ListQA: A Benchmark for Evaluating List-Formatted Factual Knowledge Retrieval in Large Language Models

Mayank Kulkarni*
Amazon AGI

Rakesh Vaideeswaran*
Amazon AGI

Abstract

Many real-world questions demand not a single fact but an organized collection of them, yet existing factual knowledge benchmarks almost exclusively target single-answer retrieval. We introduce ListQA, a benchmark of 9,045 human-curated, cross-validated questions (audited error rate of 1.7%) that require LLMs to recall multiple facts and compose them into structured lists of 3–10 elements. Questions range from flat lists (*"Which NCAA teams went undefeated between 2000 and 2024?"*) to hierarchical lists with sub-attributes (*"... and what was their record and result?"*), spanning eight categories across 60+ countries. An evergreen design (explicit temporal constraints or historically immutable facts) keeps ground truth valid without periodic updates. For evaluation, we frame element matching as a linear sum assignment problem: optimal bipartite matching paired with LLM-based semantic grading jointly captures factual accuracy, hallucination, and format compliance. We benchmark 34 models (8 families, 42 configurations across standard and thinking-enabled inference) and find: the best standard-inference model scores below 37% LLM-Judge-F1, multi-faceted questions are harder across the board, and providing the source Wikipedia page lifts scores dramatically, confirming that models can organize facts but struggle to retrieve them from parameters alone. Extended analyses cover base vs. instruction-tuned checkpoints and token efficiency. We release the dataset and evaluation code [1].

1 Introduction

Large language models encode substantial factual knowledge in their parameters [1, 2]. When we evaluate this knowledge, we almost always ask for a single fact: *Who directed Oppenheimer? What is the capital of Bhutan? When was penicillin discovered?* Yet many real information needs have not one correct answer but an entire set: a traveler asks which direct routes an airline launched since 2022, a researcher asks which countries ratified a treaty before a given year, a clinician asks which drugs were approved for a condition in a specific window along with their mechanisms of action. Answering such questions requires a model to recall multiple discrete entities that meet specified criteria, verify each against temporal and categorical constraints, and in the more demanding cases, enrich each with structured sub-attributes. In short, the user needs the *complete* set of qualifying facts, without fabricated additions, organized with the requested attributes. Moreover, structure matters: when an answer contains multiple independent facts, organizing them as a list, which is the most natural format for enumeration in language, turns evaluation from a monolithic judgment into a per-element one, enabling users and automated metrics alike to audit each item independently, detect individual hallucinations, and act on specific entries without parsing an undifferentiated paragraph. These properties are not merely convenient but functionally necessary in domains such as

*Equal contribution.

[1] Dataset and evaluation code can be found at `https://listqa.github.io`

40th Conference on Neural Information Processing Systems (NeurIPS 2026). Track on Evaluations and Datasets.

Table 1: Comparison of LISTQA with representative benchmarks across the following dimensions: **ME**: multi-element retrieval. **SO**: structured output. **EG**: evergreen design. **K+IF**: knowledge + instruction following. Shading indicates partial coverage.

Benchmark	ME	SO	EG	K+IF
Single-answer factual QA				
TriviaQA [3]	✗	✗	✗	✗
Natural Questions [4]	✗	✗	✗	✗
PopQA [5]	✗	✗	✗	✗
SimpleQA [6]	✗	✗	✓	✗
FACT-Bench [7]	✗	✗	✗	✗
Multi-answer factual QA				
QAMPARI [8]	✓	✗	✗	✗
RoMQA [9]	✓	✗	✗	✗
Multi-hop & reasoning QA				
HotpotQA [10]	✗	✗	✗	✗
MuSiQue [11]	✗	✗	✗	✗
GPQA [12]	✗	✗	✗	✗
Structured & table QA				
WikiTableQuestions [13]	✓	✗	✗	✗
HybridQA [14]	✗	✗	✗	✗
Temporal & dynamic				
FreshQA [15]	✗	✗	✓	✗
RecencyQA [16]	✗	✗	✓	✗
Long-form factuality				
FActScore [17]	✓	✗	✗	✗
FACTS Grounding [18]	✓	✗	✗	✓
Knowledge calibration				
SelfAware [19]	✗	✗	✗	✗
LISTQA (ours)	✓	✓	✓	✓

clinical decision-making and legal research, where each element may carry independent downstream consequences.

Most factual benchmarks [3–7, 20] currently are not designed to measure this: they ask for a single answer, so they cannot assess how completely a model retrieves all qualifying entities, how precisely it avoids hallucinating extras, or whether it can structure the result coherently. Multi-answer factual QA datasets [8, 9] retrieve multiple entities per question, but evaluate only set overlap, not structured output or format compliance. Furthermore, much of the factual knowledge that underpins list-formatted answers resides not in running text but in structured sources such as tables and infoboxes, where answering a single question may require reasoning across multiple columns, applying filtering constraints, and aggregating rows, a retrieval mode that is fundamentally harder to perform parametrically than locating a fact stated explicitly in a narrative paragraph. Addressing this gap requires navigating several further challenges:

The intersection of factual retrieval and structured output remains uncharted. Significant progress has been made on each capability individually. Multi-hop benchmarks [10–12] and fact reasoning tasks [21–23] have advanced factual reasoning evaluation, though they typically evaluate a single final answer per query; table QA datasets [13, 14] have enabled rich evaluation of comprehension over provided tables. Instruction-following benchmarks [24–27] have improved the measurement of models' ability to satisfy user-specified constraints and formats. Yet real information needs often require models to simultaneously provide accurate factual information while adhering to user-specified constraints, and their interaction is non-trivial: formatting constraints can degrade factual accuracy [18, 28], and populating a multi-element structure may increase the tendency to hallucinate. This interplay has not been systematically measured by a dedicated benchmark.

Temporal drift is an unsolved problem for multi-element benchmarks. Jiang et al. [29] show that 24–64% of time-sensitive samples in popular factual datasets already carry outdated gold answers, silently inflating model scores. All existing mitigations, periodic refresh [15, 30, 31], drift detection [16, 32–36], inherently stable questions [6], assume a single answer per sample. Lists break this assumption: when ground truth contains multiple elements, temporal validity must hold across *every* entry simultaneously, and the probability of silent corruption grows with each element added. To the best of our knowledge, no existing benchmark guarantees this property, leaving multi-element factual evaluation vulnerable to the same decay that has already compromised single-answer benchmarks.

Standard metrics fail on set-based, multi-element outputs. Consider a model that correctly recalls "NE Patriots" when the reference says "New England Patriots", lists elements in a different order, and includes one hallucinated entry. Exact match scores this as entirely wrong; ROUGE [37] computes overlap across the entire response, unable to distinguish which elements are correct and which are fabricated. Evaluating such outputs requires solving two coupled problems: (1) aligning each predicted element to its reference counterpart, a combinatorial assignment problem when element order is irrelevant and the predicted and reference sets may differ in size and, (2) judging semantic equivalence at the element level, where lexical variation (e.g., abbreviations, reorderings of sub-attributes) is the norm. Existing LLM-as-judge [18, 38] and atomic-claim verification [17] approaches

Example: SIMPLELISTQA

Q: Which songs by Bruno Mars were nominated for the MTV Video Music Award for Best Choreography between the years 2010 and 2024, where he was both the performer and the choreographer?

A: The MTV Video Music Award for Best Choreography is a craft award given to the artist, the artist's manager, and choreographer of the music video.
The following songs by Bruno Mars were nominated for the MTV Video Music Award for Best Choreography between the years 2010 and 2024, where he was both the performer and the choreographer:

1. The Lazy Song
2. Treasure
3. Finesse (Remix)

Example: COMPLEXLISTQA

Q: In 2023 and 2024, for which awards was *John Wick: Chapter 4* nominated in Best Stunts, who was the recipient and what was the result?

A: The awards for which *John Wick: Chapter 4* was nominated in Best Stunts are:

1. Hollywood Critics Assoc. Midseason Awards
   - *recipient:* John Wick: Chapter 4
   - *result:* Won
2. Las Vegas Film Critics Society
   - *recipient:* John Wick: Chapter 4
   - *result:* Nominated
3. St. Louis Film Critics Assoc.
   - *recipient:* Scott Rogers & Stephen Dunlevy
   - *result:* Runner-up
4. Astra Film & Creative Arts Awards
   - *recipient:* John Wick: Chapter 4
   - *result:* Won

Figure 1: Examples from the LISTQA dataset.

provide scalable semantic grading but assume a single answer or unconstrained generation, offering no mechanism for element-to-element alignment. Without this alignment step, a judge that receives the full predicted list at once tends to accept it holistically when most elements appear plausible, rather than detecting individual hallucinations or omissions.

To address these gaps, we introduce LISTQA, which, as Tab. 1 summarizes, is to our knowledge the first benchmark to simultaneously require their conjunction: multi-element retrieval, structured output, and joint knowledge and instruction-following evaluation. Our contributions are as follows:

1. We introduce LISTQA, a benchmark of 9,045 human-curated questions that require retrieving multiple facts and composing them into structured lists. The dataset spans two subsets: SIMPLELISTQA (single-faceted flat lists) and COMPLEXLISTQA (multi-faceted hierarchical lists, see Fig. 1) grounded in Wikipedia tables across eight topical categories and 60+ countries. An evergreen design ensures ground truth remains valid without periodic updates, with temporal emphasis on 2017–2024 to probe recall of recent, less-saturated knowledge.
2. We propose an LLM-Judge-F1 metric that **(1)** optimally matches each predicted list element to its closest reference element, and **(2)** uses an LLM to grade correctness, capturing factual accuracy and penalizing hallucination. We validate this metric against human judgments and complement it with four additional metrics covering coverage, precision, format compliance, and lexical overlap.
3. We evaluate 34 models from 8 families across 42 configurations, spanning standard and thinking-enabled inference. The best standard-inference model does not exceed 37% LLM-Judge-F1 score; multi-faceted questions are harder across every model; and thinking universally improves factual accuracy but has architecture and model family-dependent effects on format compliance. An oracle RAG experiment shows that supplying the source page yields large gains for every model, identifying parametric recall as the primary bottleneck. Extended analyses covering base vs. instruction-tuned checkpoints and token efficiency are provided in the appendix.

2 Dataset

2.1 Task Definition

Given a factual question q whose correct answer is a list of n elements $\mathcal{L} = \{l_1, l_2, \ldots, l_n\}$ where $3 \leq n \leq 10$, the task requires a model to generate a structured list response $\hat{\mathcal{L}} = \{\hat{l}_1, \hat{l}_2, \ldots, \hat{l}_m\}$. Each element l_i may be a simple entity (single-faceted) or a composite of a primary entity with associated sub-attributes (multi-faceted). The model is evaluated on the correctness, completeness, and structural formatting of $\hat{\mathcal{L}}$ with respect to $\mathcal{L}$.

2.2 Data Collection

Source material. All questions and answers are grounded in Wikipedia tables, chosen as the reference source for several reasons. First, Wikipedia serves as a common knowledge source across LLM training pipelines, providing a fair evaluation baseline. Second, tabular data naturally lends itself to questions with list-formatted answers, as multiple rows of a table can satisfy a given query criterion. Third, using a single, publicly available source eliminates concerns about proprietary data access and licensing disparities across models.

2.2.1 Annotation Process.

Human annotators were instructed to: (i) identify Wikipedia tables covering popular entities and topics across eight categories: Entertainment (including Music), Sports, Politics, Movies, Lifestyle, Food, Travel, and Places of Interest; (ii) formulate questions whose answers require retrieving information from one or more columns of the table; and (iii) compose reference answers in list format with 3 to 10 elements. The lower bound of 3 avoids degenerate cases that reduce to single-answer QA, while the upper bound of 10 prevents excessively long lists that test exhaustive enumeration rather than targeted knowledge retrieval. Complete annotation guidelines are provided in App. B.

Question types. The dataset comprises two question types, defined based on their complexity: *(1) Single-faceted* questions require retrieving entities that satisfy one or more filtering conditions over the source table. The answer is a flat numbered list of entity names. *(2) Multi-faceted* questions often require reasoning across multiple columns and additionally request structured sub-attributes (e.g., recipient, result) for each retrieved entity. The answer is a hierarchical list whose elements each contain multiple fields. These questions may further incorporate exclusion or negation constraints that require filtering across several dimensions simultaneously.

Temporal constraints. To ensure the dataset remains factually valid over time, annotators were instructed to include explicit time constraints in questions where applicable. Rather than asking *"What are the Flying Star Cafe locations?"* (which may change), questions are framed as *"Which Flying Star Cafe locations opened from 2000 to 2024?"*.

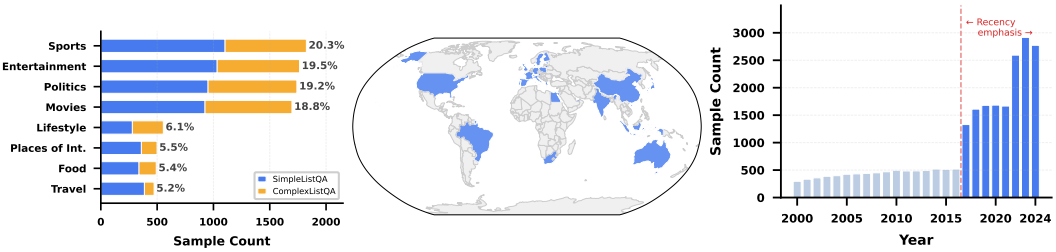
Temporal distribution. The dataset covers years from 2000 through 2024, with deliberate emphasis on 2017–2024 to probe recall of recent, less-saturated knowledge. The year 2024 is especially informative, as publicly available training data for it is substantially more restricted than for earlier years. All answers are validated as factually correct as of December 31, 2024.

Objectivity constraints. All questions must be close-ended and objective. Subjective questions (e.g., *"top 10 vegan restaurants in NYC"*) are excluded, as disambiguating via a specific source would test website knowledge rather than factual recall and unfairly disadvantage models not trained on that source. This ensures unambiguous, source-independent evaluation.

Ordering. We note that not all lists are required to be ordered, for example *"In 2022 what Artists were nominated for the MTV Video Music Award for Best Pop but did not win?"*. However, *"Who were the first 8 presidents of the US Senate, what were their exact positions, and what were their terms?"* is an ordered list, so are chronological and sequential events; annotators are instructed to tag the requirement for ordering.

2.3 Dataset Statistics

The full dataset contains **9,045** question-answer pairs sourced from **3,765** unique Wikipedia tables. Fig. 2 provides a comprehensive overview. The four largest topical categories are Sports, Entertainment, Politics, and Movies; each contributes roughly one-fifth of the dataset, with ratios fairly maintained across both subsets. The dataset covers entities from over 60 countries across six continents, and its temporal distribution is deliberately skewed toward 2017–2024 to test recall of recent knowledge. The majority of Wikipedia tables contribute one or two questions, with a long tail extending to 28 questions per table. Each sample is released with rich metadata including the source Wikipedia URL, topical category, temporal range, question type, and ordering requirement, enabling independent verification of ground truth and fine-grained analysis along any dimension. Additional dataset statistics are provided in App. C.



Property	Value	Property	Value
Total samples	9,045	Elements per answer	3–8
SIMPLELISTQA/COMPLEXLISTQA	59.5% / 40.5%	Mean elements	4.67
Unique Wikipedia tables	3,765	Median elements	4
Year-tagged / Timeless	70.1% / 29.9%	Sub-elements per item (multi)	1–10
Ordered / Unordered	14.6% / 85.4%	Mean sub-elements	8

Figure 2: Overview of the LISTQA dataset. **Top left:** Distribution across 8 topical categories, split by SIMPLELISTQA and COMPLEXLISTQA. **Top center:** Geographic coverage spanning 60+ countries across 6 continents. **Top right:** Temporal distribution; bars after the dashed line (2017–2024) reflect the deliberate recency emphasis. **Bottom:** Key dataset statistics.

2.4 Data Quality

We employed a four-stage quality assurance process. (1) Two pilot rounds calibrated annotator understanding (pilot data excluded from final dataset). (2) Each question–answer pair was then authored by one out of 44 annotators and independently validated by a second annotator with no involvement in its creation, achieving 100% coverage. (3) Validators flagged issues according to the taxonomy in Tab. 4 (App. B.1); annotations requiring significant rework were discarded (10% discard rate), and the remainder were corrected for a final review pass. (4) All corrections were verified by at least one additional reviewer, with two reviewers covering 20% of corrections. In total, every sample was assessed by at least three independent people.

Reliability estimation. Standard inter-annotator agreement metrics such as Cohen's κ assume that multiple annotators independently label the same item, making them inapplicable to constructive annotation tasks where each annotator generates a unique question–answer pair rather than classifying a shared input. We therefore quantify annotation reliability through cross-validation outcomes, which serve as a functionally equivalent quality signal. Of all annotations submitted to cross-validation, 39% were approved without any flags, 51% required minor corrections (predominantly typographical or phrasing issues), and 10% were discarded as requiring substantial rework. Among the error types identified in flagged annotations, factual accuracy errors constituted only 10% of all flags, while the remainder were surface-level issues (typos, vague phrasing, formatting). To measure the residual error rate directly, we audited 481 questions stratified across all 8 categories and both question types, verifying each reference answer against its source Wikipedia table. We found errors in 8 questions (1.7%; 95% Wilson CI [0.8%, 3.2%]), mostly caused by ambiguous question framing rather than factual mistakes (App. B.1).

3 Evaluation Framework

We design five complementary metrics, each computed per sample and averaged over the dataset. Drawing on the notion of task demands [39], they span demand levels, from format-independent Relaxed-Recall/Relaxed-Precision (reduced) to structure-dependent Format-Correctness and LLM-Judge-F1 (greater).

3.1 Metrics

3.1.1 List Element Parser

Our consistent formatting protocol enables a regex parser that extracts individual list elements from both reference answers and predicted answers, avoiding coarse-grained full-list comparison in a single evaluation pass. Details on the regex pattern are provided in App. D.

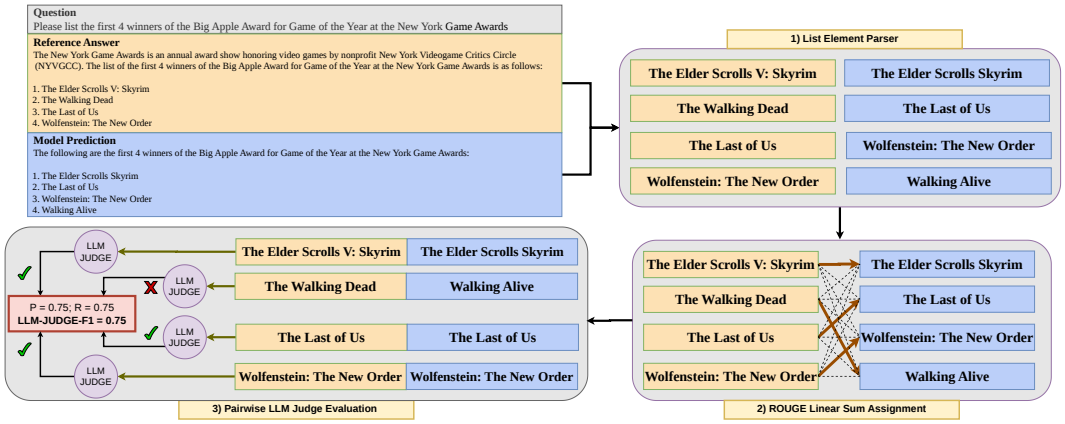


Figure 3: Overview of the LLM-Judge-F1 evaluation pipeline. Given a reference list and model prediction, **(1)** individual list elements are extracted via regex parsing, **(2)** pairwise ROUGE-1 scores form a cost matrix for optimal one-to-one element assignment via the Jonker-Volgenant algorithm, and **(3)** each matched pair is independently graded by an LLM judge as CORRECT or INCORRECT. Unmatched predicted elements are automatically graded INCORRECT. Precision and recall are computed over elements judged CORRECT.

3.1.2 Recall-focused Knowledge Metrics

We design a metric to judge how accurately the model is able to retrieve and generate factual information decoupled from the formatting capabilities. This is done by first leveraging the parser from Sec. 3.1.1 to extract each list element from the reference answer and then testing for the presence of each element independently in the unparsed model output as simple string matching. This judgment is made irrespective of whether the model prediction is formatted correctly in the list format. We compute both Precision and Recall here as *Relaxed-Precision* and *Relaxed-Recall*.

3.1.3 List-Formatting Focused Metrics

We use the parser from Sec. 3.1.1 to determine whether the model's prediction abides by the list formatting expected; we choose not to evaluate the actual content of the list elements as that is covered by the Recall-focused Knowledge metrics from Sec. 3.1.2. The model is provided with a prompt to produce different styles of lists based on the constraints presented in the question and we expect the model to be able to reason on which list style to produce that aligns with the formatting styles of single and multi-faceted depending on the question. Subsequently, the model is evaluated on that type of list format for correctness: producing a flat list for a multi-faceted question or vice versa would be erroneous. Importantly, we require the model to produce at least 3 list elements for the output to be considered correct under this metric. This threshold mirrors the annotation constraint that all reference answers contain 3–10 elements (Sec. 2.2), ensuring that models cannot achieve high format scores with trivially short outputs that sidestep the multi-element retrieval challenge. We term this the *Format-Correctness* metric.

3.1.4 Precision-focused Metric (Format + Knowledge)

This metric jointly evaluates factual correctness and list formatting. Using the parser from Sec. 3.1.1, we independently extract individual list elements from both the reference answer and the model prediction, yielding two element lists. Rather than judging the predicted list holistically, which we find correlates poorly with human evaluation, we assess each predicted element against its reference counterpart using an LLM as a judge.

A naïve all-pairs comparison requires $\mathcal{O}(n^2)$ LLM judge calls, which is prohibitively expensive. We therefore reframe element matching as a *linear sum assignment* problem: we construct a bipartite graph between reference and predicted elements, weight edges by pairwise ROUGE overlap, and solve for the optimal one-to-one assignment using the Jonker-Volgenant algorithm [40] as implemented in SciPy [41, 42]. Although the assignment itself is $\mathcal{O}(n^3)$, it reduces LLM judge calls from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$, which is a substantial saving in practice.

Each matched pair is then independently scored by an LLM judge, GPT-OSS-120B, in all experiments reported in this paper (see Sec. 3.1.6 for selection rationale and robustness analysis across judge model sizes). We aggregate per-pair precision and recall into a final F1 score, which we term *LLM-Judge-F1*. A visual representation of this workflow is shown in Fig. 3. As a lightweight alternative, we also report ROUGE F1 computed directly over the matched pairs, which serves as a cheaper but less discriminative proxy for early-stage evaluation. We call this *ROUGE-F1*. A full scoring example is provided in App. E.

3.1.5 Ordering Metrics

Given that we must honor the ordering of elements in the reference wherever applicable, we implement this metric by replacing the linear sum assignment algorithm with positional pairing: list elements are paired in the order the model predicted them and then graded by an LLM as a Judge. We call this metric *LLM-Judge-F1-Ordered* and evaluate only a subset of LISTQA samples with this metric, specifically those requiring implicit or explicit list sequencing.

3.1.6 Correlation with Human Judgement

Judge model selection. All experiments in this paper use GPT-OSS-120B as the default LLM judge. We chose GPT-OSS-120B for two reasons: (1) as an open-weight model, it ensures full reproducibility without dependence on proprietary API access, and (2) it is large enough to perform reliable semantic grading of factual equivalence across diverse topics. The judge prompt builds on the grading template from SimpleQA [6], extended to handle list-structured, multi-element responses.

Correlation with human judgment. We validated LLM-Judge-F1 against 3,313 human-graded list elements from 571 sample–model pairs (roughly 200 stratified questions, 3 models). LLM-Judge-F1 correlates strongly with human F1 (Pearson $r = 0.965$; Spearman $\rho = 0.956$) (see App. R.1). Agreement is consistent across models, categories, and question types, and the metric is slightly conservative relative to human judgment. The ROUGE-based matching step aligns elements correctly 99.7% of the time, and replacing it with embedding-based matching changes aggregate scores by less than 0.002 (App. R.2).

Robustness to judge model size. To assess whether practitioners can substitute smaller, more affordable judges, we rescored all 9,045 samples from a single evaluated model using three additional reasoning-based open-weight judges: Qwen 3.5 27B, 9B, and 4B. All pairwise Spearman rank correlations exceed $\rho = 0.96$, and mean absolute score differences are ≤ 0.024 across all judge pairs (App. R.3). Even the 4B judge produces near-identical sample rankings to the 120B default ($\rho = 0.972$, MAD = 0.022), confirming that the metric is robust to judge scale and that lighter-weight alternatives are viable with minimal impact on conclusions.

3.2 Evaluation Protocol

Instruction prompt. Each question is preceded by a system instruction (prompt templates are provided in App. F) that specifies the expected output format.

Evaluation modes. The benchmark supports two prompting modes: **zero-shot**, where only the instruction and question are provided, and **few-shot**, where 5 in-context examples are prepended. For few-shot evaluation, examples are drawn from the corresponding question type: single-faceted examples for SIMPLELISTQA, multi-faceted for COMPLEXLISTQA, and a mix for the full dataset.

Dataset variants. Evaluation can be run on three splits: `full` (all 9,045 samples), `simple` (single-faceted only), and `complex` (multi-faceted only).

4 Experiments

4.1 Experimental Setup

We evaluate 34 models spanning 8 model families across 42 configurations, covering closed-source and open-source models and parameter counts from 3B to 675B. All models are evaluated using the zero-shot instruction prompt described in App. F.1 on the full LISTQA dataset, with scores reported overall and separately for the SIMPLELISTQA and COMPLEXLISTQA subsets. Unless otherwise

Table 2: Results on LISTQA under standard (non-thinking) inference. Models are grouped by family and sorted by overall *LLM-Judge-F1* within each group. **LLM-Judge-F1**: LLM-as-judge semantic F1 measuring factual accuracy. **Format**: Format-Correctness score. S = SIMPLELISTQA, C = COMPLEXLISTQA. Per-sample distribution details are discussed in App. S.5.

Closed-source models

Model	LLM-Judge-F1 ↑			Format ↑		
	All	S	C	All	S	C
GPT [43]						
GPT-5.4	36.5	44.5	24.9	87.2	85.0	90.6
GPT-5.4 mini	27.0	35.2	15.1	64.1	84.8	33.6
GPT-5.4 nano	14.5	20.4	5.7	58.1	77.7	29.2
Claude [44–49]						
Claude Opus 4.7	35.0	42.1	24.6	75.0	75.4	74.5
Claude Opus 4.6	34.6	43.0	22.4	78.3	73.9	84.8
Claude Opus 4.5	33.9	42.0	22.2	70.8	63.0	82.1
Claude Sonnet 4.6	28.0	35.5	16.9	60.2	45.3	82.2
Claude Sonnet 4.5	25.3	30.2	18.1	49.2	41.1	61.0
Claude Haiku 4.5	11.1	14.1	6.7	49.1	58.6	35.2
Gemini [50]						
Gemini 2.5 Flash	33.7	41.9	21.6	76.4	72.5	82.2
Gemini 2.5 Flash Lite	20.4	27.2	10.4	63.9	83.6	35.1
Nova [51, 52]						
Nova Premier	31.6	39.0	20.6	60.7	60.6	61.0
Nova Pro	22.0	29.3	11.4	65.6	53.8	83.0
Nova Micro	12.0	17.5	4.0	56.8	79.1	24.1

Open-source models

Model	LLM-Judge-F1 ↑			Format ↑		
	All	S	C	All	S	C
DeepSeek [53, 54]						
DeepSeek V3.1	29.3	36.7	18.3	73.3	76.0	69.4
DeepSeek V3.2	20.3	26.5	11.1	54.8	71.9	29.8
Qwen [55–57]						
Qwen 3.5 397B-A17B (FP8)	24.8	31.6	14.8	73.0	69.0	78.8
Qwen 3.5 122B-A10B	18.9	25.3	9.7	82.2	84.1	79.5
Qwen 3 Next 80B-A3B	17.4	23.6	8.3	39.1	7.8	85.1
Qwen 3.5 27B	14.8	20.1	7.1	71.9	71.6	72.4
Qwen 3.5 35B-A3B	13.4	18.5	5.9	51.7	37.0	73.2
Qwen 3.5 9B	8.3	12.0	2.9	62.0	65.5	56.8
Qwen 3.5 4B	5.1	7.8	1.3	39.2	43.9	32.3
Mistral [58–60]						
Mistral Large 3 675B-A41B	25.7	33.2	14.8	58.9	77.9	31.0
Magistral Small 24B	15.8	21.8	7.0	59.6	48.0	76.8
Ministral 3 14B	7.2	10.4	2.6	46.9	76.2	3.8
Ministral 3 8B	4.8	6.9	1.7	45.7	71.0	8.6
Ministral 3 3B	3.3	5.1	0.5	46.2	76.3	2.1
Gemma [61]						
Gemma 4 31B	16.0	22.0	7.3	54.3	45.8	66.7
Gemma 4 26B-A4B	13.8	19.0	6.0	52.5	44.2	64.6
Gemma 4 E4B	6.2	9.1	1.9	33.6	44.9	17.0

noted, all evaluations use standard (non-thinking) inference. All results use GPT-OSS-120B as the LLM judge. Subsets of models are additionally evaluated with thinking enabled (Sec. 4.2) and with the source Wikipedia page supplied as context (oracle RAG; Sec. 4.3).

Metric selection. We report two headline metrics: *LLM-Judge-F1* and *Format-Correctness*. These two metrics capture the orthogonal capabilities that LISTQA is specifically designed to disentangle: factual knowledge retrieval (*LLM-Judge-F1*) and instruction-following compliance (*Format-Correctness*). LLM-Judge-F1 subsumes ROUGE-F1, which serves as a cheaper but less discriminative proxy for the same dimension, and captures semantic equivalences that Relaxed-Recall and Relaxed-Precision miss due to their reliance on string matching. Full results across all five metrics are provided in App. G. Details on the evaluation infrastructure, including API providers and self-hosted model configurations, are provided in App. H.

4.2 Results & Analysis

Tab. 2 presents results for all models evaluated under standard (non-thinking) inference, grouped by model family. We report *LLM-Judge-F1* and *Format-Correctness* scores for SIMPLELISTQA (S), COMPLEXLISTQA (C), and the full LISTQA dataset (All).

The Knowledge Bottleneck. Multi-element factual retrieval remains **largely unsolved: the best standard-inference model scores below 37% LLM-Judge-F1**, with the top tier (GPT-5.4, Claude Opus variants, Gemini 2.5 Flash, Nova Premier) clustering in a narrow band that suggests a shared ceiling on parametric retrieval; the majority of models fall below 20%. **Multi-faceted questions are consistently and substantially harder**: every model scores higher on SIMPLELISTQA than COMPLEXLISTQA, with the gap widening for stronger models and the weakest collapsing to near-zero on Complex, confirming that compounding knowledge demands (reasoning across multiple dimensions, producing hierarchical sub-attributes, applying filtering constraints) amplify the bottleneck rather than merely adding formatting difficulty. Thinking-enabled inference (Fig. 4) reinforces this diagnosis: **thinking yields larger gains on SIMPLELISTQA than COMPLEXLISTQA** (Fig. 4b), indicating it helps models organize knowledge they already possess but provides diminishing returns when parametric recall is the limiting factor. Gemini 3.1 Pro achieves the strongest overall result and is the only model to cross 50% LLM-Judge-F1 on COMPLEXLISTQA (App. K), though only barely,

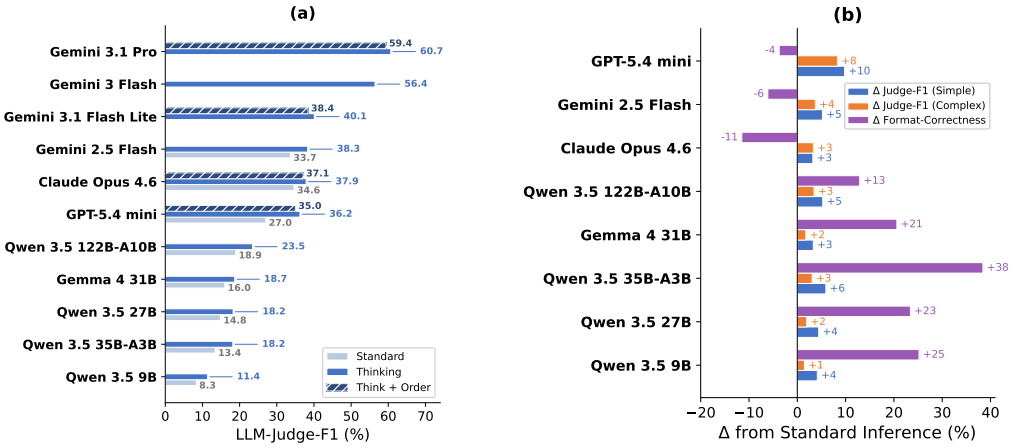


Figure 4: (a) *LLM-Judge-F1* scores across inference settings. Hatched bars show *LLM-Judge-F1-Ordered*, a stricter metric penalizing incorrect sequencing for select samples (details in App. J). (b) Improvement from thinking over standard inference. Thinking universally improves factual accuracy, but its effect on Format-Correctness may be architecture and mode family-dependent. Exact scores and thinking levels are reported in App. K.

underscoring multi-faceted retrieval's difficulty even for the strongest models. **Supplying the source page (oracle RAG) yields large gains for every model**, confirming parametric retrieval as the primary bottleneck (Sec. 4.3). Additionally, **every model performs better on timeless than time-constrained questions** (App. M), confirming that knowledge more heavily reinforced in training corpora is easier to retrieve parametrically. **Category-level difficulty** is model-invariant and intrinsic to training corpora distribution (App. O). **A consistent geographic hierarchy** is observed (strongest on South American/North American content, weakest on African/Asian content), likely reflecting uneven reinforcement of regional knowledge during pretraining (App. P). Further, **list length** itself has only a **modest effect** (App. N), and instruction tuning improves LLM-Judge-F1 with thinking (App. Q).

The Knowledge–Instruction Tension. Orthogonal to the knowledge bottleneck, **factual retrieval and format compliance are partially decoupled**: several models achieve high Format-Correctness while scoring low on LLM-Judge-F1, following structural instructions without the parametric knowledge to populate them, while others with moderate factual accuracy produce relatively poorly formatted output. This decoupling manifests as a **striking format asymmetry between question types**: many models score higher on ComplexListQA than SimpleListQA formatting, not because hierarchy is easier, but because they default to sub-bullet formatting unconditionally, inflating Complex scores while failing the flat-list requirement; the dominant error is producing the wrong list structure (App. I). Thinking sharpens this tension along model family lines: **closed-source models slightly lose format compliance while open-weight models gain it dramatically** (Fig. 4b), suggesting that extended reasoning can either refine or disrupt structural output depending on the model's baseline strategy. The stricter positional metric LLM-Judge-F1-Ordered further isolates sequencing as a distinct error source separable from both retrieval and formatting (App. J), and generation verbosity does not correlate with factual accuracy (App. S). Qualitative examples illustrating these failure modes are provided in App. T. The strongest models are those that jointly retrieve sufficient parametric knowledge *and* adapt output structure to question complexity, **a frontier capability that no current model fully achieves**.

4.3 Oracle RAG: Separating Knowledge from Composition

Closed-book failures on ListQA conflate two error sources: not knowing the qualifying facts, and failing to compose known facts into the requested structure. To separate them, we simulate perfect retrieval by supplying the source Wikipedia page used during annotation as context, keeping the list-formatting instructions and metrics unchanged (setup in App. L), for a subset of closed-source and open-weight models.

Tab. 3 yields three takeaways. **Knowledge is the dominant bottleneck:** oracle context brings large gains for every model, far exceeding those from thinking (Fig. 4b), so inference-time reasoning

Table 3: Oracle RAG vs. parametric-only LLM-Judge-F1. All = Overall, S = SIMPLELISTQA, C = COMPLEXLISTQA. **Context / Src. budget**: context window (tokens) and maximum number of source characters provided to open-weight models (App. L); closed-source models received the full page.

			Parametric			Oracle RAG			
Model	**Context**	**Src. budget**	**All**	**S**	**C**	**All**	**S**	**C**	**ΔAll**
Closed-source									
Claude Opus 4.6		Full	34.6	43.0	22.4	84.7	86.0	82.8	+50.1
Claude Sonnet 4.6		Full	28.0	35.5	16.9	83.0	85.0	79.9	+55.0
GPT-5.4 mini		Full	27.0	35.2	15.1	75.8	79.8	70.0	+48.8
Gemini 2.5 Flash Lite		Full	20.4	27.2	10.4	70.6	74.8	64.3	+50.2
Open-weight									
Gemma 4 31B	256K	800K	16.0	22.0	7.3	80.5	83.2	76.6	+64.5
Gemma 4 26B-A4B	256K	800K	13.8	19.0	6.0	73.3	76.8	68.1	+59.5
Magistral Small 24B	128K	400K	15.8	21.8	7.0	63.5	67.8	57.3	+47.7
Ministral 3 8B	128K	400K	4.8	6.9	1.7	47.7	52.2	41.1	+42.9
Ministral 3 3B	128K	400K	3.3	5.1	0.5	37.6	43.7	28.7	+34.3

cannot substitute for missing knowledge. **Closed-book scores reflect parametric knowledge, not reading ability:** model rankings reshuffle once the source page is provided; for example, Gemma 4 31B trails GPT-5.4 mini and Gemini 2.5 Flash Lite closed-book but overtakes both with context. **Composition is the residual bottleneck, especially at small scale:** oracle RAG does not saturate the benchmark, and with the source page in context the Simple–Complex gap becomes small for the strongest models but grows as models get smaller, even among models with identical context budgets. For frontier models, the multi-faceted penalty is thus largely a knowledge penalty; smaller models additionally struggle to reason across table dimensions and extract hierarchical sub-attributes even when the facts are in context.

5 Conclusion

LISTQA reframes factual evaluation as a multi-element problem: not whether a model can retrieve one fact, but whether it can recall, aggregate, and structurally organize many. Its evergreen design guarantees temporal validity at the list level, and its two-subset structure cleanly separates simple vs complex questions, and the provided metrics help separate knowledge retrieval from instruction following. The picture that emerges from 34 models across 42 configurations is stark. Parametric knowledge is the bottleneck: the best model (standard inference) scores below 37%, oracle RAG adds 34–65 points, and reasoning provides diminishing returns where knowledge is absent. Format compliance is the second axis: models trade off retrieval accuracy against structural correctness rather than jointly optimizing both. No current model masters both dimensions simultaneously. Multi-element factual composition underlies summarization, report generation, and structured information retrieval, and LISTQA demonstrates that current models have substantial room for improvement. We release the dataset and evaluation code.

Limitations. LISTQA is English-only and US-centric (78% of samples), does not reward abstention, incurs LLM-judge inference cost, and verifies reference completeness only against a single Wikipedia table; we discuss these limitations in detail in App. A.

Acknowledgments and Disclosure of Funding

We thank all our colleagues and mentors at Amazon AGI for their support and internal feedback in the building of this benchmark including Yang Liu, Jianhua Lu, Shankar Ananthakrishnan, Arindam Mandal, Kang Min Yoo, He Xie and Anna Rumshisky.

References

[1] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 2463–2473, 2019.

[2] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 5418–5426, 2020.

[3] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, 2017.

[4] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[5] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 9802–9822, 2023.

[6] Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024.

[7] Jiaqing Yuan, Lin Pan, Chung-Wei Hang, Jiang Guo, Jiarong Jiang, Bonan Min, Patrick Ng, and Zhiguo Wang. Towards a holistic evaluation of llms on factual knowledge recall. *arXiv preprint arXiv:2404.16164*, 2024.

[8] Samuel Amouyal, Tomer Wolfson, Ohad Rubin, Ori Yoran, Jonathan Herzig, and Jonathan Berant. Qampari: A benchmark for open-domain questions with many answers. In *Proceedings of the third workshop on natural language generation, evaluation, and metrics (GEM)*, pages 97–110, 2023.

[9] Victor Zhong, Weijia Shi, Wen-tau Yih, and Luke Zettlemoyer. Romqa: A benchmark for robust, multi-evidence, multi-answer question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7055–7067, 2023.

[10] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380, 2018.

[11] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[12] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*, 2024.

[13] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, 2015.

[14] Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, 2020.

[15] Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. Freshllms: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720, 2024.

[16] Bhawna Piryani, Zehra Mert, and Adam Jatowt. How often do answers change? estimating recency requirements in question answering. *arXiv preprint arXiv:2603.16544*, 2026.

[17] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, 2023.

[18] Alon Jacovi, Andrew Wang, Chris Alberti, Connie Tao, Jon Lipovetz, Kate Olszewska, Lukas Haas, Michelle Liu, Nate Keating, Adam Bloniarz, et al. The facts grounding leaderboard: Benchmarking llms' ability to ground responses to long-form input. *arXiv preprint arXiv:2501.03200*, 2025.

[19] Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuan-Jing Huang. Do large language models know what they don't know? In *Findings of the association for Computational Linguistics: ACL 2023*, pages 8653–8665, 2023.

[20] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 2383–2392, 2016.

[21] Sirui Huang, Yanggan Gu, Xuming Hu, Zhonghao Li, Qing Li, and Guandong Xu. Reasoning factual knowledge in structured data with large language models. *arXiv preprint arXiv:2408.12188*, 2024.

[22] Tianshi Zheng, Jiaxin Bai, Yicheng Wang, Tianqing Fang, Yue Guo, Yauwai Yim, and Yangqiu Song. Clr-fact: Evaluating the complex logical reasoning capability of large language models over factual knowledge. *arXiv preprint arXiv:2407.20564*, 2024.

[23] Amey Hengle, Prasoon Bajpai, Soham Dan, and Tanmoy Chakraborty. Can llms reason over extended multilingual contexts? towards long-context evaluation beyond retrieval over haystacks. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6128–6152, 2026.

[24] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[25] Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. Infobench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13025–13048, 2024.

[26] Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. Followbench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4667–4688, 2024.

[27] Gili Lior, Asaf Yehudai, Ariel Gera, and Liat Ein-Dor. Wildifeval: Instruction following in the wild. *arXiv preprint arXiv:2503.06573*, 2025.

[28] Rudra Murthy, Prince Kumar, Praveen Venkateswaran, and Danish Contractor. Evaluating the instruction-following abilities of language models using knowledge tasks. *arXiv preprint arXiv:2410.12972*, 2024.

[29] Xunyi Jiang, Dingyi Chang, Julian McAuley, and Xin Xu. When benchmarks age: Temporal misalignment through large language model factuality evaluation. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 500–512, 2026.

[30] Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith, Yejin Choi, Kentaro Inui, et al. Realtime qa: What's the answer right now? *Advances in neural information processing systems*, 36:49025–49043, 2023.

[31] Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, et al. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*, 4:2, 2024.

[32] Constanza Fierro, Nicolas Garneau, Emanuele Bugliarello, Yova Kementchedjhieva, and Anders Søgaard. Mulan: A study of fact mutability in language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 762–771, 2024.

[33] Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273, 2022.

[34] Georg Wenzel and Adam Jatowt. Temporal validity change prediction. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1424–1446, 2024.

[35] Nishanth Sridhar Nakshatri, Shamik Roy, Manoj Ghuhan Arivazhagan, Hanhan Zhou, Vinayshekhar Bannihatti Kumar, and Rashmi Gangadharaiah. When facts change: Probing llms on evolving knowledge with evolveqa. *arXiv preprint arXiv:2510.19172*, 2025.

[36] Michael Zhang and Eunsol Choi. Mitigating temporal misalignment by discarding outdated facts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14213–14226, 2023.

[37] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[38] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

[39] Jennifer Hu and Michael C Frank. Auxiliary task demands mask the capabilities of smaller language models. *arXiv preprint arXiv:2404.02418*, 2024.

[40] Roy Jonker and Ton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38:325–340, 1987. URL `https://api.semanticscholar.org/CorpusID:7806079`.

[41] David F. Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016. doi: 10.1109/TAES.2016.140952.

[42] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[43] OpenAI. Introducing gpt-5.4, 2026. URL `https://openai.com/index/introducing-gpt-5-4/`.

[44] Anthropic. Introducing claude opus 4.7, 2026. URL `https://www.anthropic.com/news/claude-opus-4-7`.

[45] Anthropic. Introducing claude opus 4.6, 2026. URL `https://www.anthropic.com/news/claude-opus-4-6`.

[46] Anthropic. Introducing claude sonnet 4.6, 2026. URL https://www.anthropic.com/news/claude-sonnet-4-6.

[47] Anthropic. Introducing claude opus 4.5, 2025. URL https://www.anthropic.com/news/claude-opus-4-5.

[48] Anthropic. Introducing claude sonnet 4.5, 2025. URL https://www.anthropic.com/news/claude-sonnet-4-5.

[49] Anthropic. Introducing claude haiku 4.5, 2025. URL https://www.anthropic.com/news/claude-haiku-4-5.

[50] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025.

[51] Amazon Artificial General Intelligence. The amazon nova family of models: Technical report and model card, 2024. URL https://www.amazon.science/publications/the-amazon-nova-family-of-models-technical-report-and-model-card.

[52] Amazon Artificial General Intelligence. Amazon nova 2: Multimodal reasoning and generation models, 2025. URL https://www.amazon.science/publications/amazon-nova-2-multimodal-reasoning-and-generation-models.

[53] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[54] Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

[55] Alibaba Cloud. Qwen3-next: Towards ultimate training & inference efficiency, 2025. URL https://qwen.ai/blog?id=4074cca80393150c248e508aa62983f9cb7d27cd.

[56] Alibaba Cloud. Qwen3.5: Towards native multimodal agents, 2026. URL https://qwen.ai/blog?id=qwen3.5.

[57] Alibaba Cloud. Qwen3.6-35b-a3b: Agentic coding power, now open to all, 2026. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

[58] Mistral AI. Introducing mistral 3, 2025. URL https://mistral.ai/news/mistral-3.

[59] Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

[60] Abhinav Rastogi, Albert Q Jiang, Andy Lo, Gabrielle Berrada, Guillaume Lample, Jason Rute, Joep Barmentlo, Karmesh Yadav, Kartik Khandelwal, Khyathi Raghavi Chandu, et al. Magistral. *arXiv preprint arXiv:2506.10910*, 2025.

[61] Google DeepMind. Gemma 4, 2026. URL https://deepmind.google/models/gemma/gemma-4/.

Appendix Contents

A Limitations and Broader Impact **17**

B Annotation Guidelines **17**

B.1 Validation Errors 18

C Additional Dataset Statistics **19**

D List Element Parser **20**

D.1 Overview 20

D.2 List Format Specification 20

D.3 Two-Stage Parsing Architecture 20

D.4 Detailed Regex Analysis 21

E Scoring Example **22**

E.1 Relaxed-Recall 23

E.2 Relaxed-Precision 23

E.3 Format-Correctness 23

E.4 ROUGE-F1 24

E.5 LLM-Judge-F1 24

E.6 LLM-Judge-F1-Ordered 25

F Instruction Prompt Templates **26**

F.1 Instruction Prompt Template - Zero shot 26

F.2 Instruction Prompt Template - Zero shot (Order Aware) 26

F.3 Instruction Prompt Template - Oracle RAG 27

F.4 Instruction Prompt Template - Few Shot 27

F.5 Judge LLM Prompt Template 28

G Full Results **31**

H Model Evaluation Infrastructure **32**

I Format Error Analysis **33**

J Ordering Metrics - Analysis **35**

K Thinking-Enabled Inference **36**

L Oracle RAG: Setup **37**

M Timeless vs. Time-constrained Knowledge Retrieval **38**

N Effect of List Length on Performance **39**

O Performance by Topic Category **40**

P Performance by Geographic Region **41**

Q Effect of Instruction Tuning **42**

R Validation Against Human Judgment **44**

R.1 Human Evaluation of LLM-Judge-F1 44

R.2 Validating the ROUGE-Based Matching Step 44

R.3 Judge Model Agreement Analysis 45

R.4 Judge Repeatability 46

S Token Usage and Generation Efficiency **47**

S.1 Generation Cost vs. Accuracy 47

S.2 The Cost of Thinking 48

S.3 Reasoning Budget Allocation 48

S.4 Single-Faceted vs. Multi-Faceted Generation Patterns 49

S.5 Per-Sample Score Distributions 52

T Qualitative Examples **54**

T.1 Knowledge vs. Format Trade-off 54

T.2 Multi-Faceted Difficulty 55

T.3 Effect of Thinking 56

T.4 Ordering Errors 58

A Limitations and Broader Impact

Limitations. We identify the following limitations of the current work: ***(1)*** *English-only:* the benchmark is currently limited to English; multilingual extension is a natural next step. ***(2)*** *No abstention measurement:* our metrics penalize hallucinated elements, providing an indirect signal of overconfidence, but the benchmark does not yet explicitly reward models for recognizing the limits of their knowledge and declining to answer. ***(3)*** *Judge cost:* our primary metric relies on a 120B reasoning model as the judge, which adds non-trivial inference cost to each evaluation run; however, we report agreement scores with smaller judges in App. R.3 to support practitioners who prefer a lighter-weight alternative. ***(4)*** *Judge noise:* as with any LLM-as-judge evaluation, the judge model may introduce noise, particularly when a model's prediction contains additional information beyond the reference that the judge cannot verify or refute without an external grounding source. While our metric correlates well with human evaluation overall, this inherent limitation of reference-based judging applies to LISTQA as it does to other LLM-judged benchmarks. ***(5)*** *Geographic skew and language confound:* 78% of samples concern US entities, so the overall score mainly reflects US-centric knowledge and may not generalize to regions with fewer samples (App. P). Moreover, all questions are posed in English, while facts about non-English-speaking regions are often documented mainly in local languages. The regional gaps we observe therefore combine regional knowledge with English-language coverage and cannot be attributed to either alone. Posing questions in local languages against localized sources would reduce this confound but not remove it: low-resource languages are themselves underrepresented in training, and localized probing adds a language-capability factor of its own. We therefore present the geographic results as a measured boundary that LISTQA exposes rather than one it resolves. ***(6)*** *Ground-truth completeness:* each reference answer is constructed from a single Wikipedia table, and validators checked for missing items against that table (Tab. 4). We did not independently audit completeness against external sources. If a valid element exists but is absent from the source table, for example because the table is incomplete or another source lists additional qualifying entities, a model that produces it would be penalized under Relaxed-Precision and LLM-Judge-F1. The explicit temporal and categorical constraints in each question narrow the answer set and reduce this risk, and every sample includes its source URL so that such references can be audited and corrected in future releases.

Broader Impact. LISTQA encourages the development of models that are both factually comprehensive and precise, by explicitly measuring hallucination through precision and coverage through recall. The dataset is constructed entirely from publicly available Wikipedia content and contains no personally identifiable information, offensive content, or sensitive data; annotators were compensated in accordance with applicable labor standards. While improved factual retrieval could be misused to generate selectively curated information, we believe transparent evaluation of factual capabilities, including explicit hallucination measurement, is a net positive for responsible AI development.

B Annotation Guidelines

We provide the complete annotation guidelines given to human annotators for the construction of LISTQA.

Core Requirements

1. **List format.** All reference answers must be a list of at least 3 and no more than 10 elements. This avoids single-element answers (which reduce to standard QA) and very large information dumps that are difficult for users to parse. Additionally, all single and multi-faceted list answers should follow rigid formatting guidelines to ensure consistency.
2. **Close-ended and objective.** Questions must have unambiguous, objectively verifiable answers. Subjective questions (e.g., *"What are the top 10 vegan restaurants in New York City"*) are excluded to prevent evaluation disputes arising from source-dependent rankings.
3. **Wikipedia table reference.** Annotators must source questions from Wikipedia tables. Tables provide structured data that naturally supports list-formatted answers where multiple rows satisfy a query criterion.
4. **Varying difficulty.** Questions should span a range of complexity:

Table 4: Validation round outcomes and error type distribution. The same sample may be associated with one or more error types

Validation outcomes		
No issues found	3,957	39%
Fixed	5,088	51%
Discarded	987	10%
Error types identified		
Typo / grammar	3,565	44%
Vague prompt	2,502	31%
Sub-list issues	1,020	13%
Data not accurate	768	10%
Missing items	532	7%
Year range error	219	3%
Mis-matched order	144	2%

- *Single-faceted*: request a primary element from a single column.
- *Multi-faceted*: add constraints across multiple columns and request secondary sub-elements, including filtering conditions (e.g., "excluding publishers with 'entertainment' in their name").

5. **Temporal constraints.** Questions must include explicit time constraints where applicable to prevent reference answers from becoming outdated as the underlying data changes.
6. **Temporal diversity.** Questions should be distributed across time periods from 2000 to 2024, including timeless questions whose answers are historically fixed. The distribution of these time periods should favor questions that are more recent but not be lopsided.
7. **Answerable as of 2024.** All questions must be answerable as of December 31, 2024.

B.1 Validation Errors

During the validation process, annotators observed the following high level categories of errors as seen in Tab. 4. Categories or questions that required significant rewrite were discarded, constituting the 10% discard rate and other annotations were fixed.

Independent error audit. To measure the error rate of the released dataset directly, we audited a stratified random sample covering all 8 categories and both question types (5 random seeds of 100 questions each, yielding 481 unique questions after deduplication). Each reference answer was verified against its source Wikipedia table. We found errors in 8 of 481 questions (1.7%; 95% Wilson CI [0.8%, 3.2%]). These stemmed primarily from ambiguous question framing rather than factual mistakes.

C Additional Dataset Statistics

This section provides detailed breakdowns of the LISTQA dataset composition, complementing the overview in Fig. 2.

Geographic coverage. Tab. 5 shows the full country-level distribution. The United States contributes the largest share with 7,049 samples (77.9%), followed by 15 "Global" countries each contributing ∼120 samples across all major regions. Beyond these, the dataset includes countries across Europe, Asia, Africa, Australasia, and South America at smaller scales, ensuring representation from over 60 countries across 6 continents.

Table 5: Geographic distribution of samples across continents and countries (excl. USA

Global (15)		Europe		Asia		Africa		Australasia		S. America	
Spain	120	Albania	2	Cambodia	6	Angola	4	New Zealand	15	General	21
Italy	120	Lithuania	1	Sri Lanka	1	Kenya	5	Papua N.G.	3	Chile	9
Germany	119	Andorra	1	Maldives	4	Seychelles	2	Cook Islands	1	Colombia	14
Portugal	120	Austria	3	Thailand	12	Madagascar	3	General	1	Bolivia	3
France	120	Greece	1	Uzbekistan	2	Algeria	7			Argentina	6
India	120	Spain	1	S. Korea	10	Ethiopia	5			Venezuela	4
UK	120	Switzerland	1	Bangladesh	2	Mauritius	3			Uruguay	3
Japan	120	Estonia	1	Vietnam	2	Nigeria	3				
China	119	General	3	Philippines	2	Morocco	8				
UAE	120	Denmark	2	Singapore	4	Tanzania	3				
Indonesia	119	Sweden	2	China	1	General	1				
Australia	119	Finland	1	Iran	4	Uganda	1				
S. Africa	120	Poland	1	Afghanistan	1						
Brazil	120			Armenia	3						
Egypt	120			Nepal	1						

List structure. Fig. 5 (left) shows the distribution of list elements per answer. Most common answer length is 3 elements, with a right-skewed distribution tapering toward the maximum of 8. The mean is 4.67 and the median is 4, indicating that the benchmark tests retrieval of a moderate number of facts rather than exhaustive enumeration.

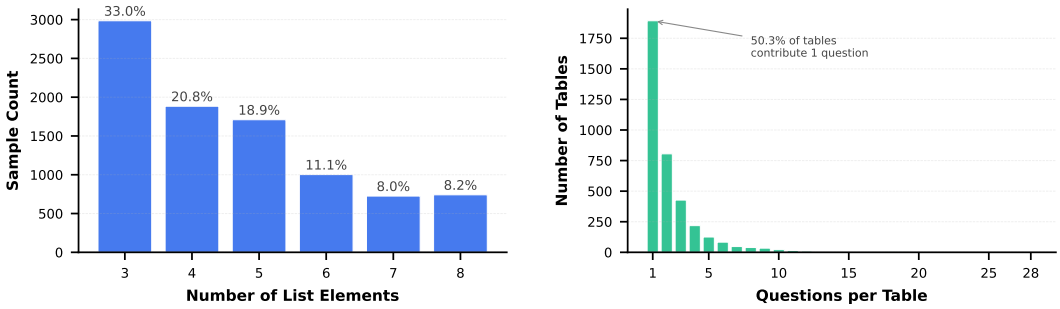


Figure 5: **Left:** Distribution of list elements per answer. Most answers contain 3 elements; the distribution is right-skewed. **Right:** Questions per unique Wikipedia table. Over half of tables contribute a single question, with a long tail extending to 28.

Source diversity. Fig. 5 (right) shows the number of questions sourced from each unique Wikipedia table. Over half of tables contribute a single question, reducing the risk of topical clustering. The long tail (up to 28 questions per table) reflects the richness of certain source tables, particularly in Sports and Entertainment.

Category balance. As shown in Fig. 2, the category distribution is fairly maintained across both SIMPLELISTQA and COMPLEXLISTQA, ensuring that neither subset is biased toward particular topics.

D List Element Parser

D.1 Overview

Evaluating list-formatted outputs from large language models (LLMs) requires the reliable extraction of individual list elements from both model-generated predictions and gold-standard reference answers. This is a non-trivial parsing problem because LLM outputs exhibit a two-level hierarchical structure: *top-level numbered items* that may optionally contain *nested sub-elements* (key–value facets). We describe a regex-based list element parser that serves as the foundational component upon which all downstream LISTQA metrics, Relaxed-Precision, Relaxed-Recall, ROUGE-F1 element matching, and LLM-Judge-F1 scoring, are built.

D.2 List Format Specification

In line with the list format guidelines provided to the annotators, the evaluation prompt instructs the model to produce lists in one of two canonical formats. Annotators' adherence to the guidelines helps us reliably parse list elements from the ground truth reference.

For **single-faceted** (simple) questions, the output format is as follows:

```
1.  List Element 1
2.  List Element 2
3.  List Element 3
```

For **multi-faceted** (complex) questions requiring structured sub-attributes, the output format is as follows:

```
1.  List Element 1
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

2.  List Element 2
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2
```

The parser must handle both formats uniformly, extracting top-level elements along with any nested sub-element structure. All answers are bounded between 3 and 10 main elements.

D.3 Two-Stage Parsing Architecture

Parsing is decomposed into two stages, each governed by its own regular expression.

D.3.1 Stage 1: Top-Level List Element Extraction

The first stage extracts all top-level numbered list items from the input string using:

$$\texttt{R}_1 = \texttt{\textbackslash n\textbackslash d+\textbackslash .\textbackslash s(.+?)(?=\textbackslash n\textbackslash d+\textbackslash .|\textbackslash Z)} \tag{1}$$

applied with the `re.DOTALL` flag. Each match captures the full content of a list element (including any multi-line body and nested sub-elements) as a single contiguous block of text. Results are lowercased and stripped of leading/trailing whitespace.

D.3.2 Stage 2: Multi-Faceted Sub-Element Extraction

The second stage is applied to each individual element extracted by Stage 1. It identifies bullet-prefixed key–value sub-elements within the body of a list item:

$$\texttt{R}_2 = \texttt{\textbackslash * (.*?): (.*?)\$} \tag{2}$$

applied with the `re.MULTILINE` flag. The main element (the text before the first `*` bullet) is extracted via string splitting, while sub-elements are parsed into structured {category $\rightarrow$ value} dictionaries.

D.4 Detailed Regex Analysis

D.4.1 Top-Level Pattern R_1

Tab. 6 provides a component-by-component breakdown of the top-level extraction regex.

Table 6: Component analysis of the top-level list element regex R_1.

Component	Purpose
`\n`	Anchors to a newline boundary, preventing false matches on numbered references appearing mid-sentence.
`\d+`	Matches one or more digits, accommodating lists of arbitrary length (1–9, 10–99, 100+) without imposing an upper bound.
`\.`	Matches the literal period delimiter, distinguishing numbered list items from other digit-newline patterns.
`\s`	Matches mandatory whitespace between the period and item text.
`(.+?)`	Non-greedy capture group. With `re.DOTALL`, the `.` metacharacter matches newlines, enabling capture of multi-line elements spanning nested sub-element lines.
`(?=\n\d+\.|\Z)`	Zero-width positive lookahead terminating capture at the start of the next numbered item or end of string (`\Z`).

D.4.2 Sub-Element Pattern R_2

Tab. 7 provides the corresponding breakdown for the sub-element extraction regex.

Table 7: Component analysis of the sub-element extraction regex R_2.

Component	Purpose
`^`	With `re.MULTILINE`, anchors to line start, matching only bullets that begin a line.
`\*`	Matches the literal asterisk bullet marker followed by a space.
`(.*?)` (1st)	Non-greedy capture of the category/key name, stopping at the first colon-space delimiter.
`:`	Literal colon-space separator between category and value.
`(.*?)` (2nd)	Non-greedy capture of the value, terminating at end of line.
`$`	With `re.MULTILINE`, anchors to line end, bounding value capture.

E Scoring Example

We illustrate how each metric is computed using two real examples from the benchmark, one each from single-faceted and multi-faceted.

Single-faceted example. *Question:* Which films won the Silwerskerm Award for Best Feature Film between 2017 and 2022?

Reference (4 elements):	**Model prediction (4 elements):**
1. Vaselinetjie	1. Wonderlus
2. Kanarie	2. Gaia
3. Poppie Nongena	3. Poppie Nongena
4. Gaia	4. Vaselinetjie

The model retrieves 3 of 4 correct names but hallucinates "Wonderlus" (which was nominated in 2017 but did not win) and misses "Kanarie". The order differs from the reference, but order is irrelevant for this question.

Multi-faceted example. *Question:* Which years between 2017 and 2024 were works performed by Dua Lipa nominated for the MTV Video Music Award for Best Choreography, and what was the name of the work?

Reference (4 elements):	**Model prediction (2 elements):**
1. 2018	1. 2018
* Name of the work: ''IDGAF''	* Work: ''IDGAF''
2. 2020	2. 2024
* Name of the work: ''Physical''	* Work: ''Houdini''
3. 2023	
* Name of the work: ''Dance the Night''	
4. 2024	
* Name of the work: ''Houdini''	

The model produces only 2 of 4 elements, missing 2020 ("Physical") and 2023 ("Dance the Night"). Both predicted elements are factually correct with no hallucination, but the output falls below the 3-element minimum.

Additionally, only content parsed from the structured list for a given element is considered; information that appears outside the expected list structure (e.g., in free-text following the list) is excluded from evaluation and deemed incorrect. Consider the following example for the question, *What are the NCAA Division I men's basketball teams that were undefeated in the tournaments between 2000 and 2024, as well as the year, the record, and the result?*

Reference Answer:	**Model prediction:**
1. Wichita State	1. Wichita State
* Year: 2014, Record: 34-0	* Year: 2014, Record: 34-0
* Result: Lost R32 to Kentucky	2. Kentucky
2. Kentucky	* Year: 2015, Record: 34-0
* Year: 2015, Record: 34-0	3. Gonzaga
* Result: Lost SF to Wisconsin	* Year: 2021, Record: 26-0
3. Gonzaga	''Wichita lost to Kentucky in R32, Kentucky lost to Wisconsin in the semis, Gonzaga lost the final to Baylor.''
* Year: 2021, Record: 26-0	
* Result: Lost final to Baylor	

As the results appear in trailing free text rather than as list sub-elements, all three `Result` fields are marked INCORRECT as factual knowledge must be delivered in the prescribed format to receive credit.

E.1 Relaxed-Recall

Relaxed-Recall measures the fraction of reference elements found *anywhere* in the model's full output text via substring matching, without requiring the output to be correctly formatted. Before comparison, both texts are normalized: the reference undergoes whitespace normalization (trailing spaces before newlines are removed) and lowercasing; the prediction text is lowercased. For multi-faceted questions, the main element and each sub-element *value* are checked independently against the full prediction text.

Single-faceted

Reference item	Found?
vaselinetjie	✓
kanarie	✗
poppie nongena	✓
gaia	✓

Recall = 3/4 = 0.75

Multi-faceted

Reference item	Found?
main: 2018	✓
sub-value: "idgaf"	✓
main: 2020	✗
sub-value: "physical"	✗
main: 2023	✗
sub-value: "dance the night"	✗
main: 2024	✓
sub-value: "houdini"	✓

Recall = 4/8 = 0.50

In the single-faceted case, substring matching works well for entity names; 3 out of 4 are found directly. In the multi-faceted case, the scorer checks each sub-element's *value* (e.g., `"idgaf"`) rather than the full key-value string, so matching succeeds for the two predicted elements. However, the two missing elements (2020/"Physical" and 2023/"Dance the Night") are entirely absent from the prediction, yielding 4 hits out of 8 total items.

E.2 Relaxed-Precision

Relaxed-Precision measures the fraction of *predicted* list elements that match a reference element, penalizing hallucinated items. The same normalization is applied as for Relaxed-Recall. For multi-faceted predictions, the main element and each sub-element value are checked independently against the reference text, mirroring the recall computation. If a direct substring match fails, a fallback check tests whether any flattened reference element is contained within the predicted item.

Single-faceted

Predicted item	In reference?
wonderlus	✗ (hallucinated)
gaia	✓
poppie nongena	✓
vaselinetjie	✓

Precision = 3/4 = 0.75

Multi-faceted

Predicted item	In reference?
main: 2018	✓
sub-value: "idgaf"	✓
main: 2024	✓
sub-value: "houdini"	✓

Precision = 4/4 = 1.00

The single-faceted example shows precision catching the "Wonderlus" hallucination. The multi-faceted example achieves perfect precision (both predicted elements are correct) but this masks the fact that 2 of 4 reference elements are entirely missing. Precision and recall are complementary: high precision with low recall indicates a conservative model that produces few but accurate elements.

E.3 Format-Correctness

Format-Correctness evaluates structural compliance without judging factual content. The prediction is parsed into list elements using a regex pattern. If fewer than 3 elements are parsed, the score is automatically 0. Otherwise, for each parsed element, the scorer checks whether sub-elements are present: single-faceted questions expect no sub-bullets, multi-faceted questions expect sub-bullets. The score is the fraction of correctly structured elements.

Single-faceted

Predicted element	Correct?
wonderlus	✓ (flat, no subs)
gaia	✓ (flat, no subs)
poppie nongena	✓ (flat, no subs)
vaselinetjie	✓ (flat, no subs)

4 elements $\geq$ 3 minimum ✓
Format = 4/4 = 1.00

Multi-faceted

Predicted element	Correct?
2018 / * Work: "IDGAF"	✓ (has subs)
2024 / * Work: "Houdini"	✓ (has subs)

2 elements $<$ 3 minimum ✗
Format = 0.00

The single-faceted prediction uses the correct flat format and meets the 3-element minimum, as instructed in the prompt (see App. F). The multi-faceted prediction uses the correct sub-bullet structure for both elements, but produces only 2 elements (below the required minimum of 3), so it receives a format score of 0 regardless of structural correctness. This design choice ensures that models cannot achieve high Format-Correctness by producing trivially short lists.

E.4 ROUGE-F1

ROUGE F1 handles element reordering and partial lexical overlap via a two-step process. First, each list element is flattened into a single string by concatenating the main element and all sub-element values (e.g., `2018 "idgaf"`). A ROUGE-1 F-measure is computed between every predicted–reference pair, forming a cost matrix. The Jonker-Volgenant algorithm finds the optimal one-to-one assignment that maximizes total ROUGE-1 score. A pair is considered matched if its ROUGE-1 score ≥ 0.5. Precision and recall are computed over matched pairs.

Single-faceted

Predicted	Matched ref.	R-1
wonderlus	*(none $\geq$ 0.5)*	0.00
gaia	gaia	1.00 ✓
poppie nongena	poppie nongena	1.00 ✓
vaselinetjie	vaselinetjie	1.00 ✓

$P = 3/4,\ R = 3/4$
ROUGE-F1 = 0.75

Multi-faceted

Predicted	Matched ref.	R-1
2018 "idgaf"	2018 "idgaf"	1.00 ✓
2024 "houdini"	2024 "houdini"	1.00 ✓

$P = 2/2,\ R = 2/4$
ROUGE-F1 = 0.67

For the single-faceted case, exact entity name matches yield ROUGE-1 = 1.0, and "wonderlus" has 0.0 overlap with all reference elements. For the multi-faceted case, both predicted elements match their corresponding reference elements perfectly after flattening. Precision is 1.0 (both predictions are correct) but recall is only 0.5 (2 of 4 reference elements are matched), yielding F1 = 0.67.

E.5 LLM-Judge-F1

The LLM-Judge-F1 follows the same Jonker-Volgenant assignment process as ROUGE-F1 but with two differences: (1) the ROUGE filter threshold is set to 0.0 instead of 0.5, allowing even low-overlap pairs to be candidates for matching, and (2) each matched pair is sent to an LLM grader that returns a CORRECT or INCORRECT verdict based on semantic equivalence (see App. F.5 for the full judge prompt). Predicted elements that remain unmatched after assignment are automatically graded INCORRECT. Precision and recall are computed over elements judged CORRECT.

Single-faceted

Predicted	Matched ref.	Verdict
wonderlus	*unmatched*	✗
gaia	gaia	✓
poppie nongena	poppie nongena	✓
vaselinetjie	vaselinetjie	✓

$P = 3/4,\ R = 3/4$
LLM-Judge-F1 = 0.75

Multi-faceted

Predicted	Matched ref.	Verdict
2018 / "IDGAF"	2018 / "IDGAF"	✓
2024 / "Houdini"	2024 / "Houdini"	✓

$P = 2/2,\ R = 2/4$
LLM-Judge-F1 = 0.67

In these examples, the LLM-Judge-F1 agrees with ROUGE-F1 because the matches are either exact or clearly unrelated. The metrics diverge in cases with heavy paraphrasing (where the judge recognizes semantic equivalence that ROUGE misses, e.g., "championship game" vs. "title game") or when token overlap is high but facts are wrong (where ROUGE gives credit but the judge does not, e.g., a predicted year of 2002 vs. a gold year of 2000 would share most surrounding tokens but be factually incorrect).

E.6 LLM-Judge-F1-Ordered

For the 14.6% of questions requiring a specific sequence, we additionally compute an ordered variant of LLM-Judge-F1 that penalizes correct elements in the wrong position. Consider the following ordered question:

Question: Who were the first 4 winners of the Big Apple Award for Game of the Year?

Reference (4 elements):	**Model prediction (4 elements):**
`1. The Elder Scrolls V: Skyrim`	`1. The Walking Dead`
`2. The Walking Dead`	`2. The Elder Scrolls V: Skyrim`
`3. The Last of Us`	`3. The Last of Us`
`4. Wolfenstein: The New Order`	`4. Wolfenstein: The New Order`

The model retrieves all four correct elements, but swaps the first two. Under the standard (order-agnostic) LLM-Judge-F1, all four are matched as CORRECT, yielding $P = R = 1.0$ and **LLM-Judge-F1 = 1.0**. Under the ordered variant, elements are matched positionally: position 1 (The Walking Dead $\neq$ The Elder Scrolls V: Skyrim) and position 2 (The Elder Scrolls V: Skyrim $\neq$ The Walking Dead) are marked INCORRECT, while positions 3 and 4 match. This yields $P = R = 0.5$ and **LLM-Judge-F1-Ordered = 0.5**. The gap between the two scores isolates ordering as a source of error distinct from factual retrieval.

F Instruction Prompt Templates

This section describes the prompt template used as part of different evaluation settings, as well as the LLM Judge prompt template.

F.1 Instruction Prompt Template - Zero shot

The following prompt template is used during evaluation in the zero-shot setting:

```
Answer the question below by producing a list-formatted output,
lists can be formatted one of two ways:

For simple questions with few constraints requesting only main
elements and not requesting sub-elements, use the following format:

1.  List Element 1
2.  List Element 2

For complex questions asking for multiple factoids in addition to
the primary question, use the following format:

1.  List Element 1
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

2.  List Element 2
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

All answers are bounded by 3 to 10 main elements, do not generate
more than 10 main elements in the list.  Do not add any information
that is not asked for in the question.

<QUESTION>
```

F.2 Instruction Prompt Template - Zero shot (Order Aware)

The following template is used during order-aware evaluation (App. J) in the zero-shot setting:

```
Answer the question below by producing a list-formatted output,
lists can be formatted one of two ways:

For simple questions with few constraints requesting only main
elements and not requesting sub-elements, use the following format:

1.  List Element 1
2.  List Element 2

For complex questions asking for multiple factoids in addition to
the primary question, use the following format:

1.  List Element 1
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

2.  List Element 2
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

All answers are bounded by 3 to 10 main elements, do not generate
more than 10 main elements in the list.  When the question specifies
or implies a natural ordering (e.g.  chronological, ranked, or
sequential), list elements must follow that order for correctness
```

```
of answer.  Do not add any information that is not asked for in the
question.

<QUESTION>
```

F.3 Instruction Prompt Template - Oracle RAG

The following template is used during evaluation in the oracle RAG setting:

```
You are given a context passage and a question.  Using only the
provided context, answer the question by producing a list-formatted
output.  Lists can be formatted one of two ways:

For simple questions with few constraints requesting only main
elements and not requesting sub-elements, use the following format:

1.  List Element 1
2.  List Element 2

For complex questions asking for multiple factoids in addition to
the primary question, use the following format:

1.  List Element 1
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

2.  List Element 2
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

All answers are bounded by 3 to 10 main elements, do not generate
more than 10 main elements in the list.  Do not add any information
that is not asked for in the question.

Context:
<CONTEXT>
Question:  <QUESTION>
```

F.4 Instruction Prompt Template - Few Shot

The following template is used during evaluation in the few-shot setting. By default, we provide 5 examples per prompt. For few-shot evaluation, examples are drawn from the corresponding question type: single-faceted examples for SIMPLELISTQA, multi-faceted for COMPLEXLISTQA, and a combination for the full dataset.

```
Answer the question below by producing a list-formatted output,
lists can be formatted one of two ways:

For simple questions with few constraints requesting only main
elements and not requesting sub-elements, use the following format:

1.  List Element 1
2.  List Element 2

For complex questions asking for multiple factoids in addition to
the primary question, use the following format:

1.  List Element 1
* Category 1:  Sub-Element 1
* Category 2:  Sub-Element 2

2.  List Element 2
* Category 1:  Sub-Element 1
```

```
* Category 2:  Sub-Element 2

All answers are bounded by 3 to 10 main elements, do not generate
more than 10 main elements in the list.  Do not add any information
that is not asked for in the question.

Here are some examples:

Question:  <EXAMPLE_1_QUESTION>
Answer:  <EXAMPLE_1_ANSWER>

Question:  <EXAMPLE_2_QUESTION>
Answer:  <EXAMPLE_2_ANSWER>

Question:  <EXAMPLE_3_QUESTION>
Answer:  <EXAMPLE_3_ANSWER>

Question:  <EXAMPLE_4_QUESTION>
Answer:  <EXAMPLE_4_ANSWER>

Question:  <EXAMPLE_5_QUESTION>
Answer:  <EXAMPLE_5_ANSWER>

Question:  <QUESTION>
Answer:
```

F.5 Judge LLM Prompt Template

Our judge prompt builds on the grading template from SimpleQA [6], a well-established factual knowledge benchmark, and extends it to handle the list-structured, multi-element responses required by LISTQA. The complete prompt template is as follows:

```
Your job is to look at a question, a gold target, and a predicted
answer, and then assign a grade of either ["CORRECT", "INCORRECT"].
The answer to the question is always going to be a list, however we
will only be judging each list element one at a time, so the gold
target and predicted answer may not completely answer the question
and that is ok.

First, I will give examples of each grade, and then you will grade a
new example.

The following are examples of CORRECT predicted answers.

Question:  What are the names of Barack Obama's children?
Gold target:  malia obama
Predicted answer 1:  malia
Predicted answer 2:  Malia Obama
* birthday:  July 4, 1998
Predicted answer 3:  malia obama (born on July 4, 1998).

Question:  What NFL teams has Tom Brady played for and what years
was he active at these teams?
Gold target:  New England Patriots
* Start Year:  2000
* End Year:  2019
Predicted answer 1:  New England Patriots (2000 - 2019)
Predicted answer 2:  patriots
* duration:  2000 - 2019
Predicted answer 3:  NE Patriots
* Start:  2000
```

```
* End:  2019

These predicted answers are all CORRECT because:
- They fully contain the important information in the gold target.
- They do not contain any information that contradicts the gold
target.
- Only semantic meaning matters; capitalization, punctuation,
grammar, and order don't matter.
- Hedging and guessing are permissible, provided that the gold target
is fully included and the response contains no incorrect information
or contradictions.

The following are examples of INCORRECT predicted answers.

Question:  What are the names of Barack Obama's children?
Gold target:  Malia
Predicted answer 1:  Susan
Predicted answer 2:  Malia, Sasha, and Susan.
Predicted answer 3:  Two children.
Predicted answer 4:  Obama Family Tree
* Father:  Barrack
* Mother:  Michelle
* Child 1:  Malia
* Child 2:  Sasha

Question:  What NFL teams has Tom Brady play for and what years was
he active at these teams?
Gold target:  New England Patriots
* Start Year:  2000
* End Year:  2019
Predicted answer 1:  New England Patriots
Predicted answer 2:  New England Patriots
* Start Year:  2002
* End Year:  2019
Predicted answer 3:  Jacksonville Jaguars
Predicted answer 4:  Jacksonville Jaguars
* Duration:  2000 - 2019
Predicted answer 5:  New England Patriots
* Duration:  2000 - 2019
* Superbowl Wins:  7
* Touchdowns:  450
* Interceptions:  6

These predicted answers are all INCORRECT because:
- A factual statement in the answer contradicts the gold target
or does not provide all the requested information.  Incorrect
statements also provide information that cannot be validated against
the gold target are also considered incorrect.

Also note the following things:


  • For grading questions where the gold target is a number, the
    predicted answer needs to be correct to the last significant
    figure in the gold answer.  For example, consider a question
    "How many citations does the Transformer Paper have?" with gold
    target "120k".
      – Predicted answers "120k", "124k", and "115k" are all CORRECT.
      – Predicted answers "100k" and "113k" are INCORRECT.
      – Predicted answers "around 100k" and "more than 50k" are
        considered INCORRECT because they neither confirm nor
        contradict the gold target.
```

- The gold target may contain more information than the question. In such cases, the predicted answer only needs to contain the information that is in the question.
  - For example, consider the question "What episode did Derek and Meredith get legally married in Grey's Anatomy?" with gold target "Season 7, Episode 20: White Wedding". Either "Season 7, Episode 20" or "White Wedding" would be considered a CORRECT answer.
- Do not punish predicted answers if they omit information that would be clearly inferred from the question.
  - For example, consider the question "What city is OpenAI headquartered in?" and the gold target "San Francisco, California". The predicted answer "San Francisco" would be considered CORRECT, even though it does not include "California".
  - Consider the question "What award did A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity win at NAACL '24?", the gold target is "Outstanding Paper Award". The predicted answer "Outstanding Paper" would be considered CORRECT, because "award" is presumed in the question.
  - For the question "What is the height of Jason Wei in meters?", the gold target is "1.73 m". The predicted answer "1.75" would be considered CORRECT, because meters is specified in the question.
  - For the question "What is the name of Barack Obama's wife?", the gold target is "Michelle Obama". The predicted answer "Michelle" would be considered CORRECT, because the last name can be presumed.
- Do not punish for typos in people's name if it's clearly the same name.
  - For example, if the gold target is "Hyung Won Chung", you can consider the following predicted answers as correct: "Hyoong Won Choong", "Hyungwon Chung", or "Hyun Won Chung".

Here is a new example. Simply reply with either CORRECT, INCORRECT. Don't apologize or correct yourself if there was a mistake; we are just trying to grade the answer.

Question: <QUESTION>
Gold target: <GOLD TARGET LIST ITEM>
Predicted answer: <PREDICTED LIST ITEM>

Grade the predicted answer of this new question as one of:
A: CORRECT
B: INCORRECT

Just return the letters "A", or "B", with no text around it.

G Full Results

Borrowing the notion of task demands from Hu and Frank [39], we vary demand at scoring time rather than elicitation: Relaxed-Recall/Relaxed-Precision ignore output structure (reduced demand), while LLM-Judge-F1 requires it (greater demand). Tab. 8 reports all five metrics across all 42 evaluated configurations (31 standard + 11 thinking-enabled) across 8 model families, sorted by overall *LLM-Judge-F1*. For each metric, we report scores on the full dataset (All), SIMPLELISTQA (S), and COMPLEXLISTQA (C).

Table 8: Full results across all five metrics. All = Overall, S = SIMPLELISTQA, C = COMPLEXLISTQA. [† = thinking-enabled inference], [* = Preview Models]. All values are percentages. Models sorted by overall *LLM-Judge-F1*. **Recall:** Relaxed-Recall; **Precision:** Relaxed-Precision; **Format:** Format-Correctness

	LLM-Judge-F1 ↑			**Recall ↑**			**Precision ↑**			**Format ↑**			**ROUGE-F1 ↑**		
Model	**All**	**S**	**C**	**All**	**S**	**C**	**All**	**S**	**C**	**All**	**S**	**C**	**All**	**S**	**C**
Gemini 3.1 Pro*†	60.7	67.7	50.3	59.6	59.9	59.4	63.7	66.5	61.8	97.7	98.5	97.2	68.5	68.8	68.3
Gemini 3 Flash*†	56.4	63.7	45.8	59.7	59.4	60.2	57.8	54.9	62.1	91.9	87.3	98.8	63.7	62.6	65.5
Gemini 3.1 Flash Lite*†	40.1	48.4	27.9	44.8	44.1	45.7	38.1	28.5	52.1	39.9	3.8	92.7	38.8	33.2	47.0
Gemini 2.5 Flash†	38.3	47.1	25.4	43.0	42.2	44.2	52.3	45.9	61.8	70.3	86.8	46.2	43.3	47.8	36.8
Claude Opus 4.6†	37.9	46.2	25.7	43.2	42.3	44.5	50.8	45.9	58.0	66.8	69.0	63.7	38.6	40.6	35.6
GPT-5.4	36.5	44.5	24.9	42.2	40.9	44.1	44.2	40.3	49.9	87.2	85.0	90.6	44.6	45.5	43.3
GPT-5.4-mini†	36.2	45.0	23.4	39.9	40.5	39.6	52.1	46.2	60.7	60.3	76.4	36.7	39.5	43.9	33.0
Claude Opus 4.7	35.0	42.1	24.6	39.5	38.2	41.3	42.1	38.9	46.9	75.0	75.4	74.5	40.0	39.6	40.5
Claude Opus 4.6	34.6	43.0	22.4	40.2	38.9	42.1	45.6	41.9	50.9	78.3	73.9	84.8	40.5	41.5	39.0
Claude Opus 4.5	33.9	42.0	22.2	38.9	37.6	40.9	41.7	37.5	47.8	70.8	63.0	82.1	38.7	38.8	38.5
Gemini 2.5 Flash	33.7	41.9	21.6	37.8	36.7	39.6	43.7	40.1	49.1	76.4	72.5	82.2	42.3	43.3	40.8
Nova Premier	31.6	39.0	20.6	39.5	38.0	41.6	41.6	33.6	53.4	60.7	60.6	61.0	32.0	31.8	32.3
DeepSeek V3.1	29.3	36.7	18.3	36.0	34.1	38.8	40.8	34.5	50.0	73.3	76.0	69.4	33.4	35.1	31.0
Claude Sonnet 4.6	28.0	35.5	16.9	34.2	32.5	36.7	36.7	30.8	45.5	60.2	45.3	82.2	31.5	31.9	31.0
GPT-5.4-mini	27.0	35.2	15.1	33.3	31.5	36.0	44.2	35.5	56.9	64.1	84.8	33.6	32.5	37.3	25.4
Mistral Large 3 675B-A41B	25.7	33.2	14.8	37.7	36.1	40.0	44.5	33.7	60.5	58.9	77.9	31.0	25.0	28.6	19.7
Claude Sonnet 4.5	25.3	30.2	18.1	30.9	27.9	35.2	31.6	25.7	40.2	49.2	41.1	61.0	27.9	26.1	30.5
Qwen 3.5 397B-A17B	24.8	31.6	14.8	32.6	30.4	35.8	33.9	29.4	40.5	73.0	69.0	78.8	29.9	31.2	28.0
Qwen 3.5 122B-A10B†	23.5	30.5	13.1	29.5	27.0	33.2	34.5	29.3	42.1	95.1	95.0	95.2	30.8	32.5	28.2
Nova Pro	22.0	29.3	11.4	30.6	28.5	33.8	31.8	24.9	42.1	65.6	53.8	83.0	25.5	25.3	25.8
Gemini 2.5 Flash Lite	20.4	27.2	10.4	28.2	25.6	32.0	37.8	28.4	51.5	63.9	83.6	35.1	25.9	30.0	20.0
DeepSeek V3.2	20.3	26.5	11.1	31.3	28.7	35.0	41.0	30.7	56.1	54.8	71.9	29.8	18.9	21.9	14.7
Qwen 3.5 122B-A10B	18.9	25.3	9.7	28.0	25.5	31.7	30.4	24.4	39.2	82.2	84.1	79.5	25.7	27.8	22.6
Gemma 4 31B†	18.7	25.3	9.0	23.3	20.9	26.8	31.2	27.3	37.0	74.9	73.9	76.5	27.4	29.1	24.8
Qwen 3.5 27B†	18.3	24.6	9.1	24.9	22.0	29.0	30.0	24.8	37.7	95.3	95.9	94.6	26.5	28.2	24.0
Qwen 3.5 35B-A3B†	18.2	24.4	9.0	24.6	21.5	29.1	29.8	24.5	37.5	90.1	89.6	91.0	25.8	27.7	22.9
Qwen 3-Next 80B-A3B	17.4	23.6	8.3	24.7	21.8	28.9	25.3	17.1	37.4	39.1	7.8	85.1	14.7	11.5	19.4
Gemma 4 31B	16.0	22.0	7.3	20.6	17.9	24.7	31.3	26.5	38.4	54.3	45.8	66.7	22.4	22.8	21.8
Magistral Small 24B	15.8	21.8	7.0	24.3	20.9	29.2	27.6	19.4	39.5	59.6	48.0	76.8	18.6	19.0	18.0
Qwen 3.5 27B	14.8	20.1	7.1	23.6	20.5	28.2	27.6	20.5	38.1	71.9	71.6	72.4	20.1	21.2	18.4
GPT-5.4-nano	14.5	20.4	5.7	22.1	19.2	26.3	31.5	20.8	47.1	58.1	77.7	29.2	16.8	20.2	11.9
Gemma 4 26B-A4B	13.8	19.0	6.0	19.2	16.1	23.7	28.7	23.4	36.6	52.5	44.2	64.6	19.6	19.7	19.4
Qwen 3.5 35B-A3B	13.4	18.5	5.9	22.4	19.0	27.4	25.7	18.8	35.9	51.7	37.0	73.2	13.9	13.7	14.2
Nova Micro	12.0	17.5	4.0	19.3	16.1	24.0	30.3	18.3	47.9	56.8	79.1	24.1	17.4	20.6	12.7
Qwen 3.5 9B†	11.4	16.1	4.4	17.1	14.0	21.6	23.1	17.9	30.8	87.2	87.0	87.5	16.5	19.0	12.8
Claude Haiku 4.5	11.1	14.1	6.7	16.4	13.7	20.3	16.7	13.4	21.5	49.1	58.6	35.2	13.7	13.5	14.1
Qwen 3.5 9B	8.3	12.0	2.9	17.9	13.9	23.8	20.9	13.1	32.4	62.0	65.5	56.8	10.8	12.2	8.8
Ministral 3 14B	7.2	10.4	2.6	18.8	15.2	24.1	34.0	19.9	54.8	46.9	76.2	3.8	3.9	6.0	0.9
Gemma 4 E4B	6.2	9.1	1.9	10.2	7.6	13.9	17.4	11.7	25.7	33.6	44.9	17.0	9.2	11.0	6.6
Qwen 3.5 4B	5.1	7.8	1.3	11.7	8.2	16.9	18.3	10.2	30.2	39.2	43.9	32.3	6.5	8.2	3.9
Ministral 3 8B	4.8	6.9	1.7	16.1	12.1	22.0	31.2	17.5	51.3	45.7	71.0	8.6	2.4	3.5	0.9
Ministral 3 3B	3.3	5.1	0.5	11.8	7.9	17.5	26.3	12.9	46.0	46.2	76.3	2.1	2.5	3.9	0.5

H Model Evaluation Infrastructure

All evaluations were orchestrated using `inspect-ai`.[2] Models accessed via API (OpenAI[3], Google Gemini[4], Amazon Bedrock[5]) were evaluated using the default inference configurations provided by each API for the respective models. Self-hosted models were served using vLLM[6] with an OpenAI-compatible API endpoint, using official HuggingFace checkpoints and the default `generation_config.json` parameters provided with each checkpoint.

For thinking-enabled evaluations, we configured `--reasoning-effort` for Gemini models and `--reasoning-tokens` (budget) for Claude models via `inspect-ai`'s generation parameters. Tab. 9 details the infrastructure used for each model family.

Table 9: Inference infrastructure for each evaluated model.

Model Family	Models	Infrastructure
GPT	*GPT-5.4* *GPT-5.4 mini* *GPT-5.4 nano*	OpenAI API
Claude	Claude *Opus 4.7, 4.6, 4.5* Claude *Sonnet 4.6, 4.5* Claude *Haiku 4.5*	Amazon Bedrock
Gemini	*Gemini 3.1 Pro* *Gemini 3.1 Flash Lite* *Gemini 3 Flash* *Gemini 2.5 Flash* *Gemini 2.5 Flash Lite*	Google (Gemini) API
Nova	*Nova Premier* *Nova Pro* *Nova Micro*	Amazon Bedrock
DeepSeek	*DeepSeek V3.2* *DeepSeek V3.1*	Amazon Bedrock
Qwen 3.5	*Qwen 3.5 397B-A17B (FP8)* *Qwen 3.5 122B-A10B* *Qwen 3.5 35B-A3B* *Qwen 3.5 27B* *Qwen 3.5 9B* *Qwen 3.5 4B*	Self-hosted (vLLM)
Qwen 3	*Qwen 3 Next 80B-A3B*	Amazon Bedrock
Mistral	*Mistral Large 3 675B-A41B* *Magistral Small 24B* *Ministral 3 14B* *Ministral 3 8B* *Ministral 3 3B*	Amazon Bedrock
Gemma	*Gemma 4 31B* *Gemma 4 26B-A4B* *Gemma 4 E4B*	Self-hosted (vLLM)

[2] `https://inspect.ai-safety-institute.org.uk/`
[3] `https://platform.openai.com/`
[4] `https://ai.google.dev/`
[5] `https://aws.amazon.com/bedrock/`
[6] `https://docs.vllm.ai/`

I Format Error Analysis

We categorize format errors into four types to understand *how* models fail to follow list-formatting instructions:

- **Wrong facet type**: The model produces a list but uses the wrong structure; e.g., generating sub-bullets for a single-faceted question or a flat list for a multi-faceted question. This is the most common error on average, indicating that many models find it difficult to infer the appropriate list style from the question.
- **Too few elements** (<3): The model produces a list with only one or two elements, falling below the minimum of 3 required by the task.
- **No list output**: The model generates a free-text response without any list structure.
- **Mixed facet type**: The model produces a list where some elements use sub-bullets and others do not, mixing single-faceted and multi-faceted formatting within a single response.

The dominant failure mode on average is *wrong facet type*, followed by *too few elements*. *Mixed facet type* is rare, and thinking nearly eliminates it. Tab. 10 shows how these error types shift when thinking is enabled.

Table 10: Format error distribution (% of 9,045 samples) under standard (Std) vs. thinking-enabled (Think) inference for models from Fig. 4(b). **Perfect** indicates accurate format compliance.

Model	Setting	Perfect	Wrong Facet	Too Few Elem.	No List	Mixed Facet
GPT-5.4 mini	Std	63.9	21.8	14.0	0.0	0.2
	Think	60.1 ↓	17.8 ↓	21.7 ↑	0.0	0.3 ↑
Gemini 2.5 Flash	Std	76.2	9.3	13.4	0.6	0.5
	Think	70.2 ↓	18.5 ↑	9.8 ↓	1.2 ↑	0.2 ↓
Claude Opus 4.6	Std	77.4	7.5	12.1	1.5	1.5
	Think	66.7 ↓	16.2 ↑	12.4 ↑	4.4 ↑	0.2 ↓
Qwen 3.5 122B-A10B	Std	81.2	5.9	6.6	4.6	1.6
	Think	95.1 ↑	4.2 ↓	0.5 ↓	0.1 ↓	0.1 ↓
Gemma 4 31B	Std	53.9	14.4	29.0	2.0	0.7
	Think	74.9 ↑	3.7 ↓	11.8 ↓	9.5 ↑	0.0 ↓
Qwen 3.5 35B-A3B	Std	49.8	26.0	20.2	0.6	3.5
	Think	90.0 ↑	9.0 ↓	0.7 ↓	0.0 ↓	0.3 ↓
Qwen 3.5 27B	Std	70.3	14.1	11.7	1.3	2.6
	Think	95.3 ↑	4.5 ↓	0.1 ↓	0.0 ↓	0.0 ↓
Qwen 3.5 9B	Std	55.4	19.8	11.0	2.8	11.0
	Think	85.8 ↑	10.7 ↓	0.3 ↓	0.8 ↓	2.4 ↓

Effect of thinking on format errors. The comparisons reveal a clear split in how thinking affects format compliance. Notably, the three closed-source models (GPT-5.4 mini, Gemini 2.5 Flash, Claude Opus 4.6) already achieve relatively strong format compliance under standard inference, but see perfect format rates *decrease* with thinking, suggesting that the thinking process produces more verbose output that disrupts an already well-structured response. In contrast, the five open-weight models (Qwen 3.5 variants, Gemma 4 31B) start with lower format compliance and benefit substantially from thinking, which helps them correctly interpret question complexity and eliminate *wrong-facet* and *too-few-elements errors*.

Several patterns emerge from the error breakdown. For the open-weight models, thinking nearly eliminates the *too few elements* error; models that previously produced truncated lists now generate the expected number of items. *Wrong facet type* errors also drop sharply, indicating that thinking helps models distinguish between single-faceted and multi-faceted question types. *Mixed facet type*, already rare under standard inference, is virtually eliminated with thinking across all models. The *no list output* error remains negligible for most models regardless of setting, though Gemma 4 31B shows

a notable increase with thinking, suggesting its reasoning process occasionally produces unstructured output.

For the closed-source models, the pattern is reversed: thinking *increases* wrong-facet errors, particularly for Gemini 2.5 Flash and Claude Opus 4.6, and GPT-5.4 mini sees a rise in too-few-elements errors. This suggests that the extended reasoning in these models may over-elaborate or restructure the output in ways that deviate from the expected format, even as it improves factual accuracy.

J Ordering Metrics - Analysis

Approximately 14.6% of LISTQA samples require ordered responses, i.e., questions that implicitly or explicitly demand a specific sequence (e.g., chronological, ranked). To evaluate this, we introduce an order-aware prompt variant (App. F.2) that instructs the model to preserve ordering when the question requires it, and evaluate three models under this prompt. We report three scores in Tab. 11: (1) *LLM-Judge-F1* under the original prompt, as a baseline; (2) *LLM-Judge-F1* under the order-aware prompt, to verify that the ordering instruction does not degrade factual accuracy; and (3) a *Blended-F1* score under the order-aware prompt, which applies the stricter *LLM-Judge-F1-Ordered* metric on the 1,319 ordered samples and the standard LLM-Judge-F1 metric on the remaining 7,726 unordered samples.

Table 11: F1 metrics on the full dataset (9,045 samples) under different prompt and scoring configurations. †Thinking enabled.

		LLM-Judge-F1		**Blended-F1**
Model	**Level**	**Original Prompt**	**Order-Aware Prompt**	**Order-Aware Prompt**
Gemini 3.1 Pro†	LOW	60.7	61.2	59.4
Gemini 3.1 Flash Lite†	MINIMAL	40.1	40.2	38.4
GPT-5.4-mini†	MEDIUM	36.2	36.6	35.0

The first two columns confirm that the order-aware prompt does not degrade factual accuracy. The *Blended-F1* score is slightly lower because it applies the stricter ordered metric on the 14.6% of samples that require sequencing. The gap is modest since the ordered subset is only a small fraction of the full dataset.

Zooming into the ordered subset reveals a larger gap. To isolate the effect of ordering, Tab. 12 restricts evaluation to the 1,319 ordered samples only. Here the gap between the standard and ordered metric becomes more pronounced.

Table 12: LLM-Judge-F1 on the *ordered subset only* (1,319 samples) under the order-aware prompt. **Unord.**: standard LLM-Judge-F1 (order-agnostic). **Ord.**: LLM-Judge-F1-Ordered (penalizes wrong sequence). Results are shown for single-faceted and multi-faceted questions *within the ordered subset*. †Thinking enabled.

	Overall		**Single-faceted**		**Multi-faceted**	
Model	**Unord.**	**Ord.**	**Unord.**	**Ord.**	**Unord.**	**Ord.**
Gemini 3.1 Pro†	55.5	43.1	62.7	47.8	45.2	36.4
Gemini 3.1 Flash Lite†	35.5	23.6	43.1	27.9	24.6	17.3
GPT-5.4-mini†	32.8	22.1	41.2	26.4	20.7	16.0
Claude Opus 4.6†	36.2	26.0	43.7	30.7	25.5	19.3

On the ordered subset, the ordered metric drops well below the standard metric for all models, confirming that models retrieve correct list elements but frequently produce them in the wrong sequence. This aligns with a central motivation of LISTQA: evaluating the interplay between factual knowledge and instruction following. Even when models possess the required knowledge, they may fail to follow the implicit or explicit ordering constraint, thereby demonstrating another instance of the knowledge–instruction-following tension observed throughout our experiments.

The gap is consistently larger on single-faceted questions than on multi-faceted ones. This may reflect the nature of the ordered single-faceted subset: these questions often ask for ranked or chronological sequences where the knowledge is simple but ordering is the primary challenge. In multi-faceted ordered questions, the model is already struggling with sub-attribute retrieval (as reflected in lower unordered scores), leaving less room for an additional ordering penalty.

K Thinking-Enabled Inference

Tab. 13 reports exact scores and thinking levels for all models evaluated with thinking enabled. Where a standard-inference baseline exists, we report the difference (Δ). Key findings from this table are discussed in Sec. 4.2; here we highlight additional observations.

Table 13: Effect of thinking-enabled inference on LISTQA. Δ denotes the change relative to the same model under standard inference (results in Tab. 2). Models with paired standard-inference baselines are grouped below the line.

		LLM-Judge-F1 ↑						Format ↑	
		Overall		S		C			
Model	**Think Level**	**Score**	**Δ**	**Score**	**Δ**	**Score**	**Δ**	**Score**	**Δ**
Gemini 3.1 Pro†	LOW	**60.7**	—	**67.7**	—	**50.3**	—	**97.7**	—
Gemini 3 Flash†	HIGH	56.4	—	63.7	—	45.8	—	91.9	—
Gemini 3.1 Flash Lite†	MINIMAL	40.1	—	48.4	—	27.9	—	39.9	—
Paired comparisons (same model, thinking vs. standard)									
GPT-5.4-mini	MEDIUM	36.2	+9.2	45.0	+9.8	23.4	+8.3	60.3	−3.8
Gemini 2.5 Flash	DYNAMIC	38.3	+4.6	47.1	+5.2	25.4	+3.7	70.3	−6.1
Claude Opus 4.6*		37.9	+3.3	46.2	+3.2	25.7	+3.3	66.8	−11.5
Qwen 3.5 122B-A10B		23.5	+4.6	30.5	+5.2	13.1	+3.4	95.1	+12.9
Gemma 4 31B		18.7	+2.7	25.3	+3.3	9.0	+1.8	74.9	+20.6
Qwen 3.5 27B		18.2	+3.4	24.5	+4.4	9.0	+1.9	95.3	+23.4
Qwen 3.5 35B-A3B		18.2	+4.8	24.4	+5.9	9.0	+3.1	90.1	+38.4
Qwen 3.5 9B		11.4	+3.1	16.1	+4.1	4.4	+1.5	87.2	+25.2

*2,048 token reasoning budget | †Preview models, evaluated as of April 21, 2026 | Gemini 3 models do not support non-thinking inference

COMPLEXLISTQA remains challenging even with thinking: only one model (Gemini 3.1 Pro) crosses 50% LLM-Judge-F1 on the complex subset, and only barely. Most thinking-enabled models remain below 30% on complex questions, confirming that multi-faceted retrieval with hierarchical sub-attributes is a fundamentally harder task that thinking alone does not resolve.

Across all paired comparisons, the improvement from thinking is consistently larger on SIMPLELISTQA than on COMPLEXLISTQA. This asymmetry suggests that thinking primarily helps models better organize and recall knowledge they already possess, but provides diminishing returns when the underlying retrieval challenge is more demanding. This further suggests that COMPLEXLISTQA tests a qualitatively different capability than simple list retrieval.

L Oracle RAG: Setup

This section describes how source pages were obtained and how context budgets were set for the Oracle RAG evaluation in Sec. 4.3.

Source pages. Each sample in LISTQA includes the Wikipedia URL used during annotation. We retrieve the full page content from a pre-indexed Wikipedia snapshot based on this URL, successfully matching context for 98% of samples (8,856 of 9,045). The remaining 2% lack matching content due to URL format variations and can be treated as a natural noise floor. The prompt template (App. F.3) instructs the model to answer *solely* from the provided context, using the same list-formatting instructions as the parametric evaluation. All other evaluation settings (metrics, parsing) are identical.

Context budget. Closed-source models have context windows large enough to hold every source page, so they always receive the full page. For each open-weight model, we convert its context window into a maximum number of source characters at roughly 3 characters per token: about 800K characters for the 256K-token windows of the Gemma 4 models, and 400K characters for the 128K-token windows of Magistral Small 24B and Ministral 3 8B/3B. This conservative conversion leaves headroom for the instruction prompt and generation. Models with smaller context windows may therefore not receive the full page when the source is unusually long.

M Timeless vs. Time-constrained Knowledge Retrieval

To understand how temporal properties affect performance, we split the dataset by whether questions include explicit year-range constraints (*time-constrained*, 70.1% of samples) or target historically immutable facts (*timeless*, 29.9%). Fig. 6 shows LLM-Judge-F1 for a representative subset of models across both splits.

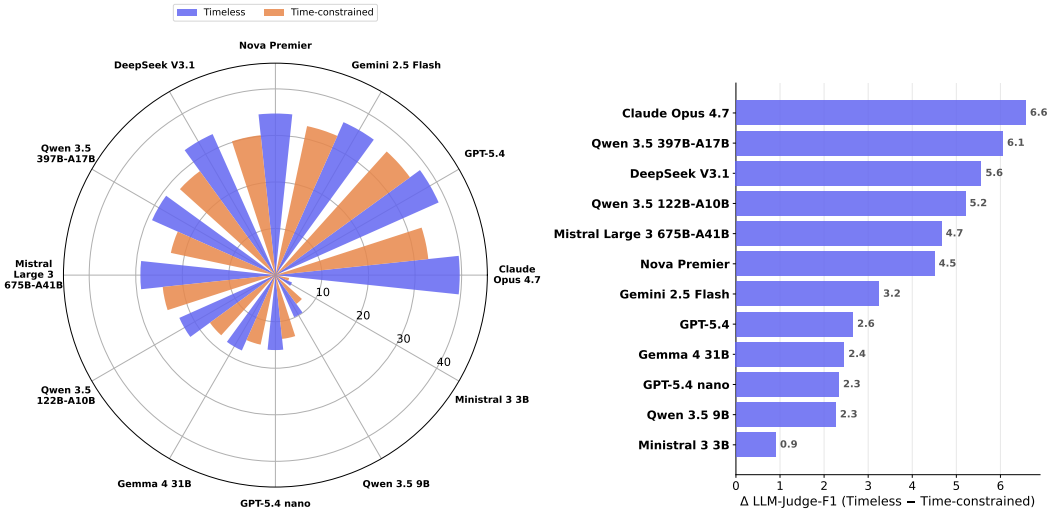


Figure 6: **Left:** LLM-Judge-F1 on timeless vs. time-constrained questions. Every model scores higher on timeless questions. **Right:** The gap (timeless − time-constrained), sorted by magnitude.

Every model without exception performs better on timeless questions than on time-constrained ones. This pattern holds across all standard-inference models evaluated, with the gap varying by model. The finding is consistent with the intuition that historically fixed facts are more likely to be well-represented and reinforced across training corpora, while time-bounded questions require recall of specific facts tied to particular periods: knowledge that may appear less frequently or less consistently in training data.

The gap is notably larger for some model families, suggesting relatively less coverage of temporally specific knowledge compared to their recall of stable historical facts. Other models show smaller gaps, indicating more uniform knowledge coverage across temporal categories.

This analysis validates a core design choice of LISTQA: by including both time-constrained and timeless questions, the benchmark can distinguish between models that have broad but shallow temporal coverage and those with deeper knowledge of specific time periods.

N Effect of List Length on Performance

To understand how the number of required list elements affects model performance, we split the dataset by the number of primary elements in the reference answer (ranging from 3 to 8) and compare scores on short lists (exactly 3 elements, 33% of samples) against longer lists (4 or more elements, 67%). Fig. 7 shows LLM-Judge-F1 for a representative subset of models across both splits.

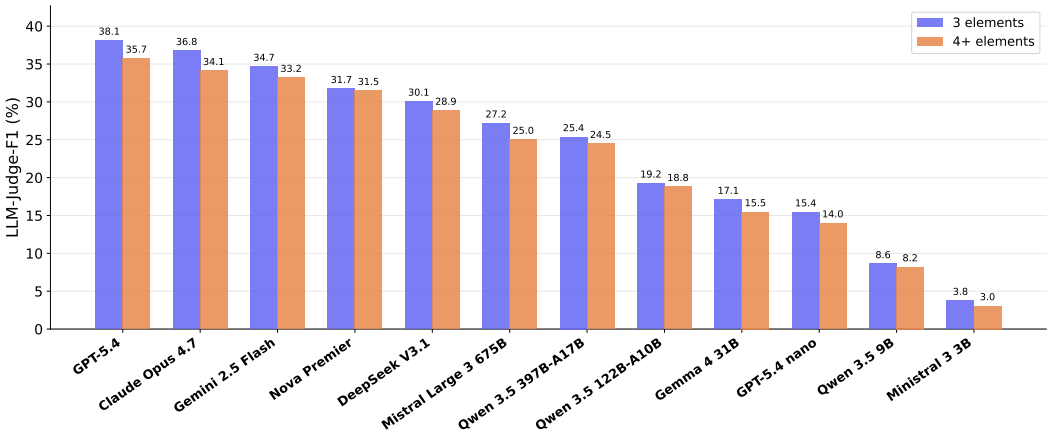


Figure 7: LLM-Judge-F1 on questions with exactly 3 reference elements vs. 4 or more elements in the reference answer. Every model performs better on shorter lists, though the gap is modest.

Every model scores higher on 3-element questions than on questions requiring 4 or more elements, consistent with the intuition that retrieving fewer facts from parametric memory is easier than retrieving many. However, the gap is notably small across all models (typically 1–3 percentage points), suggesting that list length alone is not a dominant difficulty factor in LISTQA. This contrasts with the much larger gaps observed between single-faceted and multi-faceted questions (Tab. 2), indicating that the structural complexity of sub-attributes poses a greater challenge than the sheer number of elements to retrieve.

Within the longer lists, we observe that performance is relatively stable across lengths 4 through 8. The absence of a strong monotonic decline suggests that once a model can retrieve 4 elements, the marginal difficulty of retrieving additional elements is modest. This finding validates the dataset's design choice of bounding list length between 3 and 8: the range is sufficient to test multi-element retrieval without introducing a confounding length penalty that would dominate the signal from knowledge and formatting capabilities.

O Performance by Topic Category

To examine whether certain knowledge domains are systematically easier or harder for models, we break down LLM-Judge-F1 by the eight topic categories in LISTQA. Fig. 8 shows two complementary views for a subset of models across multiple families: raw per-category scores (left) and each category's weighted contribution to the overall score (right), where the weight is the category's proportion of the dataset.

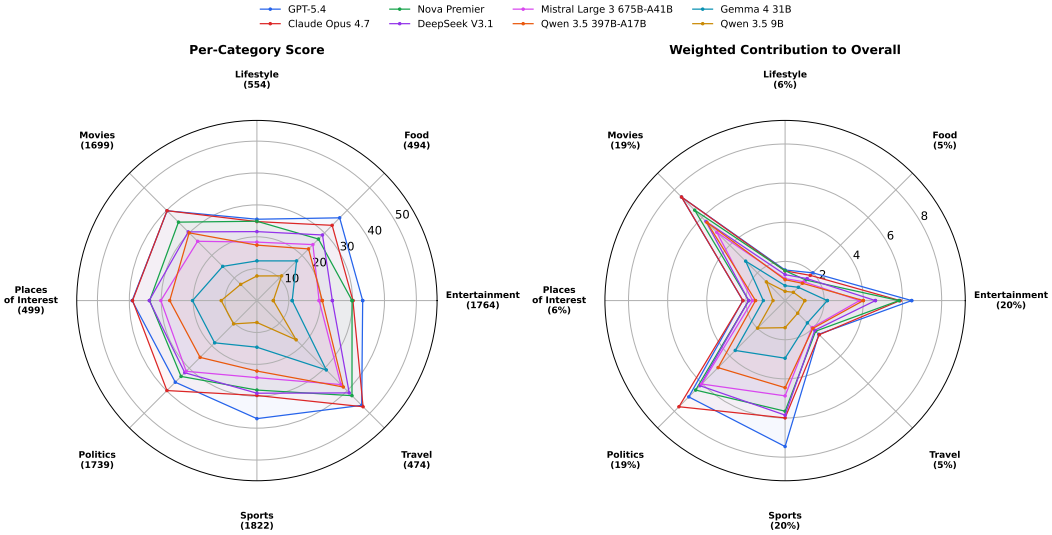


Figure 8: **Left:** LLM-Judge-F1 per category (sample counts in parentheses). **Right:** Weighted contribution to overall LLM-Judge-F1 score (category proportion in parentheses). The radar shape is consistent across models in both views, differing primarily in scale.

The raw scores (left) reveal that *Travel* is universally the easiest category and *Lifestyle* universally the hardest, with all models sharing a strikingly similar profile shape that differs only in scale. This consistency suggests that category difficulty is an intrinsic property of the dataset's knowledge distribution rather than a model-specific weakness.

The weighted view (right) tells a complementary story: the shape inverts. The four large categories: Sports, Entertainment, Politics, and Movies (each $\sim$19–20% of samples), dominate the overall score, while Travel and Lifestyle, despite being the easiest and hardest respectively, contribute minimally due to their small share ($\sim$5–6% each - See distribution in Fig. 2). This means the overall benchmark score is primarily determined by performance on the mid-difficulty categories that cluster in a narrow band, making it a stable aggregate measure not skewed by outlier categories at either extreme.

P Performance by Geographic Region

To assess whether models exhibit geographic knowledge biases, we group samples by the continent of their source Wikipedia content and compare LLM-Judge-F1 across regions. Fig. 9 shows raw per-continent scores (left) and weighted contributions to the overall score (middle and right).

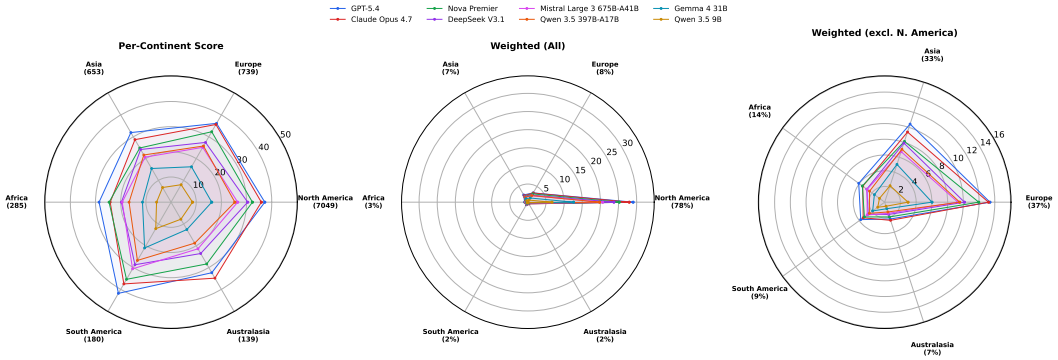


Figure 9: **Left:** LLM-Judge-F1 per continent (sample counts in parentheses). **Center:** Weighted contribution to overall score (continent proportion in parentheses). North America dominates the overall score due to its 78% share of the dataset. **Right:** Weighted contribution excluding North America (renormalized proportions).

The continent breakdown reveals a clear geographic hierarchy: models consistently perform best on South American and North American/European content, and worst on African and Asian content. This gap persists across all model families and scales, suggesting that training corpora systematically underrepresent factual knowledge about Africa and Asia relative to the Americas and Europe.

The weighted view (Fig. 9 (center)) makes the practical implication stark: North America alone accounts for 78% of the dataset, so the overall benchmark score is overwhelmingly determined by performance on US-centric content. The regions where models struggle most: Africa and Asia, contribute minimally to the aggregate despite revealing the largest knowledge gaps. This highlights a limitation of the current dataset composition and motivates future geographic rebalancing to ensure the benchmark measures global factual knowledge rather than primarily Western-centric recall.

When North America is excluded (Fig. 9 (right)), Europe and Asia together account for roughly 70% of the remaining weighted contribution, making the Europe–Africa performance gap the dominant signal. South America achieves the highest raw scores of any non-US region for most models, yet its small sample share limits its influence even in this reweighted view.

Q Effect of Instruction Tuning

To isolate the contribution of instruction tuning to LISTQA performance, we evaluate a few base (pre-trained, non-instruction-tuned) models using a minimal chat template in a few-shot setting (prompt template in App. F.4), and compare them against their instruction-tuned (IT) counterparts evaluated zero-shot. For IT models, we additionally report results with thinking enabled at inference.

This asymmetry in prompting is deliberate. Base models have had little exposure to instruction following, so a zero-shot prompt would conflate missing knowledge with a failure to understand the requested output format. Few-shot prompting is the minimal condition that reliably elicits list-formatted output from a base model, allowing its factual knowledge to be measured. IT models, by contrast, acquire format-following during post-training and are evaluated in the same zero-shot setting as the main results (Sec. 4.1). The comparison therefore isolates the effect of training stage rather than prompting protocol.

Table 14: Base vs. instruction-tuned (IT) performance. Base models are evaluated few-shot; IT models are evaluated zero-shot, with and without thinking enabled at inference. All values are percentages. All = Overall, S = SIMPLELISTQA, C = COMPLEXLISTQA.

		LLM-Judge-F1 ↑			Format ↑		
Model	**Setting**	**All**	**S**	**C**	**All**	**S**	**C**
Gemma 4 31B	Base	9.1	12.6	4.0	51.7	51.7	51.8
	IT	16.0	22.0	7.3	54.3	45.8	66.7
	IT + Think	18.7	25.3	9.0	74.9	73.9	76.5
Qwen 3.5 9B	Base	4.9	7.2	1.5	55.9	54.6	57.8
	IT	8.3	12.0	2.9	62.0	65.5	56.8
	IT + Think	11.4	16.1	4.4	87.2	87.0	87.5
Qwen 3.5 35B-A3B	Base	8.6	12.0	3.6	52.0	49.8	55.4
	IT	13.4	18.5	5.9	51.7	37.0	73.2
	IT + Think	18.2	24.4	9.0	90.1	89.6	91.0

Results. Tab. 14 reports LLM-Judge-F1 and Format-Correctness across the base, IT, and IT with thinking variants. Instruction tuning substantially improves LLM-Judge-F1 over the base model across all three models. Enabling thinking at inference on the same IT checkpoint adds a further improvement, consistent across model scales. The two stages contribute differently: IT improves the model's ability to produce structured, relevant responses, while thinking further enhances factual retrieval and output organization at inference time.

It is interesting to see that base models, evaluated in a few-shot setting, already achieve moderate format scores. This demonstrates that even without instruction tuning, these models can learn list-formatting conventions from in-context examples alone. Instruction tuning provides only marginal further improvement in format compliance. Enabling thinking on the IT checkpoint, however, produces a dramatic jump in format scores across all three models, suggesting that the reasoning process at inference is particularly effective at helping models interpret formatting instructions and produce the correct list structure.

Table 15: Format error distribution (% of 9,045 samples) for Base vs. IT models.

Model	**Setting**	**Perfect**	**Wrong Facet**	**Too Few**	**No List**	**Mixed**
Gemma 4 31B	Base	43.6	23.3	3.6	13.9	15.6
	IT	53.9	14.4	29.0	2.0	0.7
Qwen 3.5 9B	Base	37.7	20.5	6.3	1.6	34.0
	IT	55.4	19.8	11.0	2.8	11.0
Qwen 3.5 35B-A3B	Base	42.5	23.0	10.2	5.8	18.6
	IT	49.8	26.0	20.2	0.6	3.5

Format error analysis. Tab. 15 breaks down format errors across the base and IT variants. The format error profiles reveal qualitatively different failure modes between base and IT models. Base models produce a higher rate of *mixed facet type* errors, i.e., responses mixing sub-bullets and flat elements inconsistently, suggesting that without instruction tuning, models lack a coherent formatting strategy. Instruction tuning largely eliminates mixed-facet errors but introduces a shift toward *too few elements*, suggesting that IT potentially makes models more conservative, sometimes undershooting the minimum list length in favor of more structured output.

R Validation Against Human Judgment

R.1 Human Evaluation of LLM-Judge-F1

We selected about 200 questions stratified across the 8 categories and both question types, and collected responses from three models spanning the score range: GPT-5.4, Qwen 3.5 122B-A10B, and Claude Opus 4.7. Human graders labeled each of the resulting 3,313 predicted list elements (571 sample–model pairs) as CORRECT or INCORRECT using the same criteria as the judge prompt (App. F.5). Human precision, recall, and F1 were computed exactly as for LLM-Judge-F1.

Table 16: Sample-level agreement between LLM-Judge-F1 (GPT-OSS-120B) and human F1, by model.

Model	**N**	**Elem.**	**Hum. P**	**Hum. R**	**Hum. F1**	**Judge F1**	**Pearson r**	**Spearman ρ**
GPT-5.4	200	1,077	0.346	0.372	0.349	0.328	0.974	0.974
Qwen 3.5 122B-A10B	191	1,330	0.155	0.191	0.164	0.150	0.951	0.922
Claude Opus 4.7	180	906	0.385	0.383	0.374	0.346	0.958	0.951
Overall	571	3,313	–	–	–	–	**0.965**	**0.956**

Table 17: Sample-level agreement between LLM-Judge-F1 and human F1, by topic category and question type.

	N	**Elem.**	**Hum. F1**	**Judge F1**	**Pearson r**	**Spearman ρ**
By category						
Entertainment	75	380	0.293	0.269	0.958	0.974
Food	73	462	0.348	0.322	0.954	0.954
Lifestyle	75	431	0.168	0.171	0.997	0.974
Movies	73	456	0.338	0.318	0.981	0.968
Places of Interest	70	410	0.351	0.298	0.924	0.895
Politics	65	439	0.301	0.274	0.965	0.937
Sports	70	323	0.294	0.288	0.970	0.947
Travel	70	412	0.275	0.256	0.983	0.978
By question type						
SIMPLELISTQA	285	1,674	0.406	0.388	0.974	0.973
COMPLEXLISTQA	286	1,639	0.185	0.160	0.942	0.921

Table 18: Element-level agreement with human labels ($N = 3{,}313$).

Comparison	**Agreement**	**Cohen's κ**
Human vs. LLM-Judge-F1	92.8%	0.793
Human vs. ROUGE-1 ≥ 0.5	89.3%	0.735

Results. LLM-Judge-F1 correlates strongly with human F1 (Pearson $r = 0.965$, Spearman $\rho = 0.956$; Tab. 16). Agreement is high for all three models ($r \geq 0.95$), and LLM-Judge-F1 is slightly below human F1 for each, so the metric is mildly conservative rather than inflating scores. Correlation is also stable across all 8 categories ($r \geq 0.92$) and both question types (Tab. 17). It is slightly lower on COMPLEXLISTQA than on SIMPLELISTQA, but alignment on the harder hierarchical questions remains strong. At the element level (Tab. 18), the LLM judge agrees with humans more often than ROUGE thresholding ($\kappa = 0.793$ vs. 0.735), so semantic grading adds value beyond lexical overlap.

R.2 Validating the ROUGE-Based Matching Step

LLM-Judge-F1 uses ROUGE-1 overlap only to *align* predicted and reference elements; correctness is then graded semantically by the judge. Lexical overlap is a suitable alignment signal for LISTQA because list elements are short entities or phrases (names, dates, numbers, titles), where paraphrase is rare.

Alignment accuracy. We manually verified the 3,313 alignments produced by the Jonker–Volgenant assignment in App. R.1. Of these, 3,303 were correct (99.7%).

Sensitivity to the matching signal. We replaced ROUGE-1 with cosine similarity over `all-MiniLM-L6-v2` sentence embeddings and re-scored all 9,045 samples for two models (Tab. 19). Over 90% of samples receive identical scores, and aggregate LLM-Judge-F1 changes by less than 0.002. When the two strategies disagree, the reassigned element is typically incorrect under either alignment, so the verdict rarely changes. Embedding-based matching therefore adds cost without measurable benefit for short factual elements.

Table 19: LLM-Judge-F1 under ROUGE-1 vs. embedding-based matching (9,045 samples). **% Identical**: samples with the same score under both strategies.

Model	ROUGE-1	Embedding	% Identical
Claude Opus 4.6	0.346	0.3451	91.3
Claude Sonnet 4.5	0.253	0.2517	94.1

R.3 Judge Model Agreement Analysis

A key concern with LLM-as-judge evaluation is whether the metric is robust to the choice of judge model, and whether practitioners can use smaller, open-weight models as affordable alternatives. To assess this, we rescore all 9,045 samples from a single evaluated model (Claude Opus 4.6 under standard inference) using three open-weight reasoning models: Qwen 3.5 27B, Qwen 3.5 9B, and Qwen 3.5 4B, alongside the default GPT-OSS-120B judge used throughout the paper. This produces four independent LLM-Judge-F1 scores per sample, enabling a systematic analysis of inter-judge agreement.

We report two complementary agreement metrics, each capturing a different aspect of reliability.

Mean Absolute Difference (MAD). For each pair of judges, MAD computes the average absolute score difference across all samples:

$$\mathrm{MAD}(A, B) = \frac{1}{N} \sum_{i=1}^{N} |s_i^A - s_i^B| \tag{3}$$

where s_i^A and s_i^B are the LLM-Judge-F1 scores assigned by judges A and B to sample i. MAD is directly interpretable in the same units as the metric: a MAD of 0.02 means scores differ by 0.02 points on average. We additionally report the mean signed difference $\frac{1}{N}\sum(s_i^A - s_i^B)$ to distinguish systematic bias (one judge consistently more lenient) from random disagreement.

Spearman rank correlation (ρ). Spearman correlation measures whether two judges produce the same *ranking* of samples, regardless of the absolute scores. Each judge's scores are replaced by their ranks (with ties receiving the average rank), and Pearson correlation is computed on the ranks. Values range from -1 (perfectly opposite rankings) to $+1$ (identical rankings). Spearman is robust to systematic offsets, i.e., if one judge scores consistently higher but preserves the same ordering, ρ remains high.

Table 20: Basic statistics for each judge model. Scores are LLM-Judge-F1 (0–1) across 9,045 samples from Claude Opus 4.6 under standard inference.

Judge Model	LLM-Judge-F1	Zero-scored (%)
GPT-OSS-120B	0.346	39.2
Qwen 3.5 27B	0.362	37.8
Qwen 3.5 9B	0.345	39.7
Qwen 3.5 4B	0.354	38.5

Table 21: Pairwise agreement between judge models. MAD: mean absolute difference (lower = more agreement). Signed Diff: mean signed difference (positive = first judge scores higher). Spearman ρ: rank correlation (higher = more agreement).

