ARQA: A Benchmark for Grounded Table–Text QA in Enterprise Annual Reports

Ruilong Wang
Technical University of Darmstadt
Volkswagen Group
ruilong.wang@volkswagen.de

Simone Balloccu
Technical University of Darmstadt
simone.balloccu@tu-darmstadt.de

Abstract

Annual reports communicate corporate performance to stakeholders through dense tables and explanatory text, with rich grounding signals making automated reasoning challenging. Existing QA benchmarks focus on retrieval or single-modality reasoning, rarely require justification for answers with both textual and tabular evidence. We introduce **ARQA** (Annual Report QA), a benchmark of ~2.5K QA pairs spanning ten fiscal years of automotive enterprise annual reports and three reasoning families—LOOKUP, ARITHMETIC, and INSIGHT. Data are produced via a planner–generator pipeline, deterministically verified and recomputed, and fully reviewed by domain experts. We evaluate state-of-the-art instruction-tuned language models on ARQA, showing strong factual retrieval but persistent weaknesses in grounded arithmetic and causal reasoning. We release ARQA and its evaluation toolkit[1] to facilitate research on auditable, evidence-first reasoning over enterprise documents.

1 Introduction

Annual reports are a company's definitive record of performance, spanning hundreds of pages that combine audited tables with narrative explanations of why Key Performance Indicators (KPIs) changed (Lang and Stice-Lawrence, 2014). These documents inform investors, regulators, and corporate planners, but their hybrid structure of numbers and text poses unique challenges for automated analysis. Professionals rarely seek raw numbers; they want both the story and the supporting evidence behind it—which KPI moved, by how much, and why (e.g., "driven by higher BEV mix", "due to restructuring costs").[2] This task requires reasoning over both structured and unstructured evidence, currently an active research challenge for large language models (LLMs).

Recent advances mainly target isolated reasoning skills such as table arithmetic (Chen et al., 2021, 2022), hybrid table–text retrieval (Zhu et al., 2021; Chen et al., 2020), evidence attribution or long-context understanding (Dasigi et al., 2021; Mathew et al., 2021). None unifies numerical recomputation, dual-modality grounding, and causal explanation, which are essential capabilities for stakeholders to understand how KPIs change and why.

To address these gaps, we introduce **ARQA**, a benchmark built from ten years of real automotive annual reports. It spans production and management domains and contains ~2.5K QA across three reasoning families—LOOKUP (direct retrieval), ARITHMETIC (recomputable numeric reasoning), and INSIGHT (table–text causal explanations)—each grounded in a table and its explanatory paragraphs with cell- and span-level evidence.

The benchmark is constructed in a reproducible manner. Each question is first proposed by coordinated LLM agents, subjected to deterministic checks, and finally reviewed by domain experts.

We evaluate five frontier LLMs on ARQA under two inference setups: (1) **Single-pass**, measuring raw multimodal reasoning; (2) **Type-aware**, providing oracle-level routing by question family. From our results, ARQA exhibits a clear difficulty gradient: models handle simple factual lookups well but degrade on arithmetic recomputation, on INSIGHT questions requiring table–text fusion, and on citing the correct evidence. Enhanced prompting improves procedural reasoning but does not close this gap, underscoring ARQA's challenge as a diagnostic benchmark for grounded enterprise document reasoning.

[1] https://github.com/RuilongWang/ARQA-Benchmark/

[2] Derived from the authors' interviews with domain experts who routinely analyze enterprise annual reports.

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 5: Industry Track), pages 847–868
March 25-27, 2026 ©2026 Association for Computational Linguistics

Dataset	Domain	Modalities	Numeric Reasoning	Evidence Grounding	Causal Reasoning
Tat-QA (Zhu et al., 2021)	Corporate reports	Table + Text	✓	✗	✗
FinQA (Chen et al., 2021)	Financial statements	Table + Text	✓(program)	✗	✗
ConvFinQA (Chen et al., 2022)	Financial (dialog)	Table + Text	✓	✗	✗
AIT-QA (Katsis et al., 2022)	SEC filings	Tables only	✓	✗	✗
Famma (Xue et al., 2024)	Educational finance	Table + Chart + Text	✗	✗	✗
Qasper (Dasigi et al., 2021)	Research papers	Text	✗	✓	✗
AttributionBench (Li et al., 2024)	General QA	Text + Retriever	✗	✓(citation-level)	✗
ARQA (ours)	Enterprise annual reports	Table + Text	✓(program)	✓(table + text)	✓

Table 1: Comparison of related QA and grounding benchmarks by domain, modality, tasks, and requirements.

2 Related Work

Existing evidence-grounded QA and attribution benchmarks primarily evaluate reasoning within a single evidence channel, like text-only justification or table-only numerical operations. Even multimodal datasets combining tables, charts, and text do not require models to jointly interpret quantitative information together with the narrative explanations contextualizing it. In real-world annual reports, however, numerical tables are closely linked to textual causal factors, so effective evaluation must assess table–text fusion and cross-modal grounding.

Numeric and Financial QA. Hybrid reasoning over tables and text has been explored primarily in financial and business contexts. Tat-QA (Zhu et al., 2021) introduced arithmetic reasoning over annual reports, combining textual paragraphs with structured tables. FinQA (Chen et al., 2021) and ConvFinQA (Chen et al., 2022) add executable program traces for numeric reasoning, later extended to multi-turn conversations. AIT-QA (Katsis et al., 2022) examines complex table-only reasoning. Famma (Siqiao Xue and Mei, 2024) introduces multilingual and multimodal QA (charts, diagrams, tables) from educational sources. Recent studies such as FinanceBench (Islam et al., 2023) and T^{2}RAG-Bench (Strich et al., 2025) evaluate retrieval-augmented generation for financial documents, but they still assess correctness mainly at the value level. None of these benchmarks require models to fuse quantitative KPI changes with the textual rationale that explains them, or to prove recomputability from cited cells.

Grounding and Attribution Benchmarks. Fever (Thorne et al., 2018) and Qasper (Dasigi et al., 2021) target verifiable reasoning over text-only documents. AttributionBench (Li et al., 2024) measures citation accuracy in retrieval-augmented generation (RAG) systems, while DialFact (Gupta et al., 2022) introduces conversational claim verification. Recent datasets such as LongBench (Bai et al., 2024) and DocVQA (Mathew et al., 2021) probe long-context understanding but without enforcing numeric or multimodal grounding.

While prior work advances grounding, retrieval, and numerical reasoning, none integrate quantitative KPI changes with the causal narratives that justify them nor require models to prove recomputability from cited table cells. ARQA addresses this gap by unifying table-derived numerical deltas, paragraph-level causal rationale, and explicit evidence grounding into a single expert-validated benchmark tailored to enterprise reporting.

3 The ARQA benchmark

ARQA brings together three elements that have so far remained separate in existing benchmarks: (i) audited enterprise data drawn from real annual reports, (ii) dual-evidence grounding linking numeric movements with their textual explanations, and (iii) a unified evaluation suite that validates numeric recomputation, multimodal grounding, and claim-level semantic alignment. As a result, ARQA bridges the gap between financial QA datasets that prioritize numeric accuracy and grounding benchmarks that evaluate text-only citation. To our knowledge, ARQA is the first expert-audited annual-report benchmark to jointly require recomputable arithmetic, cross-modal evidence attribution, and causal explanation. It offers a closer approximation to how experts and stakeholders reason over annual reports. A comparison of ARQA's characteristics with existing benchmark can be seen in Table 1.

3.1 Data

We build ARQA from ten fiscal years of annual reports (2015–2024) released by Volkswagen Group.[3] We selected this source for two reasons:

[3] Original reports are publicly available from Volkswagen Group Investor Relations: Financial Reports.

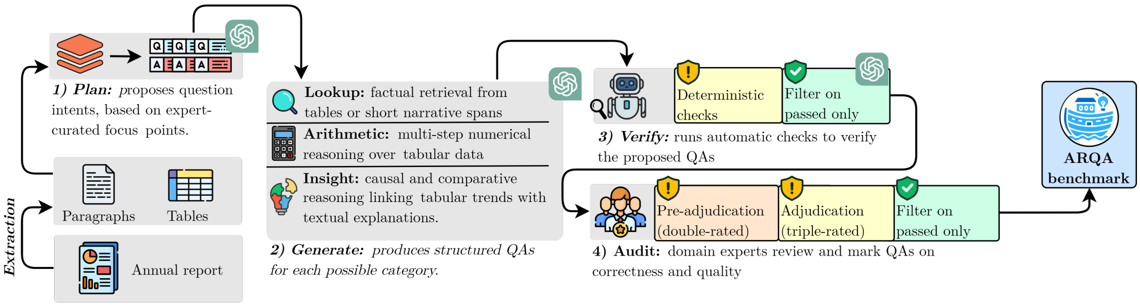

1) Plan: proposes question intents, based on expert-curated focus points.
Paragraphs
Tables
Annual report
Extraction
Lookup: factual retrieval from tables or short narrative spans
Arithmetic: multi-step numerical reasoning over tabular data
Insight: causal and comparative reasoning linking tabular trends with textual explanations.
2) Generate: produces structured QAs for each possible category.
Deterministic checks
Filter on passed only
3) Verify: runs automatic checks to verify the proposed QAs
Pre-adjudication (double-rated)
Adjudication (triple-rated)
Filter on passed only
4) Audit: domain experts review and mark QAs on correctness and quality
ARQA benchmark

Figure 1: ARQA generation and validation pipeline

(1) its multimodal structure, combining dense numerical tables with explanatory narratives; and (2) our collaboration with automotive domain experts. Two domains were prioritized as most decision-relevant: (1) **Production**, covering operational indicators such as vehicle deliveries and revenue; and (2) **Management**, covering governance, marketing, and strategic disclosures.

We define a *pack*, the unit of generation and evaluation, as a pair consisting of one table and its associated descriptive paragraphs, reflecting how analysts interpret key performance indicators within their local narrative context. Each pack contains the table (title, header, and rows)[4]; the paragraphs [5]; and extra metadata (See Appendix A.1).

We narrow the candidate paragraphs associated with each table using GPT-4o as a lightweight retrieval filter. Following prior work on LLMs as weak retrievers (Wang et al., 2023), GPT-4o assigns a coarse topical-relatedness score to nearby paragraphs (pages $p\pm1$) from a table preview. Paragraphs with scores ≥ 0.5 are kept as candidates (prompt is in Appendix C.2). Because such scoring is not fully reliable, we later verify the candidates via deterministic checks and domain-experts review (Section 3.2).

ARQA defines three families of QA (Figure 2), mirroring how analysts interpret annual reports:

- **Lookup:** factual retrieval of explicitly stated values, from table cells or narrative spans.
- **Arithmetic:** Numerical reasoning over table value, plus a symbolic program trace to enable deterministic recomputation.
- **Insight :** fused reasoning that links quantitative changes to their stated drivers, where each answer is composed of one or more *claims* grounded in specific table cells and cue-inclusive text spans.

3.2 Generation and Validation Pipeline

We adopt a four-stage *plan* → *generate* → *verify* → *audit* pipeline (Figure 1). In our setting, the goal is to build a comprehensive benchmark with sufficient coverage; having domain experts evaluate QA pairs generated by multiple different LLMs would multiply expert time and cost substantially. We therefore fix GPT-4o for the automated stages (plan, generate, and verify), consistent with our enterprise model compliance constraints, while the audit stage is conducted by domain experts. Full prompts and implementation details are provided in Appendix C.

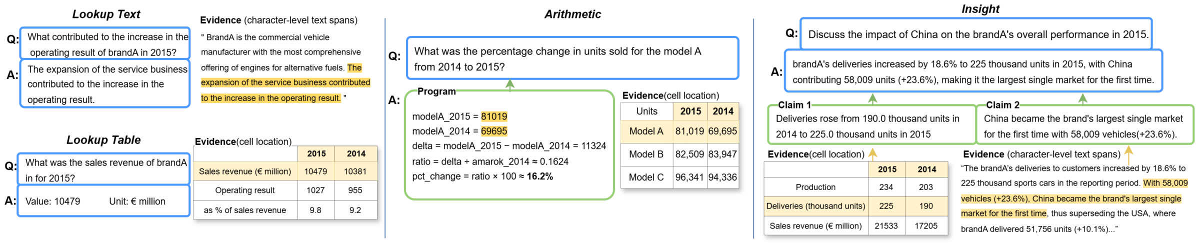
Planner with Focus Points The PLANNER proposes question intents for each pack. GPT-4o is prompted with an expert-curated list of focus points: key topics and KPIs summarized from 2019–2024 press releases and executive interviews, ensuring that generation remains anchored in decision-relevant content.

Family-specific Generators The GENERATOR produces structured QA items from the plan following the family-specific schema:

- LOOKUP: direct value retrieval with grounded cell/span evidence and an explicit unit extracted from table headers or columns.
- ARITHMETIC: a recomputable numeric program with operand references.
- INSIGHT: Two or more *claims* that pair a numeric change with its stated driver, grounded in table cells and cue-inclusive text spans.

LLM Verifier and Deterministic Check. The VERIFIER provides an LLM-based reflection step (Shinn et al., 2023) that attempts minimal self-correction before enforcing hard constraints (details in Appendix A.4). Remaining QAs are kept only if they pass the following deterministic checks:

[4] Extracted from .xlsx files and re-aligned to source page
[5] Converted from PDF to Markdown using Marker



What contributed to the increase in the operating result of brandA in 2015?

The expansion of the service business contributed to the increase in the operating result.

" BrandA is the commercial vehicle manufacturer with the most comprehensive offering of engines for alternative fuels. The expansion of the service business contributed to the increase in the operating result. "

What was the sales revenue of brandA in for 2015?

Value: 10479 Unit: € million

	2015	2014
Sales revenue (€ million)	10479	10381
Operating result	1027	955
as % of sales revenue	9.8	9.2

What was the percentage change in units sold for the model A from 2014 to 2015?

```
modelA_2015 = 81019
modelA_2014 = 69695
delta = modelA_2015 − modelA_2014 = 11324
ratio = delta ÷ amarok_2014 ≈ 0.1624
pct_change = ratio × 100 ≈ 16.2%
```

Units	2015	2014
Model A	81,019	69,695
Model B	82,509	83,947
Model C	96,341	94,336

Discuss the impact of China on the brandA's overall performance in 2015.

brandA's deliveries increased by 18.6% to 225 thousand units in 2015, with China contributing 58,009 units (+23.6%), making it the largest single market for the first time.

Deliveries rose from 190.0 thousand units in 2014 to 225.0 thousand units in 2015

China became the brand's largest single market for the first time with 58,009 vehicles(+23.6%).

	2015	2014
Production	234	203
Deliveries (thousand units)	225	190
Sales revenue (€ million)	21533	17205

"The brandA's deliveries to customers increased by 18.6% to 225 thousand sports cars in the reporting period. With 58,009 vehicles (+23.6%), China became the brand's largest single market for the first time, thus superseding the USA, where brandA delivered 51,756 units (+10.1%)..."

Figure 2: Examples of the ARQA question families

1. **Schema validation:** JSON matches the family specific schema and all identifiers resolve.
2. **Table evidence check:** each cited `table_id`, rows and cells are within valid bounds; recorded cell value matches the canonical table entry after normalization.
3. **Text evidence check:** cited `para_id` exists; the provided character span reproduces the exact substring of the source paragraph.
4. **Arithmetic recomputation:** executing the program with 28-digit precision reproduces the gold value within ±1 unit in the last place (ULP) or a relative tolerance of 10^{-6}.

Expert Audit. For the final AUDIT stage, 35 internal domain experts with backgrounds in automotive production planning and corporate management each reviewed up to 20 *packs* (a table and its surrounding explanatory paragraphs). Only ARITHMETIC and INSIGHT questions are expert-validated, as LOOKUP items are directly verified by deterministic checks as described in Section 3.2. Following protocols (Chen et al., 2021, 2022), we ask the experts to evaluate:

- **Grounded Correctness (GC; Pass/Fail)** — PASS if every claim is supported by the cited table cells and/or paragraphs and all numeric units recompute correctly; otherwise FAIL.
- **Insight Quality (IQ; 1–3)** — 3 = high-impact (decision-critical), 2 = useful (contextually informative), 1 = low value (trivial/irrelevant).

Expert rating instructions are provided in Appendix B. Our IQ scale extends FinQA's binary correctness rubric and was motivated by expert feedback, allowing raters to express graded judgments of a QA's analytical or decision-making relevance. To assess rating agreement, we use:

- **Percent Agreement** — the proportion of items receiving identical labels across annotators, reflecting raw consistency.
- **Gwet's AC1** (Gwet, 2008) — a chance-corrected coefficient robust to prevalence effects, for the binary *Grounded Correctness (GC)* judgments.
- **Krippendorff's** α (Krippendorff, 2018) — for the 3-point *Insight Quality (IQ)* scale, measuring agreement in relative ranking of informativeness.

Each item is annotated by two experts. For conflicting *Grounded Correctness* (GC), labels are reexamined by three senior reviewers (also domain experts).

Audit results. Agreement from expert audit is shown in Table 2. A total of 1,104 items passed double-rating, with very high raw **percent agreement (GC)** (92.7%); 159 items needed triple-rating, but still reached high average pairwise agreement (88.7%); **Gwet's AC1 (GC)** (Gwet, 2008) reached 0.92, indicating very high reliability.

The lower agreement on the ordinal IQ scale reflects a ceiling effect: most items were rated as useful or high-impact, leaving limited variance for disagreement. This pattern mirrors findings in subjective-utility benchmarks (Fabbri et al., 2021), indicating that experts converge on broader correctness, but insight valuation remains inherently subjective. Overall, experts show high raw consistency for GC and broadly aligned judgments for IQ, confirming the rubric's clarity and reproducibility.

Finally, we assign each QA a majority vote GC and median IQ label (Table 3). We reject QAs with GC = FAIL by $n \geq 2$ raters or mean IQ < 2. Of 1,263 validated QAs, only 105 (8.3%) failed—58 for GC, 54 for low IQ, and 7 for both—leaving 1,158 expert-validated ARITHMETIC and INSIGHT items. All released QAs therefore passed the verifier agent, deterministic checks, and expert validation. Detailed benchmark statistics are in Table 4. Additional analyses including full IQ distributions, bootstrap confidence intervals, and pass-rate by QA families are provided in Appendix A.2.

Metric	Scope	Value
Agreement (GC)		
Percent agreement	double-rated	92.7%
Avg. pairwise agreement	triple-rated	88.7%
All-three-agree rate	triple-rated	83.0%
Gwet's AC1	double-rated	0.92 (Chance = 0.10)
Agreement (IQ, linear α)		
Krippendorff's α	double-rated	0.28
	triple-rated	0.12

Table 2: Expert agreement metrics (GC and IQ).

Metric	Scope	Value
Validation pass rates		
GC pass rate	overall	95.4% [94.1–96.6]
	Arithmetic	99.5% [98.9–100]
	Insight	90.9% [88.4–93.3]
IQ pass rate	overall	95.7% [94.3–97.0]
	Arithmetic	94.9% [93.0–96.7]
	Insight	96.5% [94.7–98.2]
Overall benchmark pass rate (GC $\wedge$ IQ $\geq$ 2)		91.7% (1,158 / 1,263)

Table 3: Validation outcomes by QA family with pack-level bootstrap 95% confidence intervals.

Stage	Scope	Value
Initial generation	total QAs	3,268
Self-verification	failed / remaining	546 / 2,701
Deterministic checks	failed / remaining	148 / 2,553
Expert validation	failed / remaining	105 / 2,448
Final composition (2,448 QAs)		
Lookup	count (%)	1,292 (~53%)
Arithmetic	count (%)	623 (~25%)
Insight	count (%)	535 (~22%)

Table 4: ARQA construction statistics.

Cost estimate. To improve reproducibility for practitioners, we report an estimate of dataset construction cost. LLM consumption is approximated using the average prompt and completion tokens per QA for each stage, measured from a sample of representative runs. Details are provided in Table 5. Expert effort is reported as person-hours based on per-expert workload during the audit stage: 35 experts spent 85 minutes each on average, corresponding to approximately 49.6 person-hours in total.

4 Evaluation Protocol

We formulate our evaluation task as question answering over hybrid *annual-report packs*, where each pack contains one table and its associated explanatory paragraphs. Given a question, a system must generate a structured answer belonging to one of three predefined families—Lookup, Arithmetic, or Insight, as a JSON including the predicted answer and its grounded evidence.

Stage	Prompt	Completion	Total	Est. total tokens
Planner	502	31	533	1,741,844
Generator	2257	589	2846	9,300,728
Verifier	1621	64	1685	5,506,580
Total	4380	684	5064	16,549,152

Table 5: Estimated GPT-4o token usage for ARQA construction based on 3,268 initial QAs. Token counts are averaged and amortized per QA (the planner operates at the pack level).

4.1 Lookup Evaluation

Numerical lookup answers are evaluated using **Value–Unit Canonical Exact Match (VU-EM)**, adapted from the standard Exact Match (EM) (Rajpurkar et al., 2016). VU-EM counts a prediction as correct only when both the numeric value and the normalized unit match the gold reference after applying a controlled unit glossary. This addresses variations in unit expression across annual reports (e.g., "million EUR" vs. "€ million") and synonymous forms.

For textual lookups, we follow prior work on financial QA (Chen et al., 2021; Zhu et al., 2021) and measure semantic equivalence between predicted and reference answers via continuous **BERTScore-F1** (Zhang et al., 2020). We also compute **Evidence F1** over cited evidence, matching table cells by exact coordinates (**Cell Evi. F1**) and text spans by the Intersection-over-Union (IoU) between their character offsets within the same paragraph (IoU $\geq$ 0.5) (Zhu et al., 2021) (**Span Evi. F1**). Paragraph-ID hit rate is reported as a weak grounding signal but does not affect the main accuracy metric (see Appendix A.3).

4.2 Arithmetic Evaluation

Following the validation protocols of FinQA (Chen et al., 2021) and TAT-QA (Zhu et al., 2021), Arithmetic questions are evaluated via the arithmetic recomputation detailed in Section 3.2. We additionally compute **Evidence F1** over cited table cells, where precision and recall are defined on exact matches of table cell coordinates.

		Lookup								Arithmetic				Insight					
		VU-EM$^{\uparrow}$		Cell Evi. F1$^{\uparrow}$		Sem. F1$^{\uparrow}$		Span Evi. F1$^{\uparrow}$		Recompute Acc.$^{\uparrow}$		Evi. F1$^{\uparrow}$		Sem. F1$^{\uparrow}$		Claim F1$^{\uparrow}$		Evi. F1$^{\uparrow}$	
Model	**Setup**	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT
LLAMA-3.1 8B	S1	0.63	0.60	0.83	0.86	0.54	0.54	0.09	0.06	0.48	0.44	0.71	0.76	0.33	0.02	0.49	0.02	0.32	0.02
	S2	0.52	0.55	0.84	0.85	0.51	0.52	0.06	0.05	0.49	0.47	0.66	0.73	0.37	0.40	0.47	0.54	0.30	0.28
LLAMA-3.3 70B	S1	0.86	0.87	0.96	0.93	0.65	0.38	0.10	0.08	0.74	0.71	0.92	0.86	0.36	0.02	0.46	0.02	0.34	0.02
	S2	0.83	0.79	0.97	0.98	0.57	0.58	0.10	0.11	0.79	0.73	0.89	0.93	0.39	0.41	0.47	0.51	0.43	0.41
QWEN-2.5 32B	S1	0.81	**0.92**	0.98	0.98	0.69	0.70	0.11	0.11	0.91	0.89	0.94	**0.95**	0.37	0.25	0.57	0.37	0.41	0.18
	S2	0.47	0.80	0.98	0.97	0.69	0.69	0.09	0.10	0.91	0.80	0.94	0.94	0.37	0.40	0.53	0.55	0.40	0.38
DEEPSEEK 32B	S1	0.76	0.78	0.95	0.91	0.59	0.47	0.08	0.05	0.83	0.62	0.90	0.80	0.38	0.02	0.50	0.02	0.38	0.02
	S2	0.67	0.84	0.97	0.93	0.57	0.60	0.08	0.07	0.83	0.84	0.91	0.93	0.41	0.45	0.49	0.54	0.37	0.37
GPT-4O	S1	0.91	0.73	0.98	0.84	0.68	0.56	**0.16**	0.10	**0.92**	0.85	**0.95**	0.92	0.44	0.02	**0.61**	0.02	0.46	0.02
	S2	0.80	0.84	**0.99**	**0.99**	0.69	**0.72**	**0.16**	**0.16**	0.91	0.90	0.94	0.94	0.43	**0.47**	0.55	0.57	0.46	**0.47**

Table 6: Results on ARQA benchmark across all models and setups. DEEPSEEK 32B = DEEPSEEK-R1-DISTILL-QWEN 32B. For all metrics a higher value is better.

4.3 Insight Evaluation

Following QASPER's multi-component evaluation (Dasigi et al., 2021), we adopt a **three-level evaluation protocol** assessing *answers*, *claims*, and *evidence*:

- **Answer Semantic F1:** BERTScore-F1 between the overall predicted and gold answers.
- **Claim Semantic F1:** mean BERTScore-F1 between predicted and gold claim texts, aligned one-to-one by maximal semantic similarity (cosine space of the ROBERTA-LARGE encoder).
- **Evidence F1:** QA-level coverage over all cited evidence, combining exact table-cell matches and text spans with IoU ≥ 0.5 within each paragraph.

5 Experiments

We evaluate five open-weight instruction-tuned LLMs—LLAMA-3.1 8B INSTRUCT, LLAMA-3.3 70B INSTRUCT, QWEN-2.5 32B INSTRUCT, DEEPSEEK-R1-DISTILL-QWEN 32B, and the closed-weight GPT-4O—on the ARQA benchmark. All models are evaluated under two progressively structured inference setups:

- **S1 (Base Structured):** Single-pass zero-shot inference using the schema-explicit prompt from Section 4.
- **S2 (Type-aware):** Separate specialized prompts for LOOKUP, ARITHMETIC, and INSIGHT families. All experiment prompts are in Appendix D.

For each setup, we additionally evaluate Few-shot + Chain-of-Thought prompting (Wei et al., 2022). Prompts are included in Appendix E. We also experiment with a multi-agent setup, which we include in Appendix F as it did not show consistent improvements.

5.1 Results

Overall results (Table 6) show that the **Type-aware** setup (S2) improves grounding but often reduces answer accuracy for LOOKUP. Across all models, S2 lowers VU-EM while increasing Cell Evidence F1, suggesting that oracle routing helps models localize the correct table region but interferes with value–unit prediction. A similar pattern holds for INSIGHT: Claim F1 typically fall under S2 even though Evidence F1 remains stable, indicating that S2 guides models to the right evidence but overconstrains generation, harming semantic precision. Arithmetic shows minor mixed changes under S2, with no consistent gains in recomputation accuracy.

The **Few-shot + CoT** prompt shows another failure mode when the model is not guided on how to structure its answers. Under S1, CoT frequently breaks the output schema especially for INSIGHT, because exemplars from all families are shown together. Smaller models copy the wrong format or omit required fields. When the family is fixed (S2), enhanced prompt becomes more reliable: models are not distracted by examples from other families, yielding steadier accuracy and grounding gains. GPT-4O remains the strongest overall; among open-weight models, QWEN-2.5 32B is the most robust.

A persistent weakness across all systems is extremely low Span Evidence F1. Models often locate the correct paragraph but fail to extract the correct character-level span (diagnostics in Appendix A.3). This mirrors prior work (Zhu et al., 2021; Dasigi et al., 2021).

Overall, **ARQA** shows that LLMs suffer from brittle generation, schema sensitivity, and limited cross-modal grounding, underscoring the need for more principled multi-evidence reasoning methods.

6 Conclusion

We introduced ARQA, a benchmark for auditable, evidence-grounded reasoning over real enterprise annual reports. ARQA unifies recomputable numerical reasoning, table–text grounding, and causal explanation, and provides a validated ten-year corpus with cell- and span-level evidence. A rigorous generation pipeline and expert audit ensure that all released items are semantically correct, numerically reproducible, and decision-relevant.

Evaluations across state-of-the-art LLMs show that while models handle factual lookups reliably, they struggle with arithmetic recomputation, cross-modal causal explanation, and precise evidence citation. Enhanced prompting improves procedural reasoning but leaves large gaps on INSIGHT tasks, highlighting ARQA's value as a diagnostic testbed for grounded enterprise QA.

ARQA establishes a challenging setting for developing models that can reason over structured and narrative financial disclosures. Future work includes extending the benchmark to cross-table KPI reasoning, incorporating table-importance priors, and broadening coverage to additional industries and languages.

Limitations

We constructed ARQA using real-world data and validating it with domain experts, to present new challenging evaluation setups related to the automatic analysis of annual reports. Still, our work presents some limitations that we plan to cover in future work.

Multi-year data ARQA is limited to single-year reasoning: each pack contains one table and its local narrative, preventing cross-year or cross-document analysis. Real-world analysts often cross-validate KPI changes by comparing current-year values with previous years, or inspect multiple periods to identify trends. Future work will construct a *cross-year KPI graph* to align equivalent metrics across tables and years, enabling temporal trend and causal reasoning.

Evidence priority The current generation process also ignores *table importance*—all tables are sampled uniformly, whereas analysts prioritize high-salience financial or ESG summaries. Incorporating expert-weighted sampling could yield more decision-relevant questions.

Model heterogeneity We generated ARQA with GPT-4o only. Focusing our prompting and generation effort on a single model allowed us to have more control and knowledge of the output, resulting in a higher final quality. In addition, our enterprise setting imposed compliance constraints that restricted data construction to an internally approved model (GPT-4o). In future, we plan to inspect generation with different models, with a special focus on the gap between open and closed-weight ones, and different model scale.

Domain representation Finally, the benchmark covers only two domains (production and management) from one industrial group; while the construction recipe and evaluation protocol are designed to transfer to other annual-report corpora given comparable expert review, extending to other sectors and languages would further broaden its applicability to enterprise document understanding.

Acknowledgments

We thank the members of the UKP Lab at Technical University of Darmstadt for helpful discussions and support. We also thank the domain experts who participated in the audit and adjudication process for their time and careful feedback. This research was supported by Volkswagen Group, which provided access to domain expertise essential for constructing and validating the benchmark.

References

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. 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, Online. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, and 1 others. 2021. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 2022. ConvFinQA: Exploring the chain of numerical reasoning in conversational finance question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6292, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Prakhar Gupta, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. DialFact: A benchmark for fact-checking in dialogue. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3785–3801, Dublin, Ireland. Association for Computational Linguistics.

Kilem Gwet. 2008. Computing inter-rater reliability and its variance in the presence of high agreement. *The British journal of mathematical and statistical psychology*, 61:29–48.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. Financebench: A new benchmark for financial question answering. *Preprint*, arXiv:2311.11944.

Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. 2022. AIT-QA: Question answering dataset over complex tables in the airline industry. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 305–314, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Klaus Krippendorff. 2018. *Content Analysis: An Introduction to Its Methodology*, 3 edition. SAGE Publications.

Mark Lang and Lorien Stice-Lawrence. 2014. Textual analysis and international financial reporting: Large sample evidence. *SSRN Electronic Journal*, 60.

Yifei Li, Xiang Yue, Zeyi Liao, and Huan Sun. 2024. AttributionBench: How hard is automatic attribution evaluation? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14919–14935, Bangkok, Thailand. Association for Computational Linguistics.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2199–2208.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. 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, Austin, Texas. Association for Computational Linguistics.

Noah Shinn, Saad Labash, and Rohan Gopinath. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*.

Fan Zhou Qingyang Dai Zhixuan Chu Siqiao Xue, Xiaojing Li and Hongyuan Mei. 2024. Famma: A benchmark for financial domain multilingual multimodal question answering. *arXiv preprint arXiv:2410.04526*.

Jan Strich, Enes Kutay Isgorur, Maximilian Trescher, Christian Biemann, and Martin Semmann. 2025. T2-ragbench: Text-and-table benchmark for evaluating retrieval-augmented generation. *ArXiv*, abs/2506.12071.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Siqiao Xue, Xiaojing Li, Fan Zhou, Qingyang Dai, Zhixuan Chu, and Hongyuan Mei. 2024. Famma: A benchmark for financial domain multilingual multimodal question answering. *arXiv preprint arXiv:2410.04526*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations (ICLR)*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287, Online. Association for Computational Linguistics.

A Appendix

A.1 Metadata in the pack

Besides the basic element described in Section 3.1, each pack contains the following metadata: `table_id`, `doc_id`, `page`, `year`, `section_name` and `bucket` (a category tag, production or management).

A.2 Detailed Expert Audit Results

We report some further results from the expert audit phase described in Section 3.2. For all the insights categories that experts rated, Figure 3 shows the distribution of the IQ ratings by expert; Figure 4 we report the mean value for IQ; Figure 5 shows the pass rate for IQ and GC.

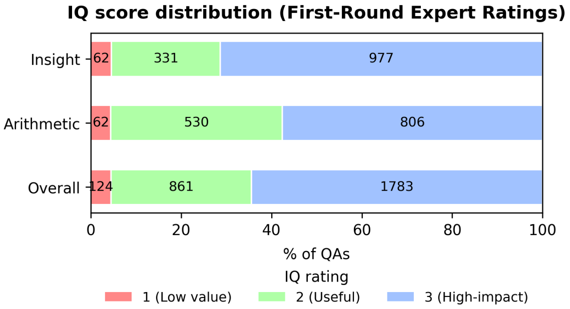


Figure 3: IQ score distribution across overall, ARITHMETIC, and INSIGHT QAs.

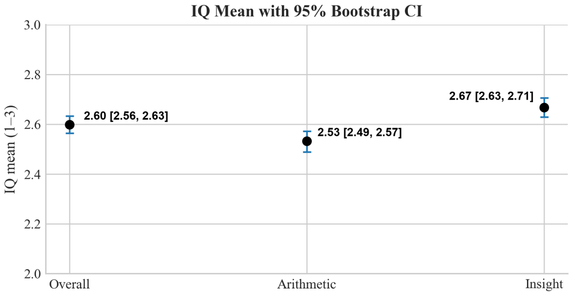


Figure 4: Mean IQ with 95% bootstrap confidence intervals.

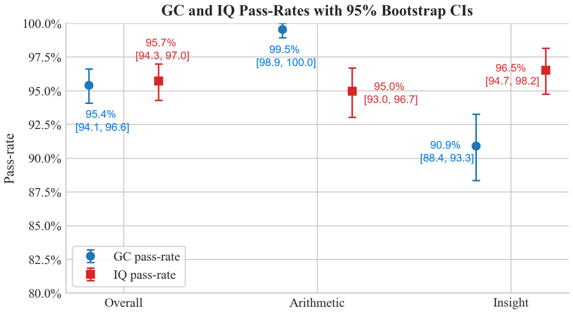


Figure 5: GC and IQ pass rates with 95% bootstrap confidence intervals.

A.3 Experiment result diagnostics

Table 7 provides further diagnostic breakdowns. Across all models, Paragraph Hit Rate is consistently high, indicating that models can reliably identify the correct paragraph. However, Span Evidence F1 remains extremely low (often below 0.15), confirming that models struggle to extract the correct character-level spans even when they retrieve the correct paragraph.

Model	Setup	Value EM		Unit EM		Span Evi. F1		P. Hit Rate	
		Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT
LLAMA-3.1 8B	S1	0.70	0.62	0.80	0.91	0.09	0.06	0.90	0.73
	S2	0.66	0.66	0.65	0.69	0.06	0.05	0.91	0.84
LLAMA-3.3 70B	S1	0.90	0.91	0.90	0.88	0.10	0.08	0.96	0.46
	S2	0.92	0.94	0.86	0.81	0.10	0.11	0.95	0.95
QWEN-2.5 32B	S1	0.96	0.94	0.83	0.96	0.11	0.11	0.97	0.97
	S2	0.92	0.94	0.52	0.82	0.09	0.10	0.96	0.95
DEEPSEEK 32B	S1	0.92	0.83	0.78	0.85	0.08	0.05	0.93	0.60
	S2	0.92	0.90	0.70	0.86	0.08	0.07	0.92	0.94
GPT-4O	S1	0.95	0.77	0.93	0.78	0.16	0.10	0.97	0.67
	S2	0.95	0.94	0.83	0.87	0.16	0.16	0.96	0.97

Table 7: Lookup QA diagnostics across all models and setups. P. Hit Rate(Paragraph Hit Rate) measures the proportion of predictions citing correct paragraph ID.

A.4 Verifier reflection behavior

The LLM verifier is a minimal post-generation correction step inspired by the error-reflection loop proposed in Reflexion (Shinn et al., 2023). In our pipeline, the verifier does not generate new answers. It performs only local repairs such as: (i) correcting mismatched or missing units, (ii) fixing malformed JSON fields, (iii) aligning table cell references with the intended schema.

The prompt used is shown in Figure 12 (temperature = 0.0).

B Survey example

Component	Model	Settings
QA Generation Pipeline (GPT-4o)		
Planner Agent	GPT-4o	temp = 0.4
Insight QA Generator	GPT-4o	temp = 0.5
Arithmetic / Lookup Generators	GPT-4o	temp = 0.2
QA Verifier	GPT-4o	temp = 0.0
Experiment Inference Settings		

Global Defaults	all models	temp = 0.0, top_p = 1.0
		max_tokens = 2000

Execution Phase	all models	temp = 0.0
Planner Phase (S3)	all models	temp = 0.0

Table 8: LLM configurations used in the ARQA generation pipeline and inference setups.

About this study

We are validating AI-generated Q&A built from company annual reports and other public documents. Your expert judgement checks both factual accuracy and business relevance. This study is totally anonymous.

For the sake of quality and consistency calibration of the survey, there are a few attention QA items. These items look the same as other questions, but contain obvious errors. Please answer them according to the same standards.

What you'll see

A table and its surrounding paragraphs (source context).

AI-generated QAs with references (incl. press interviews/conferences).

Two QA types:

- Arithmetic − multi-step calculation is shown with cited table cells.
- Insight − an answer made of claims backed by evidence from the table/text.

How you'll rate

Grounded Correctness (Pass/Fail):

Pass if every claim is supported by the cited table/paragraphs and the math/units are correct; otherwise Fail. Judge only the correctness of the provided claims − if you want more context, put it in the notes (that doesn't make the answer incorrect).

Insight Quality (1-3):

3 = high-impact (you/your company care; important)

2 = useful (interesting / nice-to-know)

1 = low value (trivial / not important)

Judge from a company perspective; it does not have to match your exact role or daily work.

What you need to do

Glance over the table and paragraphs.

Read each question and its answer.

Rate every QA:

- Arithmetic: check the calculation steps and that cited numbers come from the table. You do not need to recompute − just verify steps & sources.
- Insight: check that each claim is correct and its evidence really supports the answer.

Figure 6: Survey guidance for the experts

C ARQA generation configuration and prompts

C.1 LLM Configuration for Generation and Experiments

For reproducibility, Table 8 provides the full set of LLM hyperparameters used in ARQA's data-generation pipeline and inference experiments.

C.2 ARQA generation prompts

Task: relevance_scoring

You will evaluate whether each paragraph is relevant to the given table.

table_id: table identifier

headers_preview: list of column headers

rows_preview: first ten table rows (for context)

stub_col_preview: table stub column if available; otherwise empty

paragraphs: list of paragraph previews

Definition of relevance:

Relevance means that the paragraph matches the topic or scope of the table; it does not need to repeat specific numbers from the table.

Output schema:

list of { para_id, relevance_score in [0,1], reason }

where relevance_score is a continuous value between 0 and 1 indicating how well the paragraph matches the table's subject.

Figure 7: Relevance scoring prompt

Role: Planner for QA generation from one annual-report table and its nearby paragraphs.

Goal: Produce a JSON plan describing how many QAs of each type (lookup, numerical, insight) to generate and define each question's focus.

Context:

- Table {table_id} ({section})
- Paragraphs: {paras_text}

Caps: lookup $\leq$ 4, numerical $\leq$ 2, insight $\leq$ 2

Instructions:

1. Allocate a small, diverse set of question slots across families (lookup, numerical, insight).
2. Each slot should target a unique KPI/segment/period/entity to avoid overlap.
3. Choose topics directly from context (finance, ESG, operations, governance, outlook, risk, regions/brands).
4. Include numerical items only if the table enables meaningful calculations.
5. Include insight items only when paragraphs provide reasoning, causes, or outlook (no speculation).
6. Favor variety across entities, periods, and KPIs.
7. Return **STRICT JSON only**. No commentary.

Focus Themes (if explicitly present):

- Margin corridor & drivers (tariffs, BEV mix dilution, brand swings)
- Tariff impact & mitigation (localization, pricing levers)
- Cost-cutting / restructuring (Future Company, daughter company layoffs)
- Country strategies (China "right-size", daughter company with strategy)
- BEV orders vs. margin dilution
- Brand group contributions (BrandA, BrandB, BrandC, Porsche, etc.)
- Cash flow & liquidity
- Software strategy (daughter companyA vs. daughter companyB scope/timing)

Output JSON:

```
{
  "family_counts": {"lookup": int, "numerical": int, "insight": int},
  "lookup_items": [{"q_type": "lookup_table""lookup_text", "desc": "short focus"}],
  "numerical_items": [{"arith_type": "...", "desc": "short focus"}],
  "insight_items": [{"desc": "short focus"}]
}
```

Ensure item counts respect caps, avoid overlap, and stay grounded in the given context.

Figure 8: Planner prompt

```
Create lookup QAs strictly from the given context. Do NOT calculate or paraphrase.
Produce up to {count_table} items of type "lookup_table" and up to {count_text} items of type "lookup_text".
Writing rules:
- Questions must identify the KPI/entity/period precisely so the answer is unique.
- Keep questions concise (≤ 22 words).
- Units come from table headers (e.g., 'ppt', '- Do not generate two QAs that target the same KPI/entity/period.
Evidence format:
lookup_table → evidence: { "table_id":"{table_id}", "row":"<row_label>", "col":"<header_label>",
"value":"<exact_cell_string>" }
lookup_text → evidence: { "para_id":"<para_id>", "char_start":<int>, "char_end":<int>,
"text":"<exact_substring>" }
Return ONLY a JSON array of QA objects:
{
  "q_type": "lookup_table"|"lookup_text",
  "question": "string",
  "answer_text": "string",
  "value": "string",
  "value_canonical": number|null,
  "unit": "string",
  "evidence": {...}
}
If requested items are provided below, realize them in order: {lookup_plans}.
If fewer valid items exist, return fewer. Never invent content.
Context:
Table context: {table_context}
Paragraphs (original ids included; use these ids verbatim): {para_context}
```

Figure 9: Lookup QA Generator prompt

```
Input:
table_id: table_id
table_headers: headers (visible column headers as strings)
table_rows: rows (row labels and cell strings)
request_counts: planned (desired number of arithmetic QAs)
requested_items: numerical_plans (if any)
Rules:
- Use only operands visible in this table.
- Include all operand cells and any referenced header columns in "evidence".
- Every read step in "program" must include the exact printed cell string as "value".
- Every evidence cell must include "value" identical to the exact cell string.
- Follow consistent formulas for each arithmetic type:
• pct_change: (new - old) / abs(old) * 100
• pct_point_change: (new• share: (part / total) * 100
• weighted_average: (w_i * x_i) / (w_i)
• index_base: value_t / value_base * 100
• contribution_share: segment / segments * 100
• variance_to_target: actual - target
• rank_topk: specify k and axis; include ordered list in program
• count: number of rows/columns satisfying a condition (>0, <target, etc.)
Output schema (strict JSON):
{
  "fields_per_item": ["q_type","arith_type","question","answer_text","answer_value","unit","program","evidence"],
  "q_type": "must be 'arithmetic'",
  "arith_type": "one of: minmax, diff, pct_change, pct_point_change, share, ratio, sum_total, average, count, rank_topk,
weighted_average, index_base, contribution_share, variance_to_target",
  "answer_value": "numeric result for verification",
  "unit": "use unit from table header ('  "program": "list of read and compute steps with op, inputs, and result",
  "evidence": "list of operand and header citations with exact cell strings"
}
Validation:
- answer_text must equal answer_value + unit (e.g., "58.3- Every operand appearing in "program" must be cited in
"evidence".
- No duplicate KPI/entity/period with the same operation.
- Return a JSON array only.
Examples:
{
  "q_type": "arithmetic",
  "arith_type": "pct_point_change",
  "question": "By how many percentage points did the Group operating margin change from 2024 to H1 2025?",
  "answer_text": "-1.4 ppt",
  "answer_value": -1.4,
  "unit": "ppt",
  "program": [read, sub steps...],
  "evidence": [rows + headers for 2024, H1 2025]
}
{
  "q_type": "arithmetic",
  "arith_type": "contribution_share",
  "question": "What share of the Group's 2024→2025 revenue increase came from Brand Group Core?",
  "answer_text": "42.7  "answer_value": 42.7,
  "unit": "  "program": [read, sub, div, mul steps...],
  "evidence": [rows + headers for 2024, 2025]
}
```

Figure 10: Arithmetic QA Generator prompt

```
Input:
table_id: table_id
headers: headers (visible column headers)
rows_preview: rows (row labels and key cell strings)
paragraphs_preview: para_context (paragraph snippets related to the table)
insight_plan: planner-provided insight descriptions to prioritize
request_counts: {"insight": planned_count}
Rules:
- Use only information present in the table and paragraphs.
- Each QA must include ≥1 TABLE claim and ≥1 TEXT claim.
- For TABLE change claims, cite ≥2 cells for the same KPI/entity across periods (e.g., 2023 vs 2024).
- TEXT evidence must include the causal cue substring (e.g., "due to", "driven by", "as a result of").
- Avoid vague adverbs like "significantly" or "slightly". When describing a change, include both from→to values and
the (+- Recognize parentheses convention: e.g., "€40,083 (40,530) million" → current = 40,083; prior = 40,530.
- Entity scope must align between table and text. If paragraph mentions another entity, rescope the question or use
a matching paragraph. Never mix entities.
- Prefer 1-3 text claims (distinct drivers) instead of one long statement.
- Avoid overlapping QAs; vary KPI, entity/brand, period, or driver focus.
Output schema (strict JSON):
{
  "fields_per_item": ["question", "gold_answer"],
  "gold_answer": {
    "answer": "One or two sentences: [direction  magnitude] + [timeframe] + [driver(s)] + [share/weight if helpful].",
    "claim_object": [
      {
        "type": "table",
        "claim_text": "KPI change phrase (e.g., 'Deliveries rose from 5,980.0 to 6,230.0 thousand units in 2017').",
        "evidence": { "table_cells": [
          {"table_id":"str","row":"visible row label","col":"visible column header","cell_text":"exact string"}
        ]}
      },
      {
        "type": "text",
        "claim_text": "Driver or impact phrase including causal cue.",
        "evidence": { "text_spans": [
          {"para_id":"str","char_start":int,"char_end":int,"text":"substring including cue"}
        ]}
      }
    ]
  }
}
Validation:
- Return a JSON array only.
- Array length ≤ request_counts["insight"].
- Each item must include ≥1 table claim (≥2 cells for changes) and ≥1 text claim.
- Every claim must include evidence (non-empty table_cells/text_spans).
- Each cited cell must include exact cell_text as printed.
- No duplicate (KPI, period, driver) combinations across items.
Example:
{
  "question": "How important was the Tiguan to VW Passenger Cars’ record deliveries in 2017?",
  "gold_answer": {
    "answer": "BrandA’ deliveries rose 4.2    "claims": [
      { "type": "table",
      "claim_text": "Deliveries increased from 5,980.0 to 6,230.0 thousand units in 2017 (↑4.2      "evidence": { "table_cells": [
        {"table_id": table_id,"row": "Deliveries (thousand units)","col": "2016","cell_text": "5,980.0"},
        {"table_id": table_id,"row": "Deliveries (thousand units)","col": "2017","cell_text": "6,230.0"}
      ]}},
      { "type": "text",
      "claim_text":  "The Tiguan delivered  720,000 units in 2017 and was described as one of the world’s most
successful automobiles.",
      "evidence": { "text_spans": [
        {"para_id": "VW2017_P387e10","char_start":0,"char_end":180,
        "text":  "...  720,000 vehicles delivered in 2017, making it one of the world’s most successful automobiles
..."}
      ]}}
    ]
  }
}
```

Figure 11: Insight QA Generator prompt

```
Verifier role
You are a strict QA verifier for annual-report Q&As using table and paragraph context.
Verify four families:
  - lookup_table: exact table cell match
  - lookup_text: exact substring span match
  - arithmatic: recompute result from evidence
  - insight: fusion; must include both table and text claims
Convention: In prose like “sales revenue €40,083 (40,530) million”, the parentheses denote the prior-year value.
Output a JSON array only. For each QA, return:
{qa_id, verified: true|false, reason: string, action: "repair"|"regenerate"|"none", advice: string}.
Context provided
table: { table_id, headers, rows }
paragraphs: paragraph text array
qas: list of QAs to verify
schema:
  qa_fields: ["qa_id","q_type","question","answer_text","evidence"]
  arithmatic:
    required: ["arith_type","program","evidence"]
    notes:
      - For read steps, program.value must equal the exact cell string.
      - For computed steps, include result.
      - Evidence cells must include exact cell values when provided by the generator.
  insight:
    required: ["gold_answer"]
    notes:
      - gold_answer.claim_object must include at least one TABLE claim and one TEXT claim.
      - TABLE claims: evidence.table_cells must be non-empty and reference valid row/col labels.
      - TEXT claims: evidence.text_spans must include a substring with cue words, valid para_id, and valid char_start/char_end/text.
```

Figure 12: Verifier prompt

D Experiments prompt

```
Task:
Answer the question using only the provided pack (table + paragraphs).
Infer the correct question family and return exactly one JSON object with the envelope:
{
  "predicted_family": "lookup_table"|"lookup_text"|"arithmetic"|"insight",
  "prediction": { /* one family payload below */ }
}
Family payload schemas (choose exactly one):
A) LOOKUP – table
{
  "answer_text": "string",
  "unit": "string"|null,
  "value_canonical": number|null,
  "evidence": {
    "table_id": "string",
    "row": "string",
    "col": "string",
    "cell_text": "string"
  }
}
B) LOOKUP – text
{
  "answer_text": "string",
  "evidence": {
    "para_id": "string",
    "char_start": number,
    "char_end": number,
    "text": "string"
  }
}
C) ARITHMETIC
{
  "answer_text": "string",
  "answer_value": number,
  "unit": "string"|null,
  "program": [ /* explicit recomputable steps */ ],
  "evidence": [ /* referenced table cells */ ]
}
D) INSIGHT (fusion; requires ≥ 1 table claim + ≥ 1 text claim)
{
  "answer": "string",
  "claims": [
    {
      "type": "table"|"text",
      "claim_text": "string",
      "evidence": {
        "table_cells": [ {"table_id":"string","row":"string","col":"string","cell_text":"string"} ],
        "text_spans": [ {"para_id":"string","char_start":number,"char_end":number,"text":"string"} ]
      }
    }
  ]
}
Provided context:
PACK (JSON): contains the table + paragraphs + metadata.
QUESTION (JSON): contains the natural-language query to answer.
Return the final JSON object immediately – no prose, no Markdown.
```

Figure 13: Experiment setup 1 prompt

```
Family: Lookup (table or text)
Emit exactly one JSON object with this envelope:
{
  "predicted_family": "lookup_table" OR "lookup_text",
  "prediction": { /* one of the two schemas below */ }
}
Schemas (choose exactly one):
- If the answer is grounded in a table cell (preferred when a precise numeric value exists):
{
  "answer_text": "string",
  "unit": "string"|null,
  "value_canonical": number|null,
  "evidence": { "table_id":"string", "row":"string", "col":"string", "cell_text":"string" } // row/col are NAMES
}
- If the answer is grounded in a paragraph span:
{
  "answer_text": "string",
  "evidence": { "para_id":"string", "char_start":number, "char_end":number, "text":"string" } // char offsets into
paragraph
}
Hard rules:
- Set predicted_family to "lookup_table" when citing a table cell; otherwise "lookup_text".
- Always cite row and column by name, not by index.
- Always include para_id, char_start, and char_end for text spans.
- Units must be canonical (€, - Cite exactly one best cell or one best text span.
Provided context:
PACK (JSON): table + paragraphs for lookup evidence.
QUESTION (JSON): natural-language query to answer.
Return the final JSON immediately — no prose, no Markdown.
```

Figure 14: Experiment setup 2 prompt (Lookup QA)

```
Family: Arithmetic
Emit exactly one JSON object with this envelope:
{
  "predicted_family": "arithmetic",
  "prediction": {
    "answer_text": "string",
    "answer_value": number, // precise numeric for recomputation
    "unit": "string"|null,
    "program": [ // explicit recomputable steps
      {"op":"read","as":"new","cell":{"table_id":"string","row":"string","col":"string","value":"string"}},
      {"op":"read","as":"old","cell":{"table_id":"string","row":"string","col":"string","value":"string"}},
      {"op":"sub","inputs":["new","old"],"as":"diff"},
      {"op":"div","inputs":["diff","old"],"as":"ratio"},
      {"op":"mul","inputs":["ratio",100],"as":"pct"}
    ],
    "evidence": [
      {"table_id":"string","row":"string","col":"string","value":"string"},
      {"table_id":"string","row":"string","col":"string","value":"string"}
    ]
  }
}
Hard rules:
- Use only primitive operations: read, add, sub, mul, div (plus mul × 100 for - answer_value must recompute exactly
from the program; answer_text may be rounded.
- Always cite row and column by name (no numeric indices) in both program reads and evidence.
- Units must be canonical (€, - Cite all operands explicitly and include both in the evidence list.
Provided context:
PACK (JSON): table and paragraph data.
QUESTION (JSON): natural-language query.
Return the final JSON immediately — no prose, no Markdown.
```

Figure 15: Experiment setup 2 prompt (Arithmetic QA)

Family: Insight (fusion-only)

Emit exactly one JSON object with this envelope:

```
{
  "predicted_family": "insight",
  "prediction": {
    "answer": "string", // concise synthesis
    "claims": [ // ≥1 table-backed AND ≥1 text-backed claim
      {
        "type": "table"|"text",
        "claim_text": "string",
        "evidence": {
          "table_cells": [ {"table_id":"string","row":"string","col":"string","cell_text":"string"} ], // row/col are NAMES
          "text_spans": [ {"para_id":"string","char_start":number,"char_end":number,"text":"string"} ] // char offsets into paragraph
        }
      }
    ]
  }
}
```

Hard rules:

- Enforce dual evidence across claims: include ≥1 table claim and ≥1 text claim.
- Always cite row and column by **name** (no numeric indices) for table_cells.
- Always include para_id, char_start, and char_end for text_spans, ensuring spans are within valid offsets.
- Keep claims atomic, factual, and unit-consistent.
- Each claim must express one verifiable statement supported by cited evidence.

Provided context:

PACK (JSON): includes table and paragraphs for both quantitative and textual evidence.

QUESTION (JSON): reasoning-style query requiring synthesis of numerical change and qualitative cause/impact.

Return the final JSON immediately — no prose, no Markdown.

Figure 16: Experiment setup 2 prompt (Insight QA)

E Enhanced prompting

```
Table example
Table:
Header: ["€ million", "2015", "2014"]
Rows:
    ["Gross cash flow", "4722.0", "17965.0"],
    ["Change in working capital", "15469.0", "2682.0"],
    ["Cash flows from operating activities", "20191.0", "20647.0"]
Q: What was the gross cash flow in 2015?
A: Locate the row "Gross cash flow" and read the value under "2015". The value is 4722.0 and the unit is €
million.
Final answer (JSON):
{
  "qa_id": "655d404026",
  "q_type": "lookup_table",
  "question": "What was the gross cash flow in 2015?",
  "answer_text": "4722.0 € million",
  "value": "4722.0",
  "value_canonical": 4722,
  "unit": "€ million",
  "evidence": { "table_id":"VW2015_Tb65d24", "row":"Gross cash flow", "col":"2015", "value":"4722.0" }
,
  "table_id": "VW2015_Tb65d24"
}

Text example
Text:
"The Commercial Vehicles/Power Engineering Business Area generated gross cash flow of 2.8 billion in the reporting
period..."
Q: What was the gross cash flow in the Commercial Vehicles/Power Engineering Business Area in 2015?
A: The text states: "generated gross cash flow of € 2.8 billion".
Final answer (JSON):
{
  "qa_id": "afe9b6f309",
  "q_type": "lookup_text",
  "question": "What was the gross cash flow in the Commercial Vehicles/Power Engineering Business Area in 2015?",
  "answer_text": "€ 2.8 billion",
  "evidence": { "para_id":"VW2015_Pfe29ae", "char_start":0, "char_end":97,
                "text":"The Commercial Vehicles/Power Engineering Business Area generated gross cash flow of €2.8 billion" }
,
  "table_id": "VW2015_T3b393d"
}
```

Figure 17: Enhanced prompting example (Lookup QA)

```
Table example
Table:
    ["Gross cash flow", "2795.0", "2201.0"],
    ["Change in working capital", "810.0", "-1255.0"],
    ["Cash flows from operating activities", "3605.0", "946.0"],
    ["Cash flows from investing activities attributable to operating activities", "-2475.0", "-1534.0"],
    ["Net cash flow", "1129.0", "-588.0"]
Header: ["€ million", "2015", "2014"]
Q: What is the percentage change in gross cash flow in the Commercial Vehicles/Power Engineering Business Area in
2015 compared to 2014?
A: Read the values for 2015 and 2014: 2795.0 and 2201.0. Their difference is 594. Dividing 594 by 2201.0 yields about 0.27, or 27%.
Final answer (JSON):
{
  "qa_id": "292903ef00",
  "q_type": "arithmetic",
  "arith_type": "pct_change",
  "question": "What is the percentage change in gross cash flow in the Commercial Vehicles/Power Engineering Business Area in
2015 compared to 2014?",
  "answer_text": "27.0%",
  "answer_value": 27.0,
  "unit": "%",
  "program": [
    { "op": "read", "as": "gross_2015", "cell": { "table_id": "VW2015_T3b393d", "row": "Gross cash flow", "col": "2015",
"value": "2795.0" }, "value": "2795.0" },
    { "op": "read", "as": "gross_2014", "cell": { "table_id": "VW2015_T3b393d", "row": "Gross cash flow", "col": "2014",
"value": "2201.0" }, "value": "2201.0" },
    { "op": "sub", "inputs": [ "gros_2015", "gross_2014" ], "as": "numerator", "result": 594.0 },
    { "op": "div", "inputs": [ "numerator", "gross_2014" ], "as": "ratio", "result": 0.27 },
    { "op": "mul", "inputs": [ "ratio", 100 ], "as": "percentage", "result": 27.0 }
  ],
  "evidence": [
    { "table_id": "VW2015_T3b393d", "row": "Gross cash flow", "col": "2015", "value": "2795.0" },
    { "table_id": "VW2015_T3b393d", "row": "Gross cash flow", "col": "2014", "value": "2201.0" }
  ],
  "table_id": "VW2015_T3b393d"
}
```

Figure 18: Enhanced prompting example(Arithmetic QA)

Table example

```
Table:
    ["Deliveries (thousand units)", "5823.0", "6119.0", "-4.8"],
    ["Vehicle sales", "4424.0", "4583.0", "-3.5"],
    ["Production", "5898.0", "6156.0", "-4.2"],
    ["Sales revenue (€ million)", "106240.0", "99764.0", "6.5"],
    ["Operating result before special items", "2102.0", "2476.0", "-15.1"],
    ["as % of sales revenue", "2.0", "2.5", ""]
Paragraph:
"In a continuously challenging market environment, the brand delivered 5.8 million vehicles in the reporting period.
The decrease of 4.8% year-on-year was attributable in particular to the markets in Brazil, China and Russia..."
Q: Analyze the factors that impacted on the brand’s financial performance in 2015.
A: Sales revenue increased from 99,764 to 106,240 million (6.5%), but operating profit decreased from 2,476 to 2,102
million (15.1%). The text explains that promotion activities related to the emissions issue contributed to the decline.
Final answer (JSON):
{
  "qa_id": "e0138b014d",
  "q_type": "insight",
  "question": "Analyze the factors that impacted on the brand’s financial performance in 2015.",
  "gold_answer": {
    "answer": "band’s sales revenue rose 6.5% to €106,240 million in 2015, but operating profit fell 15.1% from €2,476 million
to €2,102 million due to market-related promotion activities stemming from the emissions issue.",
    "claims": [
      {
        "type": "table",
        "claim_text": "Sales revenue increased from €99,764 million in 2014 to €106,240 million in 2015 (↑6.5%).",
        "evidence": { "table_cells": [
          { "table_id": "VW2015_T19a389", "row": "brand", "col": "SALES REVENUE 2014", "cell_text": "99764" },
          { "table_id": "VW2015_T19a389", "row": "brand", "col": "SALES REVENUE 2015", "cell_text": "106240" }
        ] }
      },
      {
        "type": "table",
        "claim_text": "Operating profit decreased from €2,476 million in 2014 to €2,102 million in 2015 (↓15.1%).",
        "evidence": { "table_cells": [
          { "table_id": "VW2015_T19a389", "row": "brand", "col": "OPERATING PROFIT 2014", "cell_text": "2476" },
          { "table_id": "VW2015_T19a389", "row": "brand", "col": "OPERATING PROFIT 2015", "cell_text": "2102" }
        ] }
      },
      {
        "type": "text",
        "claim_text": "Market-related promotion activities resulting from the emissions issue negatively impacted the operating
result.",
        "evidence": { "text_spans": [
          { "para_id": "VW2015_Ped3427", "char_start": 420, "char_end": 490,
                  "text": "market-related promotion activities resulting from the emissions issue" }
        ] }
      }
    ]
  },
  "table_id": "VW2015_T19a389"
}
```

Figure 19: Enhanced prompting example (Insight QA)

F Experiment setup 3 - multi-agent

For completeness, we also evaluate a lightweight multi-agent configuration, denoted **S3**, which decomposes inference into a *Planner* → *Solver* → *Verifier* pipeline, no external tools or iterative loops are involved.

F.1 Method Overview

Planner. Given the table and paragraphs, the Planner predicts (i) the question family (LOOKUP, ARITHMETIC, or INSIGHT) and (ii) a coarse set of *focus regions*, such as relevant table rows or paragraphs. The abbreviated prompt is shown in Figure 20.

Solver. Conditioned on the predicted family and focus regions, the Solver generates a structured JSON answer conforming to the family-specific schema. The Solver uses the same typed prompts as in S2 D.

Verifier. The Verifier serves as a single-pass reflection step that inspects the Solver's JSON output and applies only minimal local corrections. It checks that all cited table and paragraph identifiers exist in the pack, that table coordinates and text spans are within bounds and match the canonical source, that units are present and normalized, and that arithmetic programs correctly recompute the numerical answer under high-precision execution. For INSIGHT items, it further enforces atomic claims, prohibits invented numbers, and ensures the presence of both table-grounded and text-grounded evidence. If the answer is valid, it is returned unchanged; otherwise, the Verifier produces a minimally repaired JSON structure. The prompt is shown in Figure 21.

F.2 Additional Results

Table 9 reports full results for the S3 Planner–Solver–Verifier setup across all models. Overall, S3 yields mixed and model-dependent effects. On Lookup tasks, performance often decreases relative to S1/S2, with several models showing drops in VU-EM and semantic scores, likely due to Planner misrouting or unnecessary evidence adjustments. Arithmetic accuracy remains generally stable—QWEN-2.5 32B and GPT-4O continue to perform strongest—but S3 provides no systematic improvements. For INSIGHT, S3 occasionally improves evidence grounding or claim F1 (e.g., GPT-4O and DEEPSEEK), but semantic fidelity does not consistently increase, and several models show declines. These patterns indicate that the lightweight agentic pipeline introduces additional structure without reliably enhancing reasoning or grounding, and we therefore exclude it from the main comparison.

Model	Setup	Lookup								Arithmetic				Insight					
		VU-EM		Evi. F1		Sem. F1		Evi. F1		Acc.		Evi. F1		Sem. F1		Claim F1		Evi. F1	
		Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT	Base	FS+CoT
LLAMA-3.1 8B	S3	0.53	0.56	0.84	0.85	0.55	0.53	0.05	0.05	0.49	0.46	0.67	0.73	0.37	0.41	0.46	0.54	0.31	0.30
LLAMA-3.3 70B	S3	0.83	0.79	0.97	0.98	0.59	0.60	0.10	0.11	0.81	0.75	0.90	0.94	0.38	0.41	0.48	0.51	0.43	0.41
QWEN-2.5 32B	S3	0.47	0.80	0.98	0.97	0.70	0.69	0.09	0.10	0.91	0.75	0.94	0.94	0.38	0.38	0.54	0.53	0.46	0.39
DEEPSEEK 32B	S3	0.68	0.84	0.97	0.93	0.57	0.61	0.08	0.08	0.83	0.86	0.91	0.94	0.39	0.44	0.47	0.53	0.39	0.40
GPT-4O	S3	0.80	0.84	0.99	0.99	0.70	0.73	0.17	0.17	0.92	0.89	0.95	0.94	0.43	0.46	0.53	0.58	0.46	0.48

Table 9: Results for all models under the S3 setup only, across all Lookup, Arithmetic, and Insight metrics.

```
Role: Planner for a finance QA system (annual report domain)
Decide which question family applies and where to focus inside the given pack.
Families:
- lookup_table ⟶ question asks for a numeric fact from a table cell
- lookup_text ⟶ question asks for a textual explanation from paragraphs
- arithmetic ⟶ question requires computing a value from ≥2 table cells
- insight ⟶ question requires multi-sentence reasoning across table and text
Output (STRICT JSON):
{
  "family": "lookup_table"|"lookup_text"|"arithmetic"|"insight",
  "focus": {
    "table": {"table_id":"string"|null, "rows":["..."], "cols":["..."]},
    "text": {"para_ids":["..."]}
  },
  "reason": "short natural-language justification (1–2 sentences)"
}
Use only the provided PACK and QUESTION—no outside knowledge.
Return JSON directly (no prose, no Markdown).
Provided context:
PACK: JSON object containing table(s) and related paragraphs.
QUESTION: user query in natural language.
```

Figure 20: Experiment setup 3 prompt (Planner)

```
Role: Verifier for a finance QA system (annual report domain)
Input includes PACK, QUESTION, and the model ANSWER.
Your job: (1) verify schema and grounding, (2) minimally repair problems in one pass.
Families and required payloads:
- lookup_table ⟶ {answer_text, unit|null, value_canonical|null, evidence{table_id,row,col,cell_text}}
- lookup_text ⟶ {answer_text, evidence{para_id,text}}
- arithmetic ⟶ {answer_text, answer_value, unit|null, program[read/add/sub/mul/div...], evidence[...]}
- insight ⟶ {answer, claims[type(table|text), claim_text, evidence{table_cells[], text_spans[]}]}
Verification checks:
- All cited table_id / para_id exist in PACK; spans are in-bounds; use labeled row/col names.
- Units included and canonicalized.
- Arithmetic: program must recompute answer_value precisely; answer_text may be rounded.
- Insight: claims atomic and factual; ≥1 table claim and ≥1 text claim; no invented numbers.
Output format:
If valid:
{ "final": <original answer JSON>, "repaired": false }
If repaired:
{ "final": <corrected answer JSON>, "repaired": true }
Return STRICT JSON only (no prose, no code fences).
Provided context:
Family: {family}
PACK: table + paragraph data
QUESTION: user query
ANSWER: model-generated JSON answer
```

Figure 21: Experiment setup 3 prompt (Verifier)