule. *arXiv e-prints*, art. arXiv:2412.06464, December 2024. doi: 10.48550/arXiv.2412.06464.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv e-prints*, art. arXiv:2210.03629, October 2022. doi: 10.48550/arXiv.2210.03629.

Zhenrui Yue, Honglei Zhuang, Aijun Bai, Kai Hui, Rolf Jagerman, Hansi Zeng, Zhen Qin, Dong Wang, Xuanhui Wang, and Michael Bendersky. Inference Scaling for Long-Context Retrieval Augmented Generation. *arXiv e-prints*, art. arXiv:2410.04343, October 2024. doi: 10.48550/arXiv.2410.04343.

Jihao Zhao, Zhiyuan Ji, Simin Niu, Hanyu Wang, Feiyu Xiong, and Zhiyu Li. MoM: Mixtures of Scenario-Aware Document Memories for Retrieval-Augmented Generation Systems. *arXiv e-prints*, art. arXiv:2510.14252, October 2025. doi: 10.48550/arXiv.2510.14252.

Shu Zhou, Jinman Leng, Yufei Song, Xin Wang, Tao Fan, and Hao Wang. The retrieval bottleneck: Scaling laws for reinforcement learning in RAG. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32045–32069, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1478. URL https://aclanthology.org/2026.acl-long.1478/.

A Compute accounting

Per-token cost. We count 2 FLOPs per multiply–add for every matrix multiplication of a forward pass, plus the attention core and the Gated DeltaNet state update; elementwise operations (normalization, activations, residual additions, rotary embeddings) are excluded, as is usual, and embedding lookups are free. Every Qwen3.5 size is a hybrid of one gated full-attention layer for every three Gated DeltaNet layers (Yang et al., 2024), each followed by a SwiGLU MLP (Shazeer, 2020). A token at position p (attending to p tokens, itself included) costs

$$C(p) = C_{\text{dense}} + C_{\text{attn}}\, p \quad \big[\, + \; C_{\text{head}} \text{ if the model samples a token there} \big], \tag{3}$$

where C_{dense} sums the projections, the MLPs, the DeltaNet convolution and its recurrent update (7 FLOPs per state element, from its per-token recurrent form; the chunked kernel's cost depends on its chunk size, not on the model), $C_{\text{attn}} = 4\, n_{\text{attn layers}}\, n_{\text{heads}}\, d_{\text{head}}$ covers $QK^\top$ and the attention-weighted values in the full-attention layers only, and $C_{\text{head}} = 2\, d_{\text{model}}\, |V|$ is the output projection, charged only for generated tokens: prompt, summary and passage tokens need no logits. The distinction matters for small models, where the output projection is 49% of the dense per-token cost at 0.8B, against 15% at 9B and 5% at 27B. The 27B model is served in FP8; quantization changes bytes, not multiply–adds, so we count its bf16 FLOPs.

Per-trajectory cost. With ideal key–value caching every token is computed once, when it enters the context, so a trajectory costs $\sum_t C(p_t)$ over its tokens at their own positions. Because our trajectories are append-only (each turn's prompt extends the previous turn's prompt and output), this depends only on the tokens and their positions; we check the property on every trajectory and drop the two units (of 860,764) that violate it, where a long generation was re-tokenized a few tokens shorter in the next prompt. Tokens a turn generated but the server did not return (a median of 2 per turn) are charged once, at the positions they were generated at, but do not enter the context. We split each question's compute into four parts — the prompt (system prompt, question and choices, chat template), the summary or document, the retrieved passages (tool results; each call's result carries exactly 54 template tokens beyond the passages, and the last one also the short reminder of Section C) and the generated tokens — which sum exactly to the total for every unit.

Validation. Against the reference HuggingFace implementation, for every size: the parameter count our architecture description implies equals the model's own (built from its configuration without weights), and the FLOPs of every matrix multiplication and of the causal attention core match PyTorch's FLOP counter on a truncated model (the first four layers at full width) exactly.

Measured against nominal compute, and generation. Nominal compute is the same question's compute with the generated tokens fixed at one reference rate (the 9B model's without a summary at two calls, 1,608 per turn), so only the lever's own tokens vary. Measured over nominal is 1.01 to 1.06 across passages per call, 0.90 to 1.03 across passage lengths and 0.90 to 1.10 across summary lengths (9B), so these levers cost what their tokens imply. Model size and calls deviate, because generation changes with them (Table 2): the 27B model generates far fewer tokens than the 9B at every number of calls, so it costs only about 1.25 times as much without a summary despite 3.5 times the non-embedding parameters, and more calls mean less reasoning per turn (2,356 generated tokens per turn without retrieval against 667 with eight calls, at 9B). Generation length is agent behaviour, specific to Qwen3.5.

Table 2: **Mean generated tokens per question** on the main grid (one 512-token passage per call), by model size, condition and number of calls.

	calls	0.8B	2B	4B	9B	27B
no summary	0	1,046	2,297	1,956	2,356	1,500
	2	2,147	3,572	2,275	4,823	977
	8	3,172	4,216	3,161	6,002	1,649
summary	0	4,524	5,382	4,468	4,319	3,512
	2	5,970	7,640	4,826	5,270	2,573
	8	6,877	9,065	6,325	6,573	3,165

Amortization. We exclude the one-time cost of writing a summary. For each summary, the number of questions about its document after which its cost falls below a share s of each question's own cost is $C_{\text{summary}}/(s\, C_{\text{question}})$; Table 3 gives its distribution over every question with a summary. There is no break-even point: not summarizing costs nothing to build, so the only question is when the summary's cost becomes negligible.

Table 3: **Questions per document after which a summary's one-time cost falls below 10% or 1% of each question's own cost**, over every summary configuration.

	below 10%		below 1%	
size	median	90th pct.	median	90th pct.
0.8B	234	3,102	2,337	31,016
2B	129	1,118	1,286	11,181
4B	192	1,716	1,915	17,158
9B	148	1,105	1,476	11,055
27B	219	1,638	2,191	16,378

B EXPERIMENTAL INTEGRITY

Damaged turns. A non-final turn is damaged if the server returned a reasoning trace containing a stray think tag, a malformed tool call, or more than 16 generated tokens that it did not return; the agent would then see something other than what the model generated. Damaged turns are re-drawn live. Runs that began before this check existed were scanned after the fact: damaged records (about 0.2% in the first scan) were removed and the units re-run. A run counts as verified when every expected unit is present and no turn is damaged. A tool call whose arguments contain a lone UTF-16 surrogate is also re-drawn (no tokenizer accepts it), and a trajectory that exceeds the context window is recorded as incorrect, not dropped; no record in the paper needed the latter.

Chat-control strings. Seven LongBench-v2 documents, all code repositories, contain chat template control strings as code text, which the tokenizer turns into real control tokens. Only two are in our study, and we found no effect on the trajectories that read them.

C SETUP DETAILS

Data. LongBench-v2 (Bai et al., 2024): 503 four-way multiple-choice questions about 462 documents; we keep the 469 questions (434 documents) whose document fits the summarizer's context budget of about 1M tokens (median 99k tokens; 10th and 90th percentiles 19k and 312k).

Models and serving. Qwen3.5 0.8B, 2B, 4B, 9B and 27B (FP8), served with vLLM (Kwon et al., 2023) at their native 262k-token context; summarizing the longest documents and full context use YaRN rope scaling (Peng et al., 2023) to about 1M tokens, on separate servers, since YaRN changes the computation even for short inputs. The agent samples with temperature 1.0, top-p 0.95, top-k 20 and presence penalty 1.5, with reasoning enabled and up to 32,768 output tokens per turn.

Prompts. The system prompt names the condition ("You are given a summary of a document and a multiple-choice question about the document's contents."; without a summary: "You are given a multiple-choice question about the contents of a document."), followed by the retrieval instruction, which with forced calls is "You must call search_document exactly f times before giving your final answer. Once you have used all f calls, answer using only the information you have gathered – do not attempt to call search_document again." ("You must answer directly, without calling search_document." when $f = 0$), and ends with the answer format, "Format your response as follows: "The correct answer is (insert answer here)"." The user message holds the summary (if any) between text tags, then "What is the correct answer to this question:", the question and the lettered choices under "Choices:". The call count is enforced by the serving request: the tool is required on every turn until f calls are made and disabled afterwards, and the last tool result ends with "You have now used all f allowed calls to search_document. Answer the question now, using only the information you have gathered – do not attempt to call search_document again." Full context's system prompt is "You are given the full text of a document and a multiple-choice question about its contents. Answer using only the document text below.", and its user message places the whole document between text tags ahead of the question.

Retrieval. One tool, search_document, takes a free-text query and returns the top k passages of c tokens (recursive splitter, 10% overlap), ranked by reciprocal rank fusion (Cormack et al.,

2009) ($k = 60$, equal weights) of dense retrieval with BGE-large-en-v1.5 (Xiao et al., 2023) and BM25 (Robertson & Zaragoza, 2009).

Summaries. The pipeline model summarizes each document in one call ("Summarize the following document in about N words", with N = target fraction $\times$ document tokens), without reasoning, with temperature 0.7, top-p 0.8, top-k 20 and presence penalty 1.5 (for 2B, as its model card recommends, temperature 1.0, top-p 1.0 and presence penalty 2.0). Target fractions follow a 17-value octave grid from 0.0011% to 73.9%; a target outside 10 to 10,000 tokens is dropped for that document rather than clipped, so each document gets about ten lengths. Summaries often miss their target, so every analysis uses the realized length (summary tokens over document tokens), binned on the same octave grid.

Grids. Table 1 lists the grids and their sizes. The main grid crosses every summary length with 0, 1, 2, 4 or 8 calls of one 512-token passage; the retrieval grid crosses 1, 2, 4 or 8 passages per call with passage lengths from 64 to 4,096 tokens, at two calls and every summary length (27B up to 2,048 tokens); the calls $\times$ retrieval cells cross 1, 4 or 8 calls with eight 512-token passages per call (4B, 9B, 27B) and with one or eight 2,048-token passages (4B, 9B), without a summary and at three target lengths (0.14%, 1.15% and 9.2%).

D PER-LEVER ANALYSES

Clustering by document. Our tests cluster standard errors by question. Since 47 of the 469 questions share 12 documents, we reran every test on the final data clustered by document instead: no conclusion changes. Among the p-values quoted in this appendix, two borderline ones cross 0.05: the dip at 2B over the shortest summaries ($p = 0.047$ becomes 0.087) and the concavity of the full-context curve from 2B up ($p = 0.049$ becomes 0.056).

The 0.8B model. No lever moves the 0.8B model off chance: the odds ratios of one to eight calls against none are 0.83 to 1.06 (all $p \geq 0.16$), a summary's gain over none is about 3 points, and its retrieval slopes are near zero (Table 5); the one nominally significant term, passages per call without a summary ($+0.035$, $p = 0.048$), is consistent with chance among the roughly twenty tests we ran at this size.

Summary length. Table 4 compares functional forms for accuracy against summary length (per-question likelihood, summary conditions, at least one call). A form that is flat and then rises — broken (flat, then linear in log length), sigmoid, or quadratic in log length — is within about 8 AIC of the best at every size, while a log-linear form is 28 to 56 behind. The broken fit's threshold is at 0.21, 0.80 and 0.70% of the document at 4B, 9B and 27B, with slopes of 3.0, 4.4 and 3.7 points per e-fold above it. There is no dip below the threshold: the slope over the shortest summaries (up to 0.1, 0.2 or 0.4% of the document) is never significantly negative, except marginally at 2B up to 0.2% (odds ratio 0.95 per e-fold, $p = 0.047$).

Table 4: **Functional forms for summary length**: AIC difference from the best form at each size (lower is better).

form	4B	9B	27B
sigmoid (ceiling = full context)	0.0	0.8	8.0
sigmoid (free ceiling)	1.0	0.0	2.5
quadratic in log length	1.9	3.3	0.0
broken (flat, then linear)	7.1	2.7	4.7
floor + power	6.7	16.3	18.6
log-linear	27.9	56.2	50.0

Calls. With a summary, the odds ratio of eight calls against none, at matched summary length and document length, is 1.27, 1.34 and 1.32 at 4B, 9B and 27B, and no size shows curvature in $\log_2$ calls from one to eight (every quadratic term $p \geq 0.09$). At 0.8B and 2B, calls do not help with a summary; without one, the 2B model gains from the first call and then plateaus.

Per-call retrieval. Table 5 gives the log-odds slope per doubling of passages per call and of passage length at each size, and the split test: at a fixed number of retrieved tokens per call, the gain from splitting them into twice as many passages. The split is positive at every size and significant at 9B and 27B, and with a summary at 2B and 4B; at 9B it holds when excluding questions whose retrieval could reach the whole document ($+0.075$) and with a quadratic in volume ($+0.070$). Curvature is absent except for mild diminishing returns in passages per call with a summary at 4B (-0.011 per doubling squared, $p = 0.022$) and 27B (-0.010, $p = 0.043$), and two terms at 2B without a summary (passage length -0.012, $p = 0.021$; passages per call $+0.035$, $p = 0.029$), near chance.

Table 5: **Retrieval slopes** (log-odds per doubling, two calls; without a summary / with one) and the split test at fixed retrieved volume, with p-values.

size	passages per call	passage length	split, no summary	split, summary
0.8B	$+0.035$ / $+0.003$	$+0.012$ / $+0.002$	$+0.027$ (0.19)	$+0.001$ (0.93)
2B	$+0.028$ / $+0.023$	$+0.013$ / $+0.009$	$+0.030$ (0.11)	$+0.015$ (0.017)
4B	$+0.057$ / $+0.063$	$+0.039$ / $+0.030$	$+0.023$ (0.18)	$+0.032$ (7×10^{-6})
9B	$+0.116$ / $+0.067$	$+0.046$ / $+0.033$	$+0.073$ (10^{-5})	$+0.036$ (6×10^{-8})
27B	$+0.102$ / $+0.071$	$+0.059$ / $+0.049$	$+0.043$ (0.02)	$+0.023$ (0.008)

Model size. Accuracy rises with model size almost everywhere (the exceptions are within noise). From 2B up, the summary and full-context curves are concave in log parameters (full context only weakly, $p = 0.049$); without a summary the curve is linear, with no saturation by 9B (9B to 27B: odds ratio 1.37, $p = 0.002$). The largest step for the summary curves is 2B to 4B (odds ratios 1.59 and 1.71 for the two summary lengths), where the capability threshold lies. From 4B up, the summary-length curves are parallel across sizes (size $\times$ summary length interaction, $p = 0.67$), and those for the number of calls nearly so (27B's is slightly steeper). A long summary's advantage over none is +3, +5, +15, +16 and +17 points from 0.8B to 27B, on matched questions.

Document length. At matched summary length, document length does not change the gain from a longer summary (interaction -0.15 points per unit of log document length, 95% CI $[-1.16, +0.87]$, pooled over 4B to 27B).

E INTERACTIONS PER SIZE

Table 6 gives the payoff of each retrieval lever per doubling, by summary length, at the three sizes past the threshold, and how much each e-fold of summary length cuts it. The substitution is as strong at 27B as at 9B and weaker at 4B. Seen from the summary's side, a long summary's advantage over none at the centre of the retrieval grid is +9.3, +9.6 and +12.0 points at 9B, 4B and 27B, and shrinks with each doubling of passages per call by 1.9 and 1.4 points at 9B and 27B ($p = 2 \times 10^{-4}$ and 0.004), but not significantly at 4B ($p = 0.4$). For the number of calls (main grid), the payoff per doubling falls from +3.3 without a summary to +0.3 with a long one at 9B, and from +1.8 to +0.9 at 27B. Figure 8 shows the substitution at each of the three sizes.

Table 6: **Summaries and retrieval substitute, by size**: accuracy points per doubling of passages per call and of passage length, by summary length ("below" = below the size's threshold), and the change in that payoff per e-fold of summary length (95% CI, p).

size	lever	none	below	to 4.6%	$\geq$4.6%	per e-fold
4B	passages	+1.3	+1.7	+1.5	+0.9	-0.22 [-0.40, -0.05], 0.013
	length	+0.9	+0.8	+0.7	+0.5	-0.08 [-0.18, $+0.01$], 0.085
9B	passages	+2.8	+2.0	+1.3	+0.8	-0.53 [-0.78, -0.27], 5×10^{-5}
	length	+1.1	+1.0	+0.6	+0.5	-0.21 [-0.36, -0.05], 0.008
27B	passages	+2.5	+2.1	+1.7	+1.0	-0.44 [-0.67, -0.22], 1×10^{-4}
	length	+1.5	+1.5	+1.0	+0.8	-0.28 [-0.44, -0.12], 7×10^{-4}

F THE LAW: FITTING AND VALIDATION

Fitting. Targets are the adjusted accuracies of the 2,456 configurations with at least 20 questions (full context excluded); weights are their numbers of questions. We minimize weighted squared error with bounded least squares from three starting points and keep the best; ρ is bounded to $[0.05, 1]$ and ϵ is fitted on the log scale. Table 7 gives the fitted parameters. Uncertainty bands come from 50 refits on question-bootstrap resamples, each with its own adjustment; they reflect parameter uncertainty, conditional on the law's form.

Table 7: **Fitted parameters of the law** (Equations (1) and (2)).

	0.8B	2B	4B	9B	27B
θ_N	-0.22	-0.62	0.67	0.75	0.89
β_N	0.79	0.41	0.41	0.38	0.34
$\rho = 0.744$, $w = 0.247$, $\gamma_d = 0.836$, $\gamma_c = 0.437$, $\epsilon = 1.8 \times 10^{-4}$					

Candidate retrieval terms and the selection rule. We compared four retrieval terms, each with the same 15 parameters except where stated: (i) nominal, $R = f\,k^{\gamma_k}(c/512)^{\gamma_c}/1000$, the form fit and recorded first; (ii) saturation, the nominal term passed through $R/(1 + R/K)$, with one more parameter K; (iii) distinct passages, Equation (2); and (iv) distinct, rank-weighted, $R = D\,k^{\gamma_k - 1}(c/512)^{\gamma_c}/1000$, in which lower-ranked passages within a call are worth less. The rule, fixed before comparing them: the lowest mean RMSE over the structured holdouts below, with D predicted by the stage-1 model as a practitioner would use it; within 0.05 of the best, prefer the better of the two distinct-passage versions, since they model a measured mechanism. On the final data the means are 3.20 (nominal), 2.93 (saturation), 2.972 (distinct) and 2.977 (distinct, rank-weighted), so the rule selects distinct passages. On the data available when the second set of predictions was recorded, the rule had selected the rank-weighted version; the two tie, and the second test scored both (2.13 and 2.16).

Stage 1: distinct passages from the levers. $\log_2 \frac{D}{f\,k} = \log_2 f\,(a + b\log_2 k + c\log_2 \frac{c}{512} + d\log_2 f + o_N)$, fit by least squares on per-configuration means weighted by units: $a = -0.244$, $b = -0.058$, $c = -0.044$, $d = -0.016$, and size offsets o_N from $+0.045$ (2B) to $+0.154$ (27B) relative to 0.8B. The non-redundant fraction thus falls by about 16% per doubling of calls at 0.8B, faster with more or longer passages per call, and more slowly for larger models. It predicts D to within 2.8% (median absolute error) in sample, 4.0% on the calls $\times$ retrieval cells when fit without them, and 3.1% on every configuration run after the first predictions when fit on the data available then.

Held-out validation. Each holdout refits the law, the flexible surface and the adjustment without the held-out block (Table 8). The flexible surface is linear in the logit: per-size intercepts, per-size slopes in the summary-length hinge and in $\log_2$ calls, shared slopes in $\log_2$ passages per call and $\log_2$ passage length, a no-retrieval offset, and the hinge's interaction with each retrieval term (22 parameters).

Table 8: **Held-out RMSE (accuracy points)**: the flexible surface, the nominal law, and the law with distinct passages (predicted by stage 1; measured in brackets), against the held-out configurations' own noise.

held out	flexible (22)	nominal (15)	distinct (15)	noise
random 5-fold	2.35	2.39	2.35 [2.34]	2.47
eight calls	2.39	3.06	2.31 [2.31]	2.64
eight passages per call	2.58	3.06	2.80 [2.71]	2.53
4,096-token passages	2.65	2.72	2.79 [2.75]	2.48
two longest summary lengths	4.55	4.04	4.43 [4.42]	4.86
long summary $\times$ large budget	3.11	3.19	3.09 [3.10]	3.55
calls $\times$ retrieval cells	2.41	3.12	2.41 [2.36]	2.92

Units. With the nominal retrieval term, counting summary information as a fraction of the document and retrieval in passages gives a random-fold RMSE of 2.39, against 2.51 with both as fractions of the document and 2.65 with both in tokens (6.36 on the calls $\times$ retrieval holdout).

The implied threshold. On the 9B retrieval grid, grouped from the smallest to the largest per-call retrieval settings, the fitted summary-length thresholds are 0.54, 0.80, 0.80 and 1.73% of the document, and the law implies 0.60, 1.30, 2.28 and 4.98%. Redundancy does not explain the lag, since the law counts distinct passages and 67 to 94% of the passages on this grid are distinct.

Order of events. Table 9 gives the order in which the law was fit, its predictions recorded and scored, so that it is clear which results were predictions.

Table 9: **Order of events.**

1	Fit the nominal law on the 770 configurations then available (main grid at every size, retrieval budget at 9B); record its predictions for every configuration still running (1,376 cells).
2	Test 1: score the recorded predictions on the configurations that finished (1,272 with a recorded prediction), without refitting (Table 10).
3	Diagnose the calls $\times$ retrieval miss: re-retrieval, measured. Define four candidate retrieval terms and the selection rule; hold out the configurations still being verified.
4	Record all four candidates' predictions for the held-out configurations (228 cells).
5	Test 2: score them once verified (152 configurations; Section 4).
6	Refit every model on all data; apply the selection rule (distinct passages); run the held-out validation (Table 8) and the allocation (Section 5).

Table 10: **Predicting unseen lever combinations.** The law, fit on the main grid at every size and the retrieval grid at 9B (770 configurations), predicts every configuration run afterwards that has a recorded prediction, without refitting. RMSE and mean error (predicted minus observed) in accuracy points; noise is the RMSE the observed accuracies' own sampling error alone would produce; "within band" is the share of configurations inside the prediction's 95% parameter band widened by that noise.

new configurations	sizes	n	RMSE	mean error	noise	within band
all		1,272	2.33	-0.01	2.29	97%
retrieval	0.8B, 2B, 4B, 27B	1,188	2.28	-0.09	2.28	97%
calls $\times$ retrieval	4B, 9B, 27B	84	3.05	$+1.30$	2.44	90%

G ALLOCATION: ADDITIONAL RESULTS

Where compute goes. Along the empirical frontier, generated tokens take most of the compute at the cheaper points (57 to 84%); the summary's share grows with its length, to about half at the cheaper 27B points, where it exceeds that of generated tokens; retrieved passages rarely exceed a quarter (Figure 14).

Allocation details. Table 11 gives the compute-optimal allocation for a short, median and long document at three budgets. In the compact rule of Section 5, the summary's compute grows nearly in proportion to the budget ($\propto C^{0.84}L^{-0.16}$, $R^2 = 0.88$, for budget C and document length L). When the summary is charged at $1/Q$ (Figure 13), it enters the optimum at about 80 TFLOPs from $Q = 100$, against 64 when its cost is excluded. With the passages per call fixed at two, the best allocation reaches 39.8, 49.2 and 54.6% at 80, 200 and 580 TFLOPs, against 40.8, 49.3 and 55.3% unconstrained; its largest loss, 2 points, is around 40 TFLOPs. Against two forced calls with the same per-call retrieval, the free-choice agent over the retrieval grid saves 21% of the compute for 0.9 points of accuracy.

Table 12 lists the empirical frontier: the best observed configuration at each budget, over configurations with at least 100 questions. Its accuracies are biased upward by selection (Section 5), so we use it to show which configurations are efficient, not how accurate they are.

Table 11: **Compute-optimal allocation by document length** (the law, over lever combinations we ran; the 10th, 50th and 90th percentiles of document length; summary cost excluded). Retrieval is calls $\times$ passages per call $\times$ passage length; accuracy is predicted.

budget	short (19k tokens)	median (99k)	long (312k)
50 TFLOPs	4B, 0.6%, $2 \times 8 \times 64$: 38%	4B, none, $2 \times 4 \times 512$: 37%	4B, none, $2 \times 4 \times 512$: 37%
150 TFLOPs	4B, 37%, $1 \times 8 \times 512$: 54%	4B, 9.2%, $2 \times 8 \times 64$: 46%	27B, none, $2 \times 8 \times 64$: 44%
500 TFLOPs	27B, 18%, $2 \times 8 \times 128$: 58%	4B, 37%, $2 \times 8 \times 256$: 55%	27B, 0.1%, $2 \times 8 \times 512$: 48%

Table 12: **The empirical frontier** (adjusted; retrieval as calls $\times$ passages per call $\times$ passage length; 95% intervals).

	size	summary	retrieval	TFLOPs	accuracy (%)	questions
1	0.8B	none	none	2.0	26.0 [22.6, 29.7]	469
2	0.8B	none	$2 \times 1 \times 128$	3.9	27.5 [22.4, 31.1]	469
3	0.8B	none	$2 \times 4 \times 256$	5.8	28.1 [24.3, 31.8]	469
4	0.8B	none	$2 \times 8 \times 128$	5.9	29.0 [24.3, 33.6]	469
5	2B	none	$2 \times 1 \times 64$	11.1	30.5 [27.0, 34.9]	469
6	2B	none	$2 \times 1 \times 256$	13.4	31.1 [27.4, 35.3]	469
7	0.8B	4.6%	none	14.3	32.1 [27.5, 38.5]	163
8	2B	none	$2 \times 8 \times 128$	19.1	32.2 [28.9, 37.1]	469
9	4B	none	$2 \times 1 \times 128$	20.0	33.9 [30.1, 38.2]	469
10	4B	none	$2 \times 2 \times 64$	20.8	34.1 [31.0, 39.4]	469
11	4B	none	$2 \times 1 \times 256$	23.1	35.0 [30.7, 39.3]	469
12	4B	none	$2 \times 1 \times 512$	27.8	35.1 [32.3, 38.8]	469
13	4B	none	$2 \times 4 \times 128$	28.4	36.9 [32.4, 40.9]	469
14	0.8B	9.2%	$2 \times 8 \times 256$	29.1	38.4 [31.4, 46.1]	107
15	4B	0.018%	$2 \times 2 \times 128$	31.4	39.4 [34.1, 45.7]	187
16	4B	none	$2 \times 4 \times 512$	52.1	40.1 [36.8, 44.6]	469
17	4B	0.58%	$2 \times 4 \times 64$	52.7	41.5 [38.0, 45.2]	385
18	4B	2.3%	$2 \times 2 \times 64$	63.7	41.7 [37.8, 46.5]	319
19	4B	0.58%	$2 \times 1 \times 1024$	64.5	42.0 [37.8, 45.3]	385
20	4B	2.3%	$2 \times 4 \times 64$	66.7	43.5 [39.6, 48.4]	319
21	4B	4.6%	$2 \times 1 \times 128$	73.3	44.8 [39.7, 49.9]	228
22	4B	4.6%	$2 \times 2 \times 64$	73.6	47.2 [42.1, 52.3]	228
23	4B	9.2%	$2 \times 1 \times 128$	86.3	47.9 [41.8, 54.1]	167
24	4B	18%	none	97.7	48.8 [43.0, 56.0]	111
25	4B	18%	$2 \times 4 \times 64$	106	49.4 [43.2, 55.8]	111
26	4B	18%	$2 \times 1 \times 256$	109	50.1 [42.7, 56.7]	111
27	4B	18%	$2 \times 2 \times 256$	114	52.8 [45.6, 60.3]	111
28	4B	18%	$2 \times 8 \times 256$	147	53.5 [45.5, 60.4]	111
29	9B	9.2%	$2 \times 4 \times 128$	173	54.0 [47.8, 60.3]	175
30	9B	18%	$2 \times 1 \times 128$	189	54.0 [46.8, 61.1]	121
31	9B	18%	$1 \times 1 \times 512$	195	54.6 [47.2, 61.5]	121
32	9B	18%	$2 \times 1 \times 512$	206	56.5 [50.1, 61.7]	121
33	9B	18%	$2 \times 2 \times 512$	218	57.1 [51.3, 63.9]	121
34	27B	18%	$2 \times 1 \times 128$	570	57.6 [52.9, 64.2]	140
35	27B	18%	$2 \times 4 \times 128$	610	58.7 [51.8, 65.8]	140
36	27B	18%	$2 \times 8 \times 64$	615	59.8 [53.8, 65.5]	140
37	27B	18%	$2 \times 1 \times 2048$	792	62.6 [56.8, 68.9]	140
38	27B	18%	$2 \times 8 \times 2048$	2,348	62.6 [55.0, 70.0]	140
39	27B	–	full context	6,712	64.8 [60.3, 69.6]	469

H Related work in detail

Scaling laws and compute-optimal allocation. Scaling laws predict a model's loss from its parameters, data and training compute (Kaplan et al., 2020), and tell how to split a training budget between parameters and tokens (Hoffmann et al., 2022). Two extensions are close to our setting in form. Data-constrained scaling laws (Muennighoff et al., 2023) discount repeated training tokens, which are worth less than new ones, through an "effective data" term; our law discounts re-retrieved passages in the same spirit, more bluntly: it counts only distinct ones, so a repeated passage adds nothing. Context-aware scaling laws (Montgomery et al., 2025) model downstream accuracy jointly in training compute and in the amount of context given to the model; we model it in inference-time levers that decide what that context contains, and in how they substitute for one another. For retrieval, Zhou et al. (2026) derive sigmoidal scaling laws for reinforcement learning in RAG and find that retrieval quality, not the reinforcement-learning compute, sets the ceiling, and Singh et al. (2026)

study how to split data between pretraining and a retrieval datastore, finding that retrieval's value depends on model capacity. Both concern training compute or data; we allocate inference compute.

Inference-time compute. Closest to ours, Yue et al. (2024) allocate inference compute for RAG with a single model over retrieved documents, in-context examples and generation steps, and fit a model that is additive in the logarithm of each lever on a logit scale. We add a question-independent summary and model size as levers, count generated tokens, which they exclude as negligible but which are the largest share of compute at most points in our setting, and couple the levers in the law because they substitute. More broadly, test-time compute can substitute for model size: Snell et al. (2024) allocate a test-time budget per prompt across search and revision strategies and, in a FLOPs-matched comparison, outperform a 14$\times$ larger model on problems where the smaller one has non-trivial success, and Wu et al. (2024) find that smaller models with sampling or tree search can be Pareto-optimal against larger ones. Both spend inference compute on generation for reasoning problems. We spend it on what enters the context, and find that the answer depends on what the compute buys: from about 60 TFLOPs per question, a longer summary beats a larger model, while more retrieval rarely does. Our agent interleaves reasoning with search calls, as in ReAct (Yao et al., 2022), and Search-R1 (Jin et al., 2025) trains a model with reinforcement learning to decide when and what to search; our free-choice runs suggest that, for a prompted 9B agent, deciding how often to search is not where accuracy is lost.

Long context versus retrieval. Xu et al. (2023) find that a 4k-token model with retrieval can match a model extended to 16k tokens at much lower computation, and that retrieval helps even at long context lengths; Li et al. (2024b) find that long-context models outperform RAG on average when they can read everything, at a much higher cost, and route queries between the two. On LongBench-v2, Bai et al. (2024) find that retrieving 32k tokens beats reading the context truncated to 128k for two of the three models they test, but not for GPT-4o. We place full context on the same compute axis as every summary and retrieval configuration: it is the most accurate option we measured with at least 100 questions, but only at budgets where little else is left to buy.

Global representations for long-document agents. Many systems give a model a global view of a long input alongside, or instead of, retrieval: trees of recursive summaries to retrieve from or navigate (Sarthi et al., 2024; Chen et al., 2023), gist memories with look-ups into the text (Lee et al., 2024), graphs explored by an agent (Li et al., 2024a), a learned global memory that drafts clues for retrieval (Qian et al., 2024), hierarchical summaries that condition retriever and generator (Li et al., 2025), and document memories built from logical outlines (Zhao et al., 2025). These systems show that a global representation can help; we treat the simplest one, a summary in context, as a compute lever and measure what it costs and buys against retrieval and model size.

Context compression. A summary is a form of hard context compression. BRIEF-Pro compresses retrieved documents into query-focused summaries of controllable length (Gu et al., 2025); our summaries are question-independent and written once per document. Compressors are not interchangeable: Feldman & Artzi (2025) find that simple soft-compression baselines strongly outperform the widely used causal compression-token approach, and Guo et al. (2026) that larger compressors can reconstruct contexts less faithfully; we therefore fix one summarization method and state it as a scope limit.

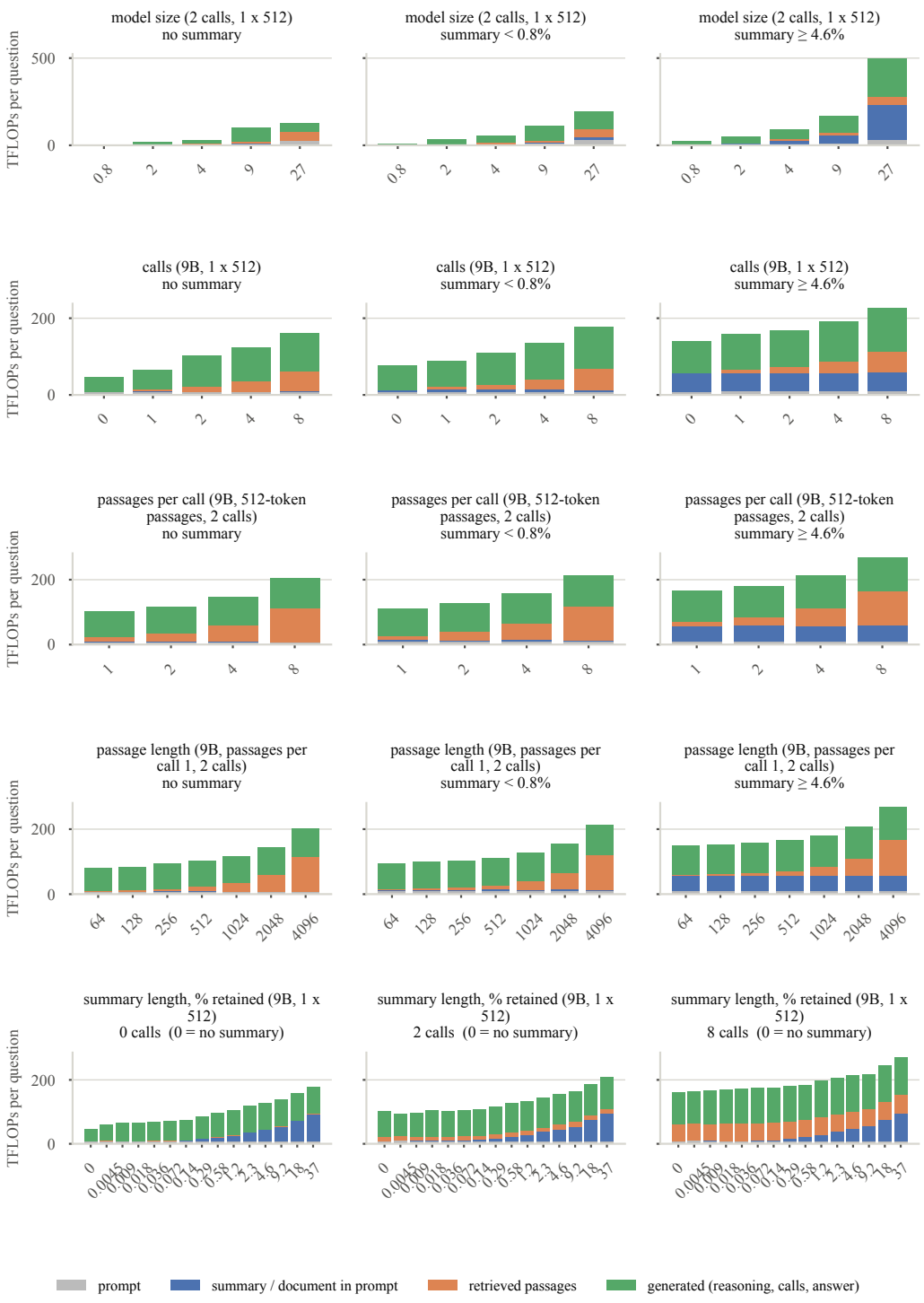

Figure 7: **Compute by part as each lever varies alone** (9B unless the panel varies model size; columns: no summary, a summary below 0.8%, and one of at least 4.6%; bottom row: summary length at 0, 2 and 8 calls). Generated tokens are the largest part in most settings; the summary adds a fixed cost per question; retrieved passages grow with calls, passages per call and passage length.

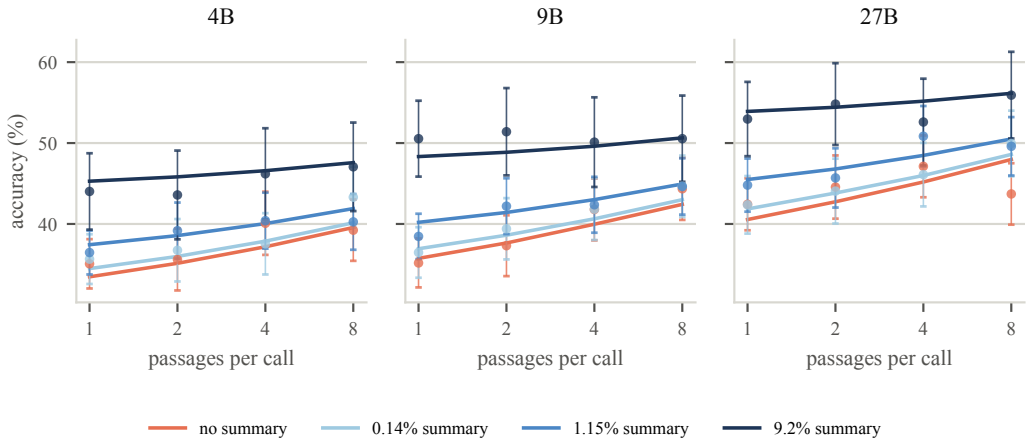

Figure 8: **Summaries and retrieval substitute, at every size past the threshold.** Accuracy against passages per call (two calls of 512-token passages) without a summary and with three summary lengths; points with 95% intervals, lines the law of Section 4. The longer the summary, the flatter the line.

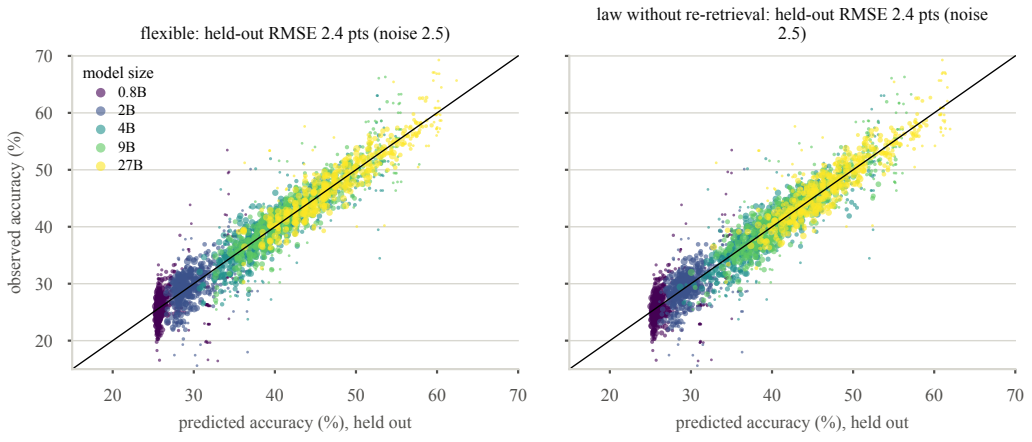

Figure 9: **Held-out predictions of the flexible 22-parameter surface and of the law without re-retrieval** (random 5-fold cross-validation over configurations, question effects re-estimated without the held-out fold).

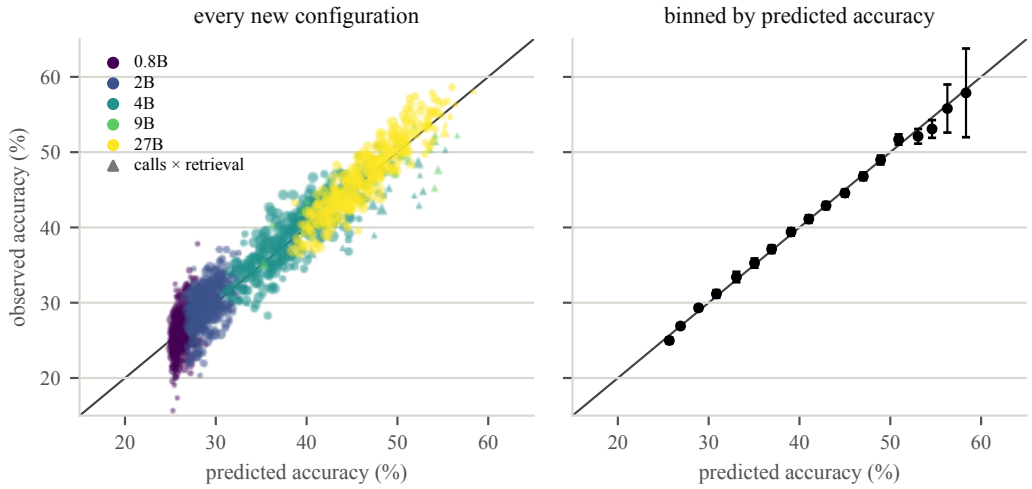

Figure 10: **Test 1, with calibration** (Table 10). Left: the law's recorded predictions against the observed accuracy of every configuration run afterwards (as Figure 3a). Right: the same configurations binned by predicted accuracy (2-point bins, 95% intervals).

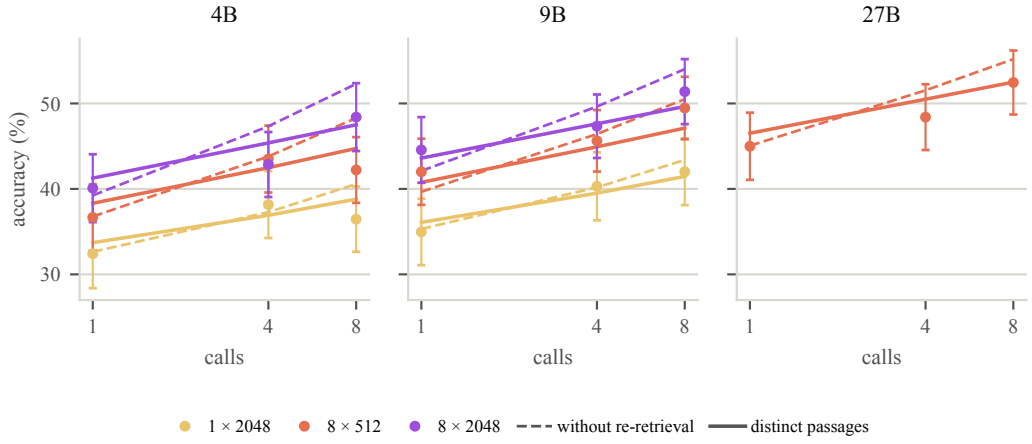

Figure 11: **Re-retrieval at every size** (as Figure 3b). Accuracy against the number of calls for three per-call retrieval settings (passages × tokens), without a summary; points observed (95% intervals). Dashed: the recorded predictions of the law without re-retrieval. Solid: the law with distinct passages, fit on the same 770 configurations, with distinct passages predicted from the levers. Counting repeats as new passages over-predicts the configurations with many calls and many passages per call.

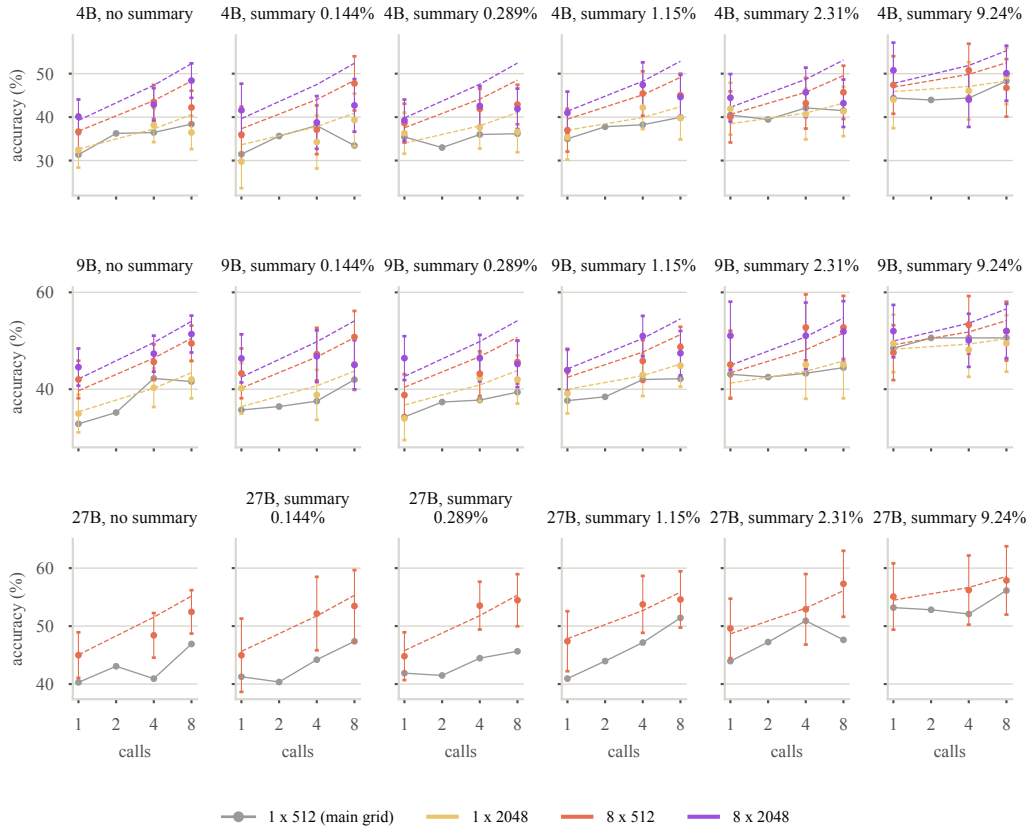

Figure 12: **All calls × retrieval cells** (every summary length with at least 100 questions; rows: 4B, 9B, 27B). Points: observed, 95% intervals; dashed: the law without re-retrieval, fit before these cells were run. The over-prediction at eight calls with eight passages per call appears at every summary length with 2,048-token passages at 4B and 9B, and at most with 512-token ones.

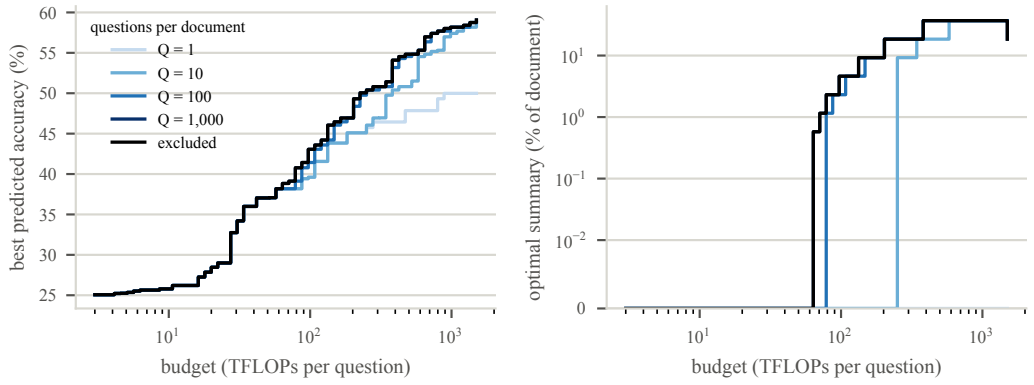

Figure 13: **Charging the summary's one-time cost** at $1/Q$ per question, for Q questions per document (the law, typical document length, observed lever combinations). Left: the best predicted accuracy at each budget (as Figure 4b). Right: the optimal summary length. For a document queried once, a summary never pays; from about a hundred questions per document the best accuracy is within 2 points of the cost-excluded optimum, and identical from a thousand.

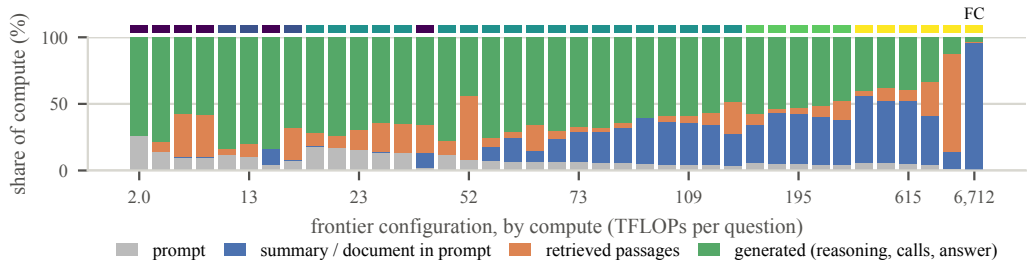

Figure 14: **Where compute goes along the empirical frontier.** Each bar is one configuration on the frontier, ordered by compute (adjusted TFLOPs per question); the strip above gives its model size (colors as in Figure 1; FC: full context). Shares are of each configuration's measured compute. Generated tokens (reasoning, calls and the answer) are the largest share at all but 7 of the 39 points; the summary's share grows with its length, to a quarter to two fifths at 18% of the document (about half at the cheaper 27B points, more than generated tokens), and retrieved passages rarely exceed a quarter.

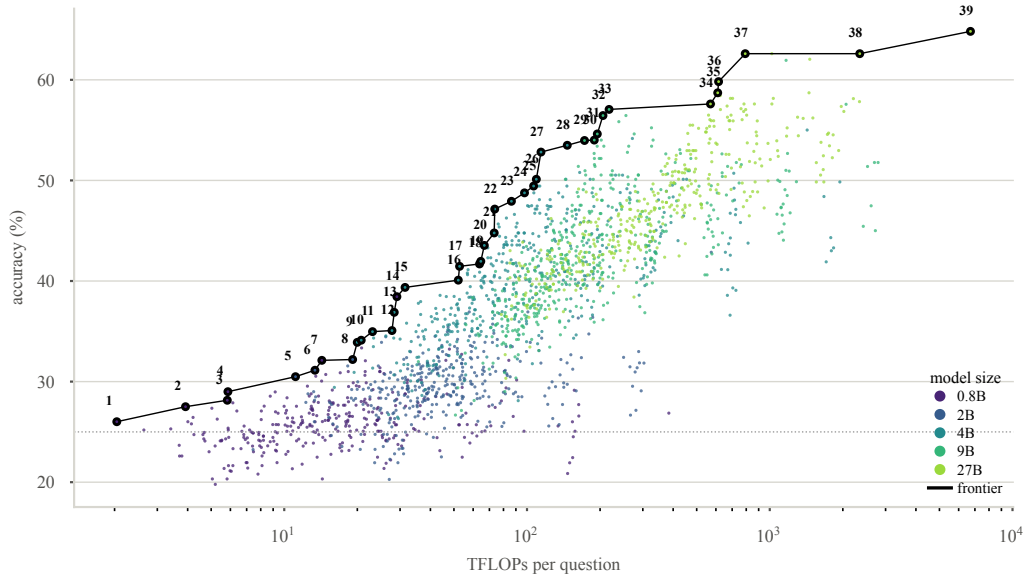

Figure 15: **The empirical frontier**: the best observed configuration at each budget (numbered), over every configuration with at least 100 questions. Its accuracies are biased upward by selection (Section 5).

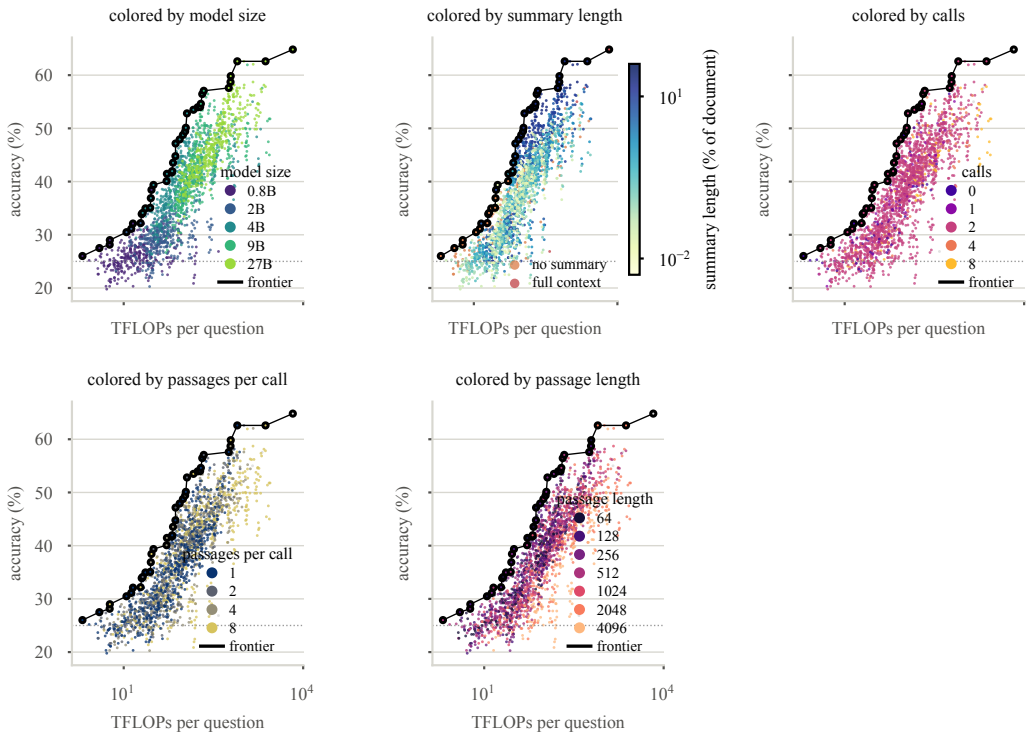

Figure 16: **The same configurations, colored by each lever's value.** Along the empirical frontier: model size grows in steps, the summary grows from none to 18% of the document, calls mostly stay at two, and passages are mostly short.

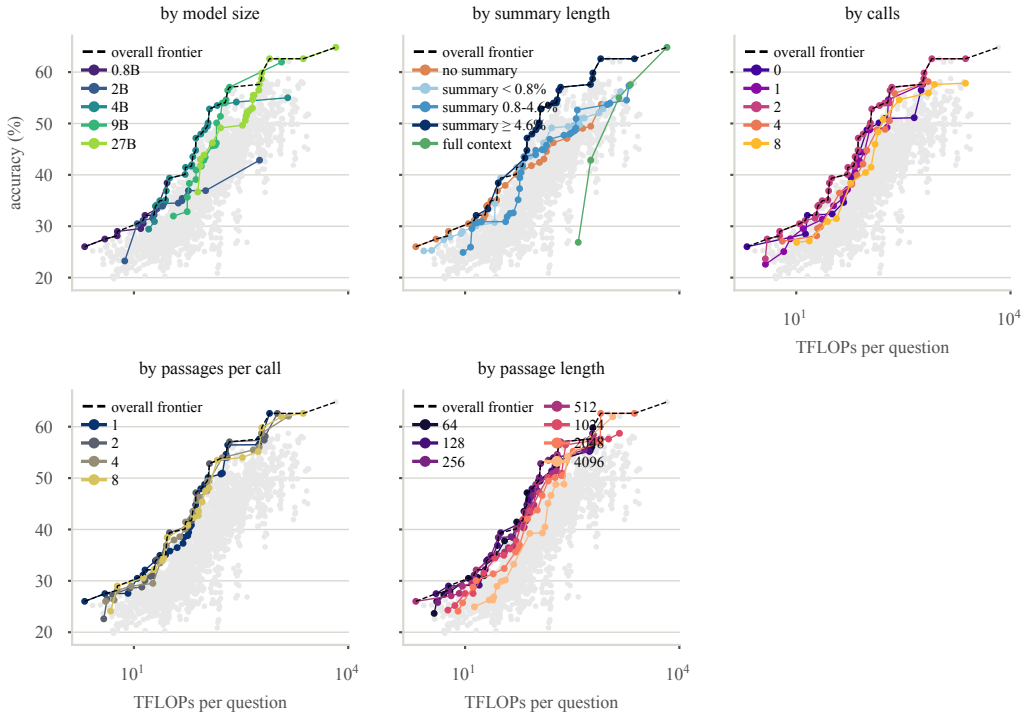

Figure 17: **Constrained empirical frontiers**: the best observed configuration at each budget with one lever held at each of its values. A fixed number of passages per call or passage length costs little (Section 5); no summary, or a model below 4B, costs much more.

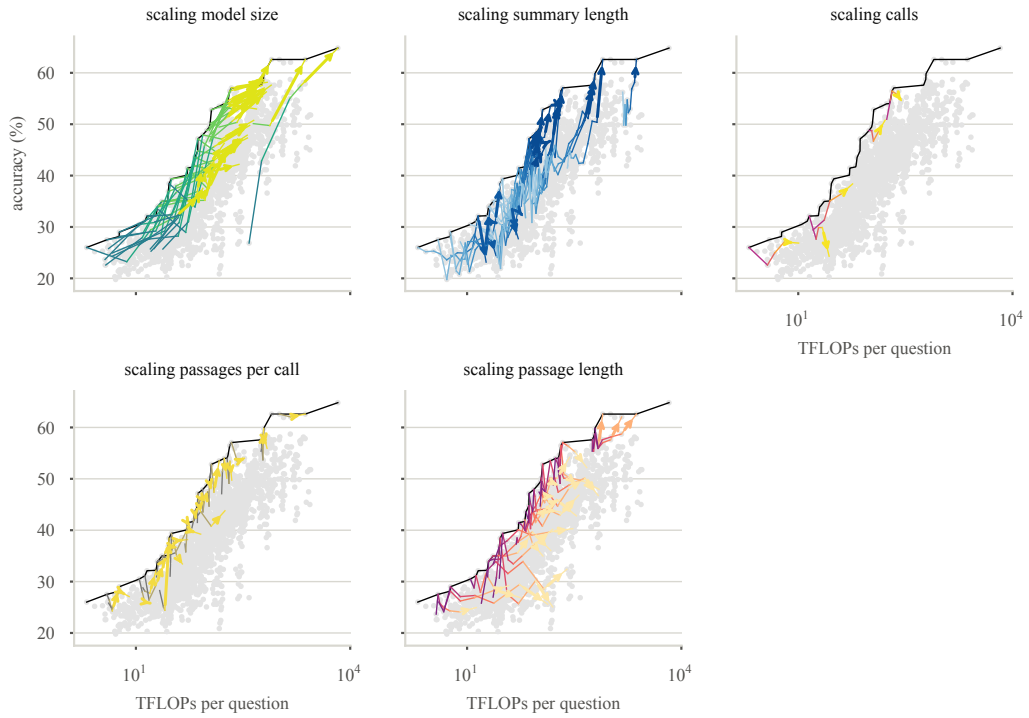

Figure 18: **Scaling one lever from each configuration on the empirical frontier** (arrows from low to high values of the lever, the others fixed). Growing the summary or the model moves configurations towards the frontier; more retrieval mostly moves them right.

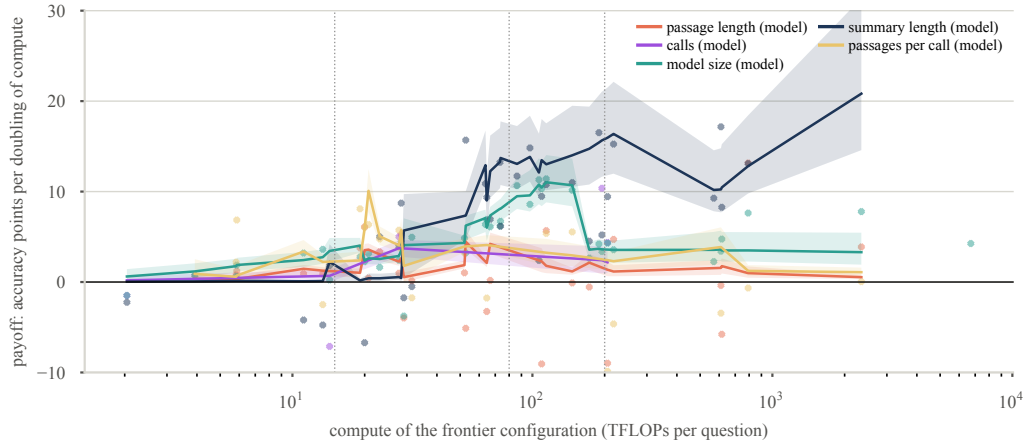

Figure 19: **Which lever to scale next, with the empirical check**: the law's payoffs along the empirical frontier (lines; Figure 4a follows the compute-optimal path instead) and the empirical local payoffs between neighbouring observed configurations (dots), which scatter around the law's and mostly below them, as selection along the empirical frontier predicts.

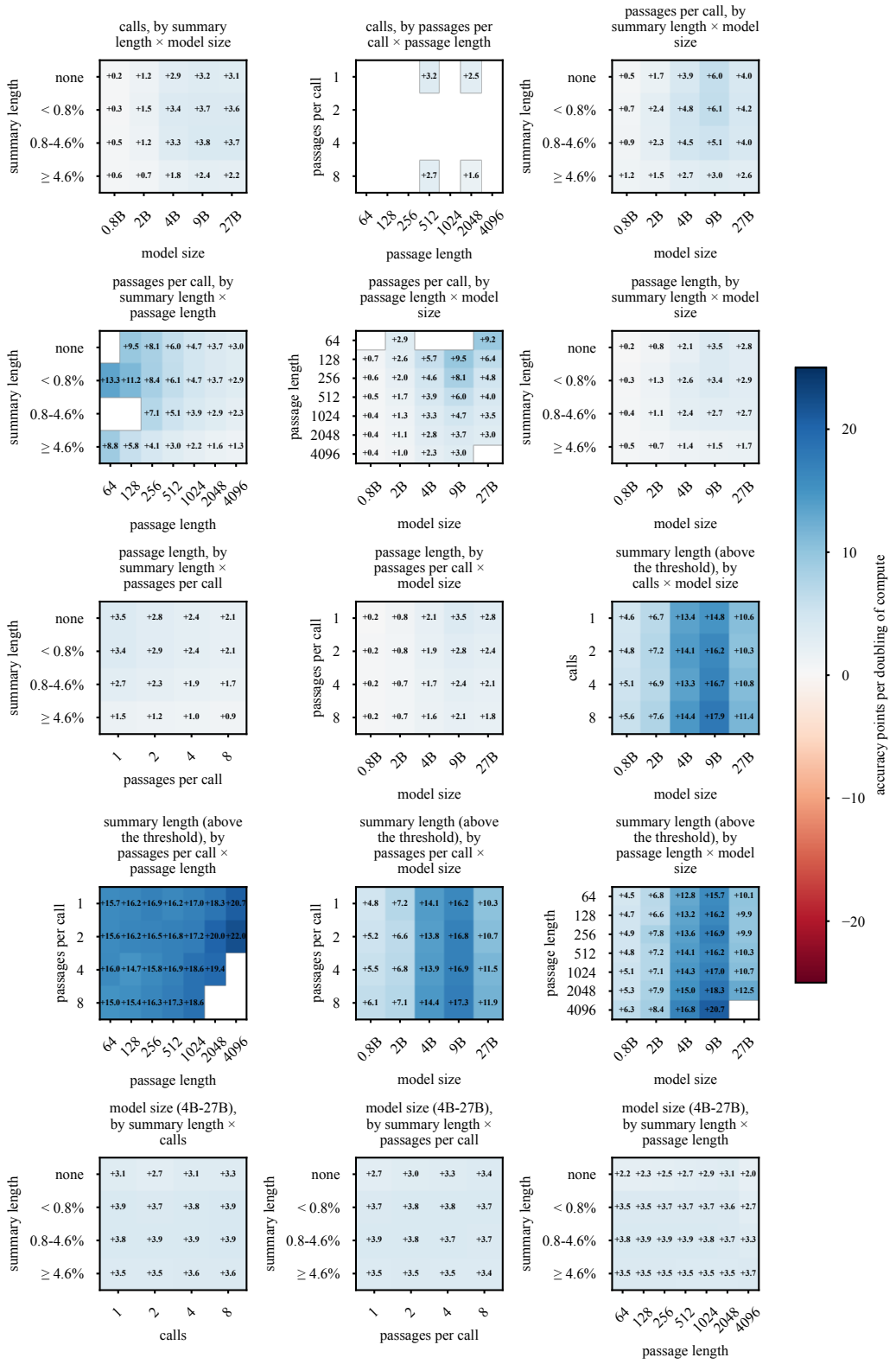

Figure 20: **Payoff of each lever per doubling of compute, over pairs of the other levers** (the law; the rest at 9B, two calls, one 512-token passage, no summary). A longer summary pays most, above the threshold, at every combination; the payoff of retrieval shrinks with summary length.

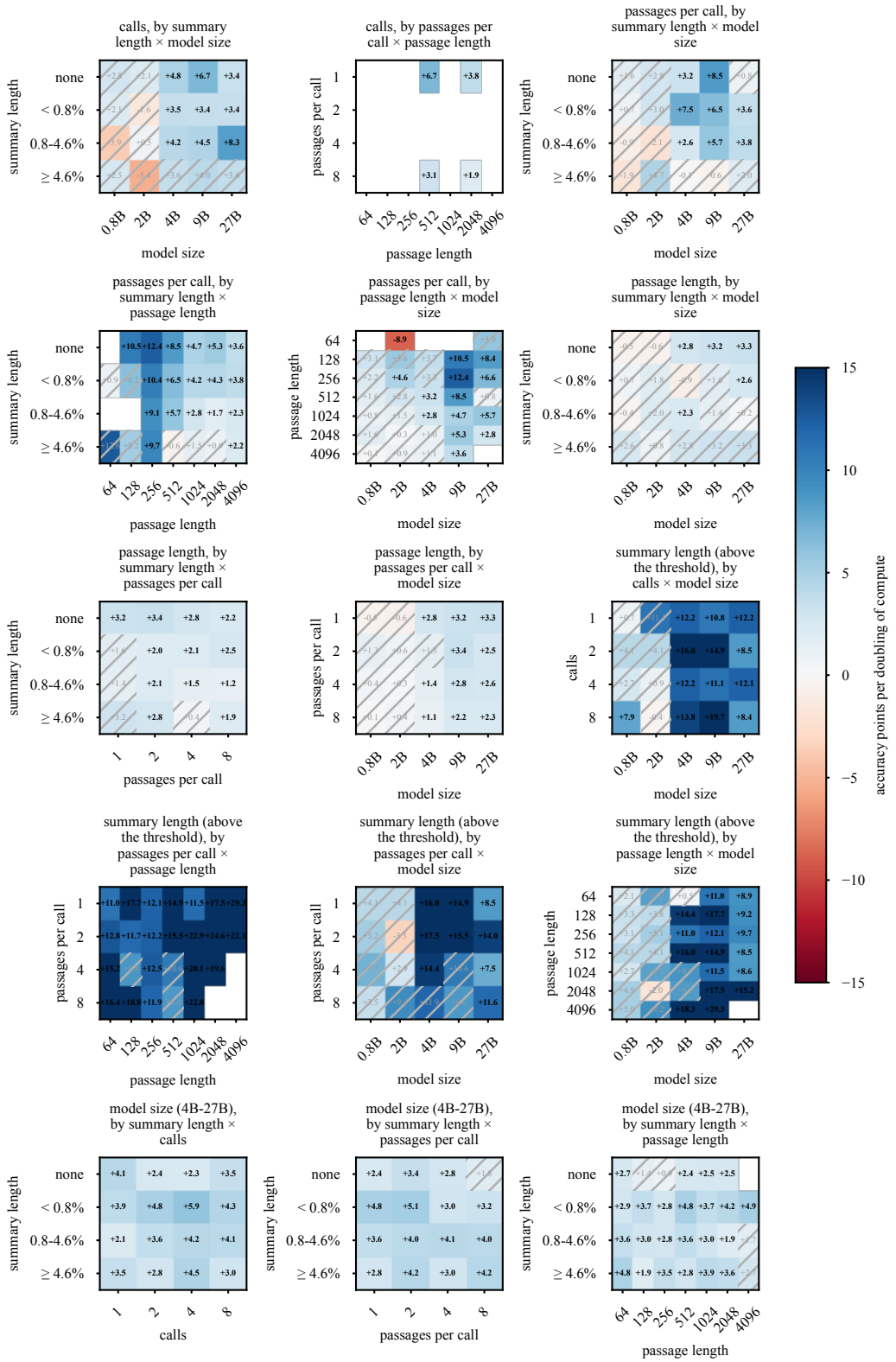

Figure 21: **The same payoffs measured directly** from neighbouring observed configurations (hatched: 95% interval includes zero; blank: not observed).

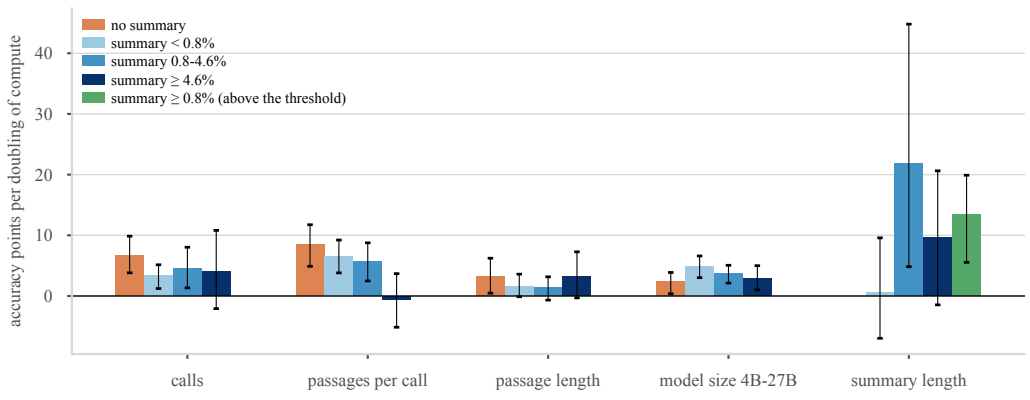

Figure 22: **Measured payoff of each lever at 9B by summary length** (one slice of Figure 21; 95% intervals).

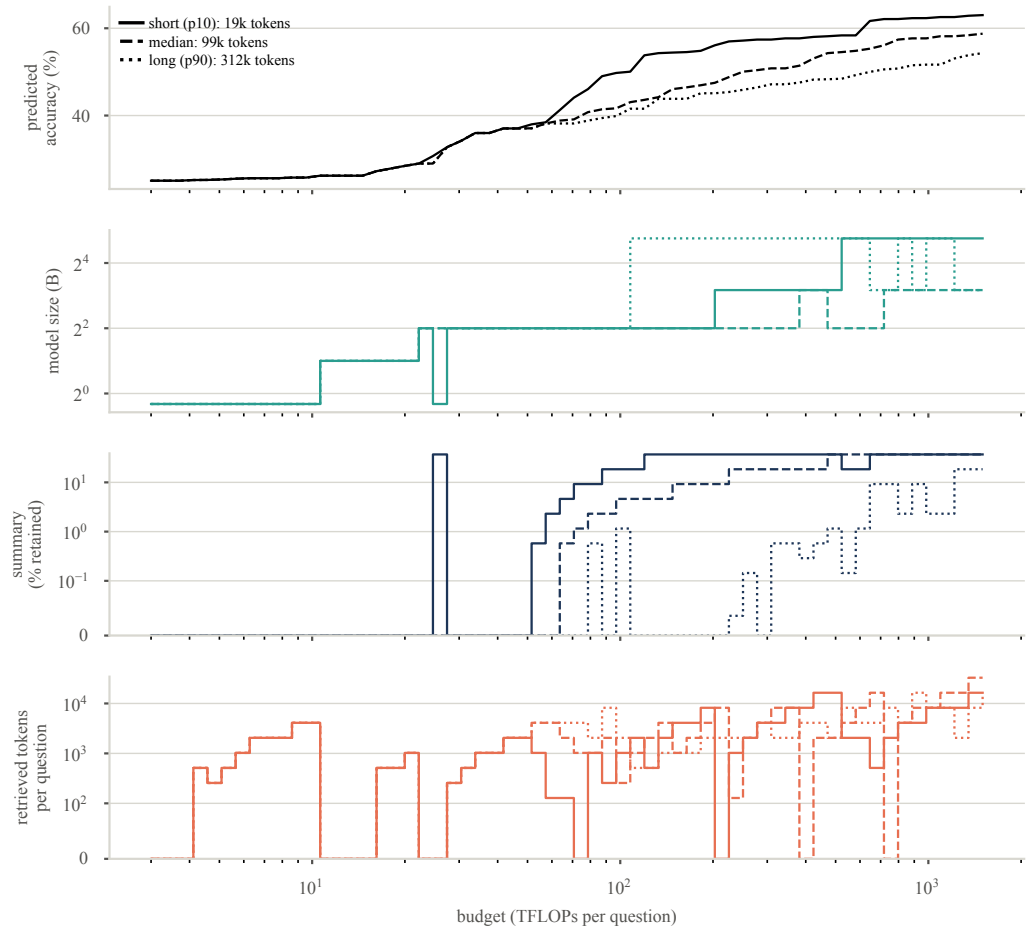

Figure 23: **Compute-optimal allocation for a short, median and long document as the budget grows** (the law, over observed lever combinations; Table 11 gives three budgets). Jagged steps are near-ties between configurations.

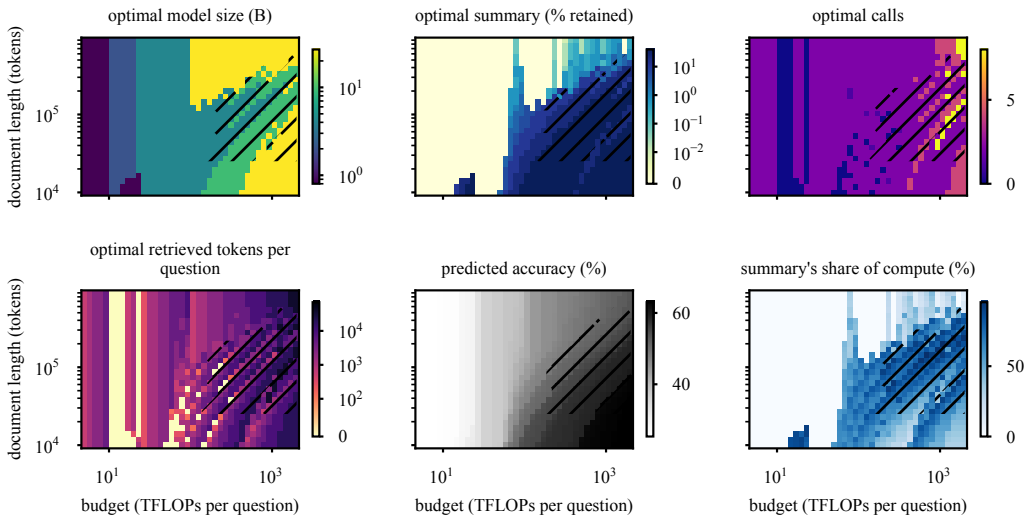

Figure 24: **All six panels of Figure 5**: optimal model size, summary length, calls and retrieved tokens, predicted accuracy, and the summary's share of compute. Hatched: the optimum needs a summary longer than any observed.

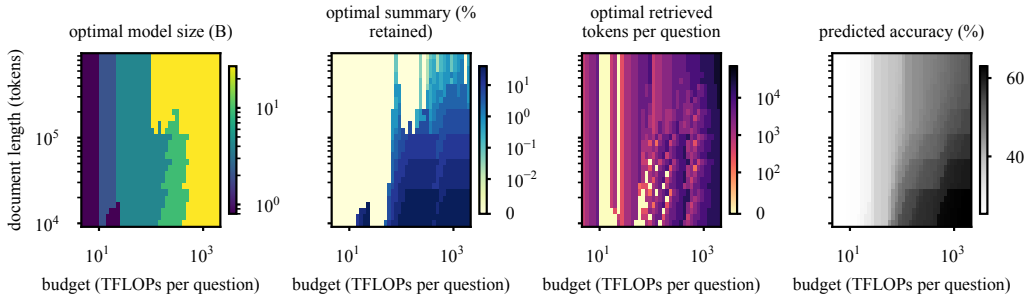

Figure 25: **The same allocation restricted to summaries of at most 10k tokens**, the longest our summarizer wrote: the best allocation with the summaries we observed.

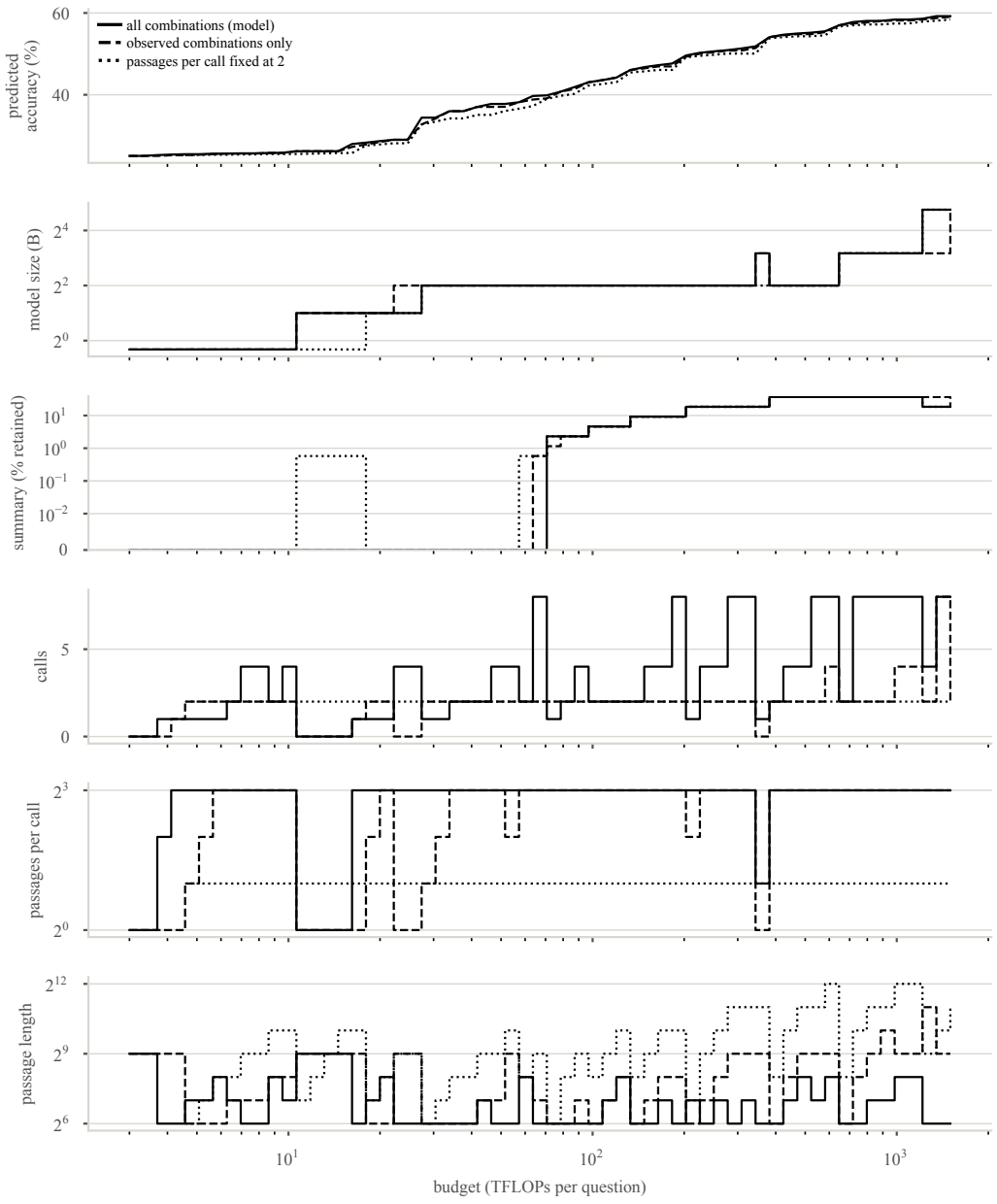

Figure 26: **Allocation over all lever combinations, over observed combinations only, and with two passages per call.** Over all combinations the optimum at the largest budgets uses eight calls of eight short (64- to 256-token) passages, which we never ran; the main text uses observed combinations.