Judge Pair	MAD ↓	Signed Diff	Spearman ρ ↑
GPT-OSS vs. Qwen 3.5 27B	0.022	−0.015	0.973
GPT-OSS vs. Qwen 3.5 9B	0.024	+0.002	0.966
GPT-OSS vs. Qwen 3.5 4B	0.022	−0.008	0.972

Results. All judges exhibit strong agreement (see Tabs. 20–21). Scores differ by only 0.02–0.024 points on average (MAD), and all pairwise Spearman correlations exceed 0.96, indicating near-identical sample rankings (see Tab. 21). Notably, even the smallest judge (Qwen 3.5 4B) achieves agreement on par with the 27B variant, with a Spearman ρ of 0.972 and MAD of 0.022. The score distributions are also closely matched: all judges assign zero scores to a similar fraction of samples (37.8–39.7%).

These results demonstrate that open-weight models as small as 4B parameters can serve as reliable judges for the LISTQA evaluation, producing both similar absolute scores and nearly identical model rankings compared to the 120B default judge. Practitioners who prefer a lighter-weight evaluation pipeline can substitute any of the Qwen 3.5 models with minimal impact on conclusions, keeping in mind the small random noise factor (MAD $\approx$ 0.02) when comparing models with very close scores.

R.4 Judge Repeatability

To measure judge repeatability directly, we re-score three models five times each with GPT-OSS-120B at temperature 0.7. Tab. 22 compares the reported score with the mean and range across the five runs, together with the percentage of samples that receive the same score in all five runs.

Table 22: Repeatability of LLM-Judge-F1 across five independent re-scoring runs with GPT-OSS-120B (temperature 0.7). **Reported**: score in Tab. 2. **% Identical**: percentage of the 9,045 samples that receive the same score in all five runs.

Model	Reported	Mean (5 runs)	Min / Max	% Identical
Claude Opus 4.6	0.3460	0.3469	0.3465 / 0.3473	86.3
Claude Sonnet 4.6	0.2800	0.2803	0.2794 / 0.2811	88.0
Claude Sonnet 4.5	0.2530	0.2531	0.2526 / 0.2536	89.9

LLM-Judge-F1 is highly reproducible. Across runs, scores vary by at most 0.0017, far less than the gaps between models in Tab. 2, and the large majority of samples receive the same score in all five runs. Together with App. R.3, this shows that the metric is stable both when the same judge is re-run and when a different judge is used.

S Token Usage and Generation Efficiency

We analyze the relationship between generation cost (measured in tokens) and factual accuracy across all evaluated models. For models where we are able to separate reasoning tokens from output tokens, we report both components. For others, we do not provide this separation, i.e., reasoning tokens are included in the output token count. Total generation cost is defined as reasoning tokens + output tokens for all models.

S.1 Generation Cost vs. Accuracy

Fig. 10 plots each non-thinking model's average total generation tokens against its overall LLM-Judge-F1 score. The highest-scoring models achieve the best accuracy with moderate output lengths, while several open-weight models generate substantially more tokens for lower scores. Generation verbosity is not a proxy for factual knowledge: weaker models tend to produce longer, less precise outputs rather than concise correct answers.

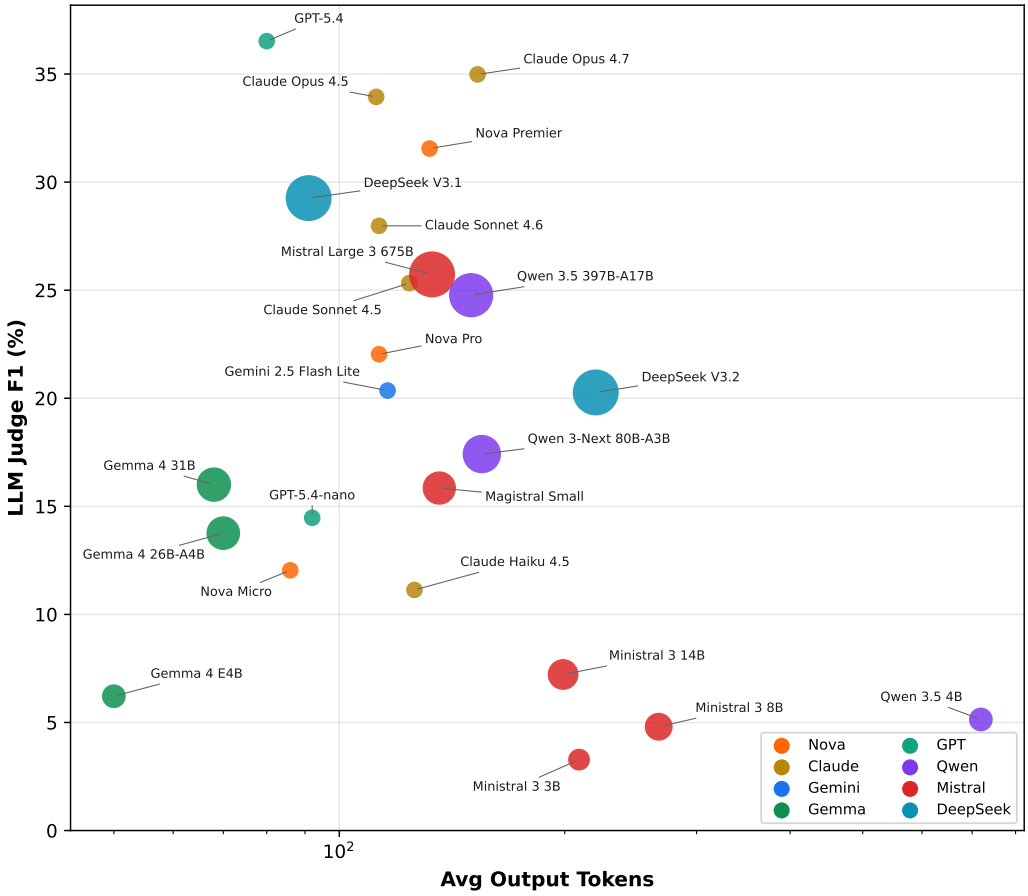


Figure 10: *LLM-Judge-F1* vs. average output tokens for non-thinking models. Marker color indicates model family; marker size is proportional to parameter count for open-weight models (fixed size for closed-source). Higher and further left is better.

Fig. 11 extends this analysis to models with both thinking and non-thinking configurations. Thinking universally shifts models rightward (more tokens) and upward (higher F1 score), but the magnitude of both shifts varies dramatically across architectures, a pattern we quantify next.

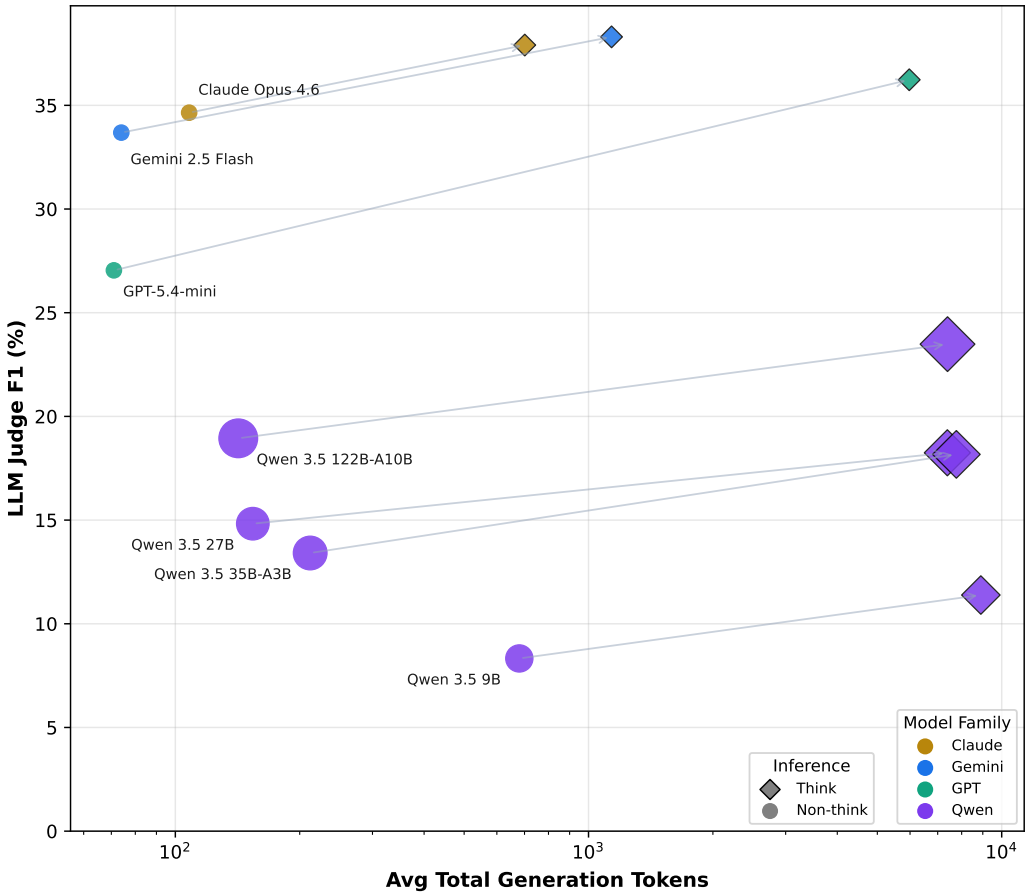


Figure 11: *LLM-Judge-F1* vs. average total generation tokens per sample for models with both thinking and non-thinking configurations. Arrows point from non-thinking (circle) to thinking (diamond) variants.

S.2 The Cost of Thinking

Fig. 12 quantifies the thinking trade-off across paired model configurations. Thinking improves LLM-Judge-F1 for every model, but the token cost varies by orders of magnitude across families. Some models achieve meaningful accuracy gains at relatively modest cost multipliers, while others pay far more in generation tokens without proportionally larger gains. The scatter of cost ratio against accuracy delta reveals that models that spend the most additional tokens on thinking do not necessarily gain the most accuracy, suggesting diminishing returns from extended reasoning on this task.

S.3 Reasoning Budget Allocation

For the models for which we have explicit reasoning token separation, Fig. 13 shows how the generation budget is split between thinking and output. The allocation strategies differ dramatically: Gemini 3 Flash, evaluated in its default high-reasoning mode, dedicates nearly all tokens to reasoning and produces the highest absolute token count of the three Gemini models, while GPT-5.4-mini splits roughly evenly between reasoning and output. Gemini 3.1 Pro (low-reasoning mode) and Gemini 2.5 Flash (dynamic-reasoning mode) fall between these extremes. Gemini 3.1 Pro achieves the best accuracy while generating the least number of tokens among the four. This suggests that for factual list retrieval, the model's underlying knowledge capacity may matter more than the amount of inference-time reasoning it is allowed to perform. An open question is how these reasoning tokens

are allocated between knowledge retrieval and instruction following (two capabilities that LISTQA is specifically designed to disentangle) and whether the balance shifts across model architectures.

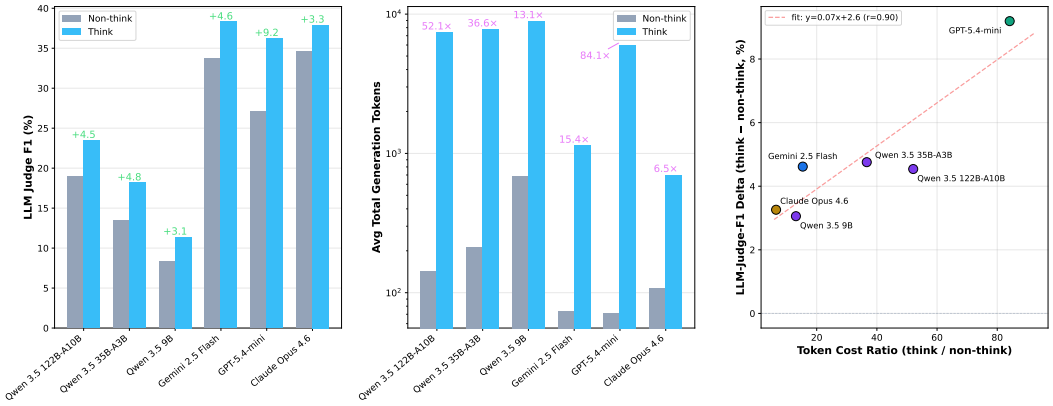


Figure 12: Comparison of thinking vs. non-thinking inference. **Left:** LLM-Judge-F1; annotations show the delta. **Center:** Average total generation tokens; annotations show the cost multiplier. **Right:** Cost ratio vs. LLM-Judge-F1 delta with best-fit line.

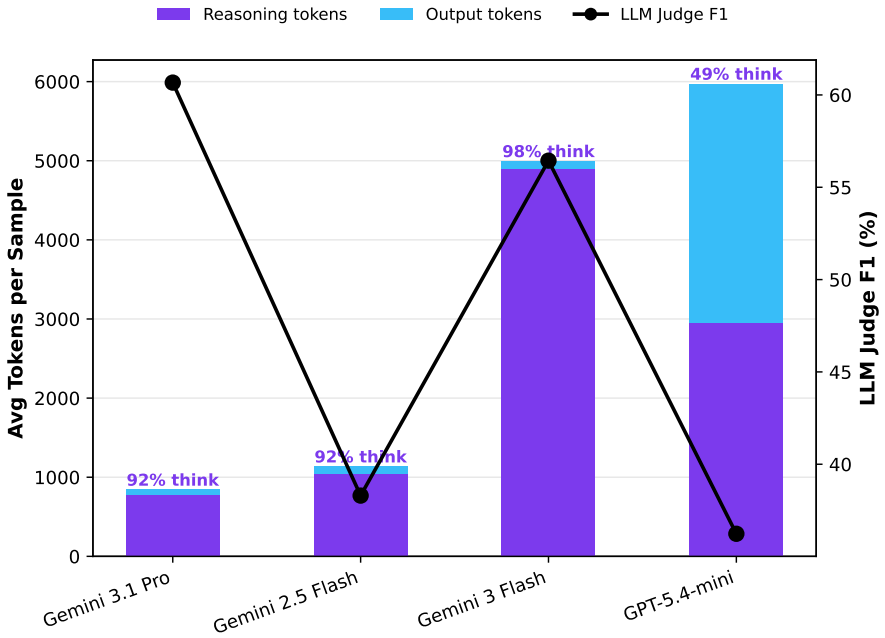


Figure 13: Reasoning vs. output token allocation for models with explicit token separation. Stacked bars show average tokens per sample; percentages indicate the fraction spent on reasoning. The black line shows LLM-Judge-F1.

S.4 Single-Faceted vs. Multi-Faceted Generation Patterns

Figs. 14 and 15 show generation cost and accuracy for single-faceted and multi-faceted prompts within each model family. Multi-faceted questions consistently elicit more tokens across all families, as expected given the relatively complex nature of the task. However, the accuracy gap between prompt types varies by family: some models maintain relatively stable accuracy despite the increased generation cost, while others show larger accuracy drops on multi-faceted questions that are not compensated by the additional generation.

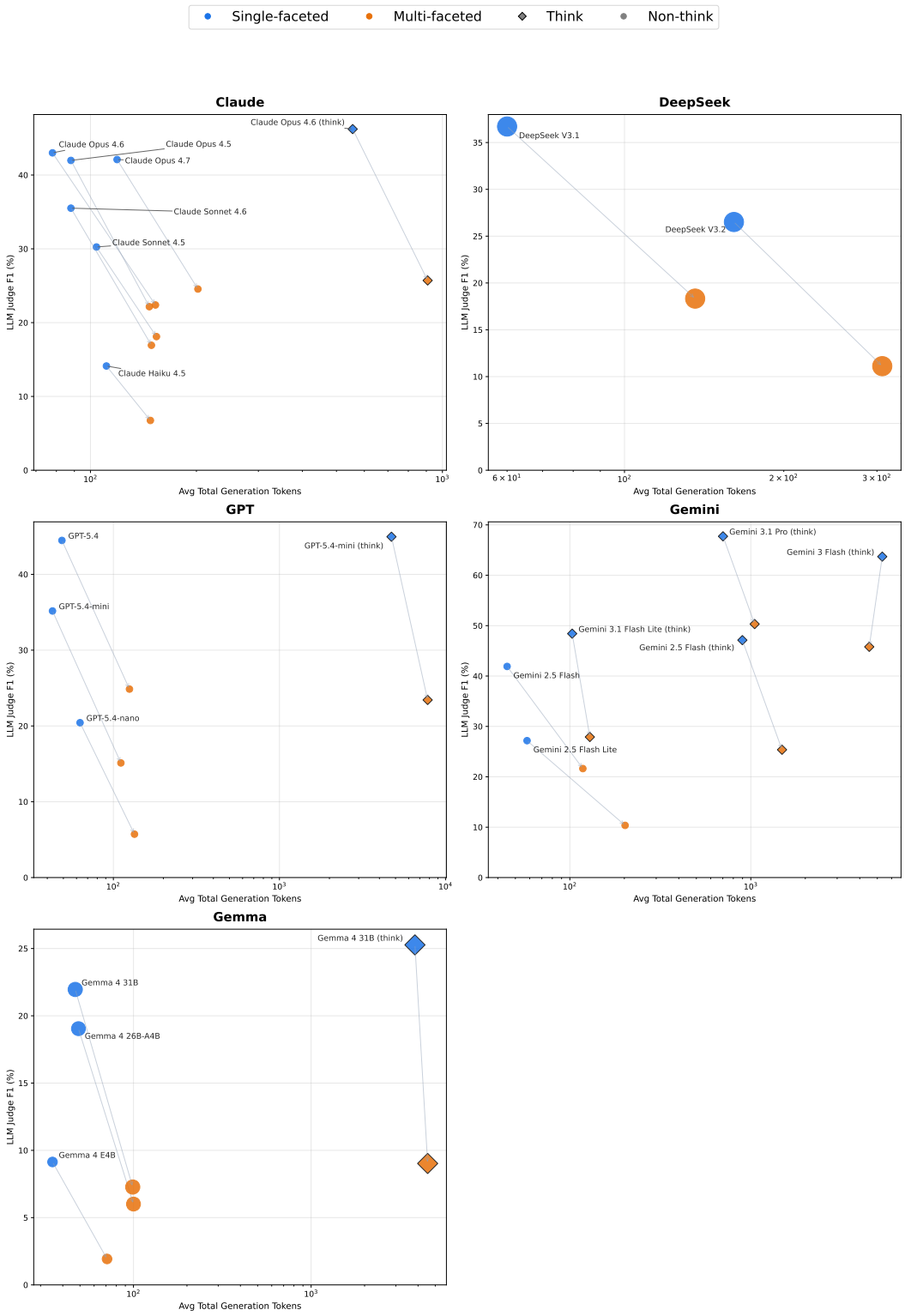


Figure 14: Generation cost vs. LLM-Judge-F1 by prompt type, grouped by model family. blue = single-faceted, orange = multi-faceted. Arrows connect the same model's single-faceted (source) and multi-faceted (target) results. Diamond markers indicate thinking-enabled models.

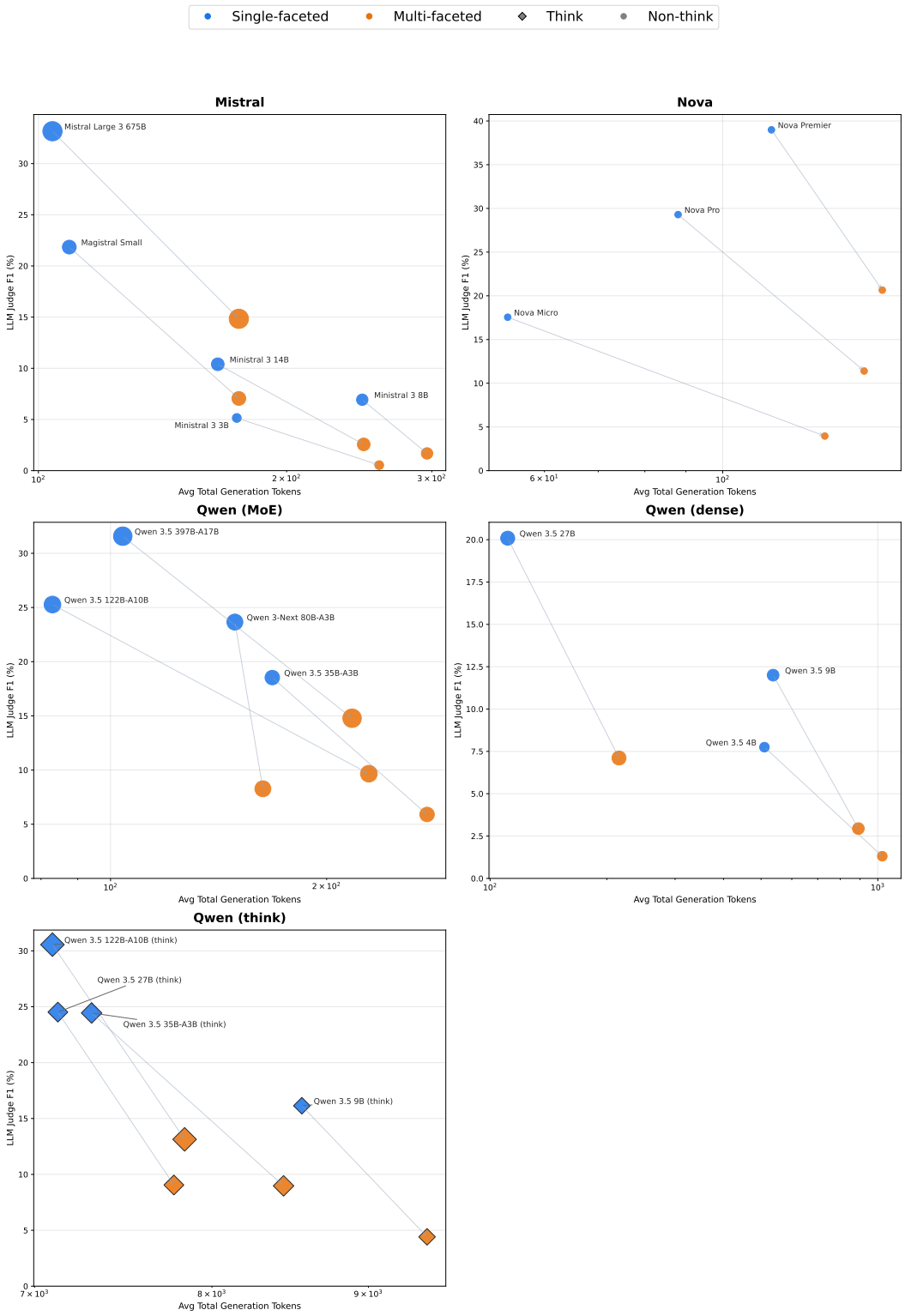


Figure 15: Generation cost vs. LLM-Judge-F1 by prompt type, grouped by model family. blue = single-faceted, orange = multi-faceted. Arrows connect the same model's single-faceted (source) and multi-faceted (target) results. Diamond markers indicate thinking-enabled models.

S.5 Per-Sample Score Distributions

Fig. 16 shows the per-sample LLM-Judge-F1 distribution for each model configuration. Most models exhibit a zero-inflated distribution: a large mass of samples receive zero scores, indicating complete failure to produce correctly formatted list elements, while the remaining samples are spread across a range of partial scores. The strongest thinking models (Gemini 3.1 Pro, Gemini 3 Flash) stand out with a genuinely bimodal distribution, showing substantial mass at both ends, suggesting these models tend to either solve a sample well or miss it entirely with less middle ground. The gap between single-faceted and multi-faceted mean F1 is visible for every model, reinforcing the consistent difficulty gap observed in the main results.

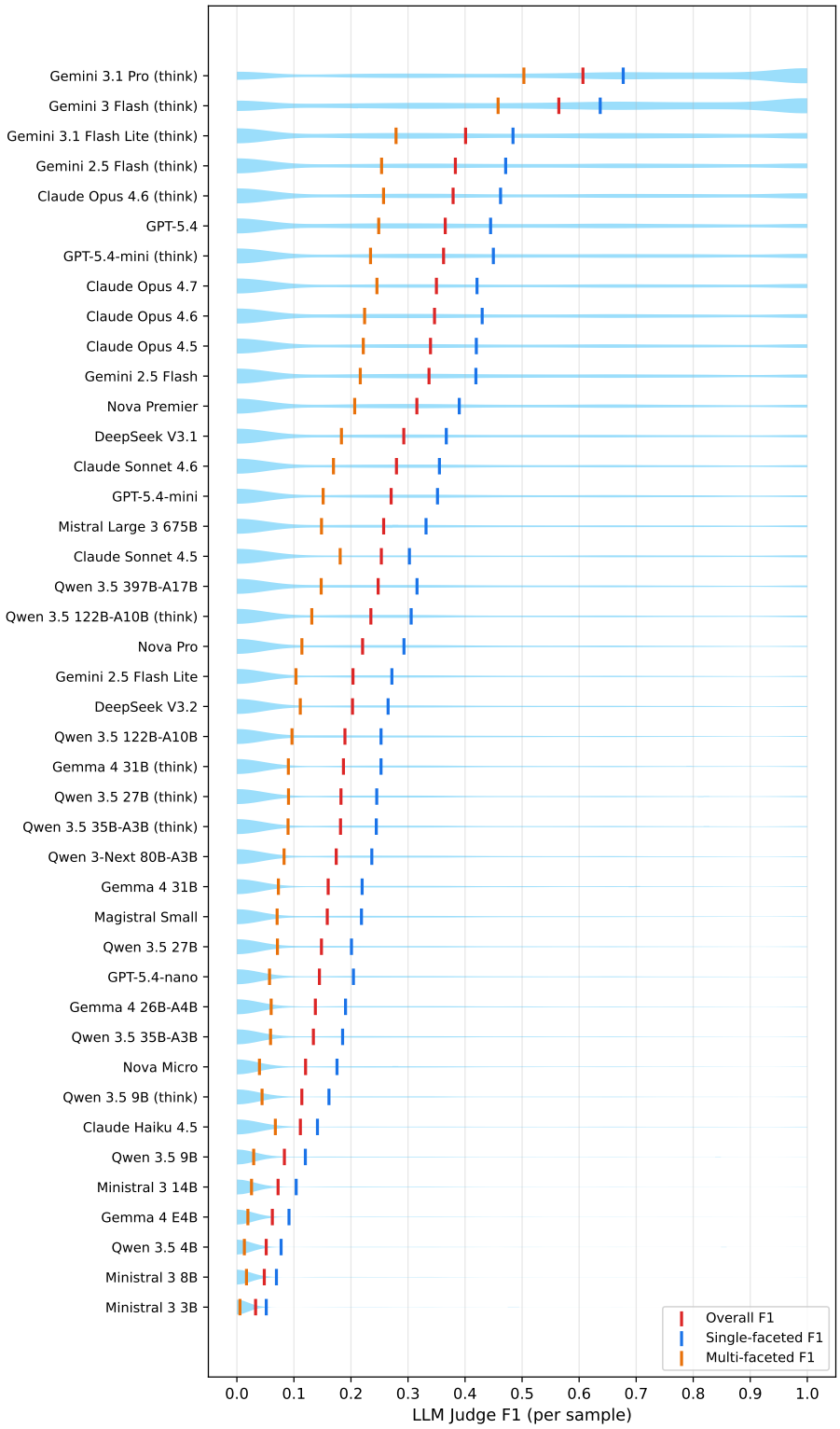


Figure 16: Per-sample LLM-Judge-F1 distributions, sorted by overall F1 descending. Vertical markers indicate overall (red), single-faceted (blue), and multi-faceted (orange) mean *LLM-Judge-F1*.

T Qualitative Examples

We present selected examples that illustrate key phenomena observed in the quantitative results. For each example, we show the question, reference answer, and model outputs alongside their LLM-Judge-F1 and Format-Correctness scores.

T.1 Knowledge vs. Format Trade-off

The following single-faceted question asks for a flat list of Academy Awards. Three models produce qualitatively different responses, illustrating how factual accuracy and format compliance can diverge.

Question: What are the Academy Awards won by the "All Quiet on the Western Front" film as of 2024?

Reference Answer (4 elements, single-faceted):

```
1.  Best International Feature Film
2.  Best Production Design
3.  Best Cinematography
4.  Best Original Score
```

Qwen 3.5 35B-A3B *LLM-Judge-F1 = 0.89* *Format = 0.00*

```
1.  Academy Award for Best International Feature Film
* Category:  International Feature Film
* Sub-Element:  Germany
2.  Academy Award for Best Cinematography
* Category:  Cinematography
* Sub-Element:  James Friend
3.  Academy Award for Best Original Score
* Category:  Original Score
* Sub-Element:  Volker Bertelmann
4.  Academy Award for Best Production Design
* Category:  Production Design
5.  Academy Award for Best Visual Effects
* Category:  Visual Effects
```

Analysis: Retrieves all four correct awards but hallucinates a fifth (Best Visual Effects), and produces multi-faceted sub-bullets for a single-faceted question (Format = 0). LLM-Judge-F1 reflects precision = 4/5 and recall = 4/4.

Qwen 3.5 122B-A10B *LLM-Judge-F1 = 0.73* *Format = 1.00*

```
1.  Best Picture
2.  Best Director
3.  Best International Feature Film
4.  Best Cinematography
5.  Best Production Design
6.  Best Original Score
7.  Best Sound
```

Analysis: Correct flat format (Format = 1.00), but hallucinates Best Picture, Best Director, and Best Sound, awards the film did not win.

Qwen 3.5 4B *LLM-Judge-F1 = 0.00* *Format = 1.00*

```
1.  Best Director for Lewis Milestone
2.  Best Motion Picture for Lewis Milestone
3.  Best Art Direction for Carl D. Smith
```

Analysis: Correct flat format (Format = 1.00), but every element is factually wrong. These awards are not from the film the question refers to. LLM-Judge-F1 = 0 despite perfect formatting.

This example illustrates that knowledge retrieval and instruction following are partially independent capabilities: Qwen 3.5 35B-A3B achieves the best factual score but fails format compliance, Qwen 3.5 4B produces perfect format but entirely wrong facts, and Qwen 3.5 122B-A10B balances both but hallucinates additional awards.

T.2 Multi-Faceted Difficulty

The following multi-faceted question requires retrieving stadiums satisfying multiple constraints (capacity, location, time range) and producing hierarchical sub-attributes for each. Even some of the proprietary models find it challenging to retrieve all elements correctly.

Question: Which sports stadiums and arenas with a capacity greater than 15,000 were opened in California between 2017 and 2024, and what year was it opened, which city is it in, what is the capacity, what type is it, and who is it home to?

Reference Answer (5 elements, multi-faceted):

```
1.  BMO Stadium
* Opened:  2018
* City:  Los Angeles
* Capacity:  22,000
* Type:  Soccer
* Home to:  Los Angeles FC and Angel City FC
2.  Chase Center
* Opened:  2019
* City:  San Francisco
* Capacity:  18,064
* Type:  Arena
* Home to:  Golden State Warriors and Valkyries
3.  SoFi Stadium
* Opened:  2020
* City:  Inglewood
* Capacity:  70,240
* Type:  Multi-purpose
* Home to:  LA Chargers and LA Rams
4.  Snapdragon Stadium
* Opened:  2022
* City:  San Diego
* Capacity:  35,000
* Type:  Football
* Home to:  San Diego Aztecs, SD FC, SD Wave FC
5.  Intuit Dome
* Opened:  2024
* City:  Inglewood
* Capacity:  18,000
* Type:  Arena
* Home to:  Los Angeles Clippers
```

Claude Opus 4.6 *LLM-Judge-F1 = 0.5* *Format = 1.00*

```
1.  Chase Center
* Year Opened:  2019
* City:  San Francisco
* Capacity:  ~18,064
* Type:  Arena
* Home To:  Golden State Warriors (NBA)
2.  SoFi Stadium
* Year Opened:  2020
* City:  Inglewood
* Capacity:  ~70,240
* Type:  Stadium
* Home To:  Los Angeles Rams (NFL), Los Angeles Chargers (NFL)
3.  Intuit Dome
* Year Opened:  2024
* City:  Inglewood
* Capacity:  ~18,000
* Type:  Arena
* Home To:  LA Clippers (NBA)
```

Analysis: Correct format and accurate sub-attributes for the 2/3 stadiums it retrieves (Chase Center misses one home team), but misses BMO Stadium and Snapdragon Stadium entirely.

Qwen 3.5 35B-A3B *LLM-Judge-F1 = 0.33* *Format = 1.00*

```
1. SoFi Stadium
* Year Opened: 2020
* City: Inglewood
* Capacity: 70,240
* Type: Multi-purpose stadium
* Home To: Los Angeles Rams and Los Angeles Chargers
2. Intuit Dome
* Year Opened: 2024
* City: Inglewood
* Capacity: 18,000
* Type: Basketball arena
* Home To: Los Angeles Clippers
3. BMO Stadium
* Year Opened: 2018
* City: Carson
* Capacity: 22,000 (expandable to 35,000)
* Type: Soccer stadium
* Home To: Los Angeles FC
4. Angel Stadium
* Year Opened: 2018 (Refurbishment/Expansion
completed, original 1966)
* Correction: The only newly constructed
stadiums/arenas >15,000 capacity opened in CA
between 2017-2024 are SoFi Stadium (2020),
Intuit Dome (2024), and BMO Stadium (2018).
[repeats elements 1-3 again]
```

Analysis: Retrieves 3 correct stadiums but then hallucinates Angel Stadium, self-corrects mid-output, and repeats the list, producing a malformed response. The parser extracts the initial elements, yielding LLM-Judge-F1 = 0.33. This illustrates how smaller models can "think aloud" in ways that degrade structured output.

Qwen 3.5 9B *LLM-Judge-F1 = 0.29* *Format = 0.00*

```
1. SoFi Stadium
* Year Opened: 2020
* City: Inglewood
* Capacity: 70,240
* Type: Multi-purpose stadium
* Home To: Los Angeles Rams, Los Angeles Chargers
2. Intuit Dome
* Year Opened: 2024
* City: Inglewood
* Capacity: 18,000
* Type: Indoor arena
* Home To: Los Angeles D-Fenders (G League),
Los Angeles Forum (NBA G League),
occasional NBA teams
```

Analysis: Retrieves only 2 of 5 stadiums and hallucinates the home teams for Intuit Dome (the actual team is the LA Clippers). Format = 0 because only 2 elements are produced, below the 3-element minimum.

T.3 Effect of Thinking

We compare the same model (Gemini 2.5 Flash) with and without thinking enabled on a multi-faceted question requiring specific player names, colleges, and student status. This example illustrates a case where thinking substantially improves factual retrieval, though even with thinking the model does not achieve full accuracy.

Question: Who made up the Midwest Regional Team for the 2018 NCAA tournament, what colleges were they from, and what was their student status at the time?

Reference Answer (6 elements, multi-faceted):

```
1.  Malik Newman (Most Outstanding Player)
* College:  Kansas
* Student status:  Sophomore
2.  Trevon Duval
* College:  Duke
* Student status:  Freshman
3.  Gabe DeVoe
* College:  Clemson
* Student status:  Senior
4.  Gary Trent Jr.
* College:  Duke
* Student status:  Freshman
5.  Marvin Bagley III
* College:  Duke
* Student status:  Freshman
6.  Devonte' Graham
* College:  Kansas
* Student status:  Senior
```

Gemini 2.5 Flash (no-think) *LLM-Judge-F1 = 0.00* *Format = 1.00*

```
1.  Kansas Jayhawks
* College:  University of Kansas
* Student Status:  Undergraduate
2.  Duke Blue Devils
* College:  Duke University
* Student Status:  Undergraduate
3.  Michigan State Spartans
* College:  Michigan State University
* Student Status:  Undergraduate
4.  Clemson Tigers
* College:  Clemson University
* Student Status:  Undergraduate
```

Analysis: Lists team names instead of individual players and uses generic "Undergraduate" for student status, a complete misinterpretation of the question. Despite perfect format compliance (Format = 1.00), every element is factually wrong, yielding LLM-Judge-F1 = 0.

Gemini 2.5 Flash (think) *LLM-Judge-F1 = 0.73* *Format = 1.00*

Reasoning trace (abbreviated):

```
I need to dig up the info on the 2018 NCAA Tournament Midwest Regional team, specifically the
All-Tournament Team.  [...]  These players come to mind:  Malik Newman (Kansas), Devonte' Graham
(Kansas), Marvin Bagley III (Duke), Gary Trent Jr. (Duke), Grayson Allen (Duke).  Now, to get the
student status [...]  Malik Newman:  redshirt sophomore, having transferred from Mississippi State.
Devonte' Graham:  played for Kansas from 2014-2018, so he was a senior.  Marvin Bagley III: one season
at Duke, so freshman.  [...]
```

Final output:

```
1.  Malik Newman
* College:  Kansas
* Student Status:  Redshirt Sophomore
2.  Devonte' Graham
* College:  Kansas
* Student Status:  Senior
3.  Marvin Bagley III
* College:  Duke
* Student Status:  Freshman
4.  Gary Trent Jr.
* College:  Duke
* Student Status:  Freshman
5.  Grayson Allen
* College:  Duke
* Student Status:  Senior
```

Analysis: Retrieves 4 of 6 correct players (Newman, Graham, Bagley III, Trent Jr.) with accurate sub-attributes. Hallucinates Grayson Allen and misses Duval and DeVoe. Correct hierarchical format throughout.

In this example, thinking enables the model to correctly interpret the question as asking for individual players rather than teams. However, even with thinking, the model retrieves only 4 of 6 correct players and hallucinates one other. Both variants produce well-formatted hierarchical lists, suggesting that the difference lies in knowledge retrieval rather than instruction following.

T.4 Ordering Errors

For the 14.6% of questions requiring a specific sequence, we additionally compute *LLM-Judge-F1-Ordered*, which matches elements positionally rather than via optimal assignment. The following example asks for retired jersey numbers in order of ceremony date, a sequence that differs from numerical order.

Question: In ascending order of ceremony date, list the first five numbers retired by the Houston Rockets basketball team.

Reference Answer (5 elements, single-faceted, ordered):

```
1.  Number 45
2.  Number 23
3.  Number 24
4.  Number 22
5.  Number 34
```

GPT-5.4-mini (think) *LLM-Judge-F1 = 1.00* *LLM-Judge-F1-Ordered = 0.20*

```
1.  24
2.  23
3.  34
4.  45
5.  22
```

Analysis: Retrieves all five correct jersey numbers (LLM-Judge-F1 = 1.00 via optimal matching), but the sequence is scrambled. Only 1 of 5 elements is in the correct position, yielding LLM-Judge-F1-Ordered = 0.20.

Claude Opus 4.6 *LLM-Judge-F1 = 1.00* *LLM-Judge-F1-Ordered = 0.60*

```
1.  #23 (Calvin Murphy)
2.  #45 (Rudy Tomjanovich)
3.  #24 (Moses Malone)
4.  #22 (Clyde Drexler)
5.  #34 (Hakeem Olajuwon)
```

Analysis: Also retrieves all five correct numbers (LLM-Judge-F1 = 1.00) and additionally provides player names. The ordering is closer to correct but swaps the first two positions, yielding LLM-Judge-F1-Ordered = 0.60.

Both models retrieve the complete set of facts but struggle with the specific sequencing constraint. The gap between unordered and ordered scores isolates ordering as a distinct error source, separate from factual retrieval.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction (Sec. 1) clearly state the contributions: the dataset, the evaluation framework, the data quality methodology, and the experimental benchmarking. All claims are supported by the corresponding sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: App. A discusses limitations including English-only scope, geographic skew, ground truth completeness beyond Wikipedia, judge cost and lack of abstention measurement.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper does not include theoretical results. It introduces a dataset and evaluation framework.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Sec. 3 fully describes the evaluation protocol, instruction prompts, and all five metrics. Sec. 4.1 specifies the experimental setup. App. H describes the infrastructure used for evaluating each model whose results are reported. Exact prompt templates are provided in App. F.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The dataset is uploaded to `https://huggingface.co/datasets/AmazonScience/ListQA`, and the evaluation code is archived on Github at `https://github.com/amazon-science/ListQA-Eval`. For reproducibility, Sec. 4.1 specifies the experimental setup. App. H describes the tools and infrastructure used for evaluating each model. Exact prompt templates are provided in App. F. Dataset and code are also linked at `https://listqa.github.io/`. Both dataset and code are released under CC-BY-NC 4.0 license.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Sec. 3.2 describes the dataset variants (full, simple, complex), prompting modes (zero-shot, few-shot), and the instruction prompt. Sec. 4.1 specifies the experimental setup. App. H describes the tools and infrastructure used for evaluating each model. Exact prompt templates are provided in App. F.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All metrics are reported as mean across samples, as described in Sec. 3. Additionally, judge repeatability and judge agreement metrics are discussed in App. R.3 and App. R.4.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Sec. 4.1 and App. H describes the compute resources used for model inference and evaluation.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research uses publicly available Wikipedia data, involves no deception, and annotators were compensated fairly. The broader impact is discussed in App. A.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: App. A discusses both the positive impact of improved factual evaluation and the potential for misuse of improved factual retrieval capabilities.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The dataset consists of factual question-answer pairs derived from publicly available Wikipedia content and does not pose risks for misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Wikipedia content is used as a reference to annotate our dataset. All referenced models and tools are properly cited.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The dataset is documented with detailed statistics (Sec. 2.3, App. C), annotation guidelines (App. B), and evaluation methodology (Sec. 3, App. H). More details can be found at `https://listqa.github.io/`.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [Yes]

Justification: The full annotation guidelines are provided in App. B. Annotators were compensated fairly according to market standards.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The annotation task involves writing factual questions and answers based on Wikipedia tables and does not pose risks to participants. No IRB approval was required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The LLM-as-judge metric (LLM-Judge-F1, discussed in Sec. 3) uses an LLM grader as a core evaluation component. The grader model and its usage are fully described in Sec. 4.1. The prompt templates are provided in App. F.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.