Onthos

The case for ontology: Efficient question answering for scalable AI systems for financial markets

Authors

Onthos (formerly Theia Insights), Cambridge, United Kingdom

Correspondence: solutions@onthos.com

Abstract

Accurately answering investment research questions such as 'which companies increased their investment in AI over the past two years?' requires compiling and combining information from the entire universe of companies. We compare three approaches to question answering over a corpus of company filings: reading everything at question time, Retrieval-Augmented Generation (RAG), and ontology-based structured retrieval (Onthos). Using 59 annual reports from 30 US listed companies and a dataset of 84 questions, we find that read-everything can produce strong answers but is too slow and expensive to scale; RAG sharply reduces the context processed but suffers in quality, especially recall; and filter-first Onthos produces the most complete answers at significantly lower cost and latency. We argue that information reused across questions should be understood once at ingestion, represented in an ontology and applied consistently across the universe, with RAG reserved as a second-stage tool for retrieving source passages.

Contents

1 Introduction

Consider a question an investor might ask: 'which companies increased their investment in AI over the past two years?' Answering it means examining every company, judging what counts as AI investment and comparing disclosures across the two years. The relevant information can be sourced from annual reports or earnings transcripts; it may also be in a firm's internal documents such as analyst research or expert call summaries. An answer that misses a company is as much a failure as one that misjudges a company. The central problem is: how can we select information that is complete while being efficient and scalable? We studied three approaches:

Method A: Direct LLM – read everything at question time

As Large Language Models (LLMs) allow increasingly longer context windows (for example, Claude Opus supports 1 million tokens), a natural question is whether it is possible to combine all the information into large prompts and rely on a model to read everything and provide the answer. Because the entire corpus exceeds 1 million tokens, we split the documents into chunks that can fit into an LLM's context window budget, ask the model to assess every chunk, and aggregate partial answers into a final answer. However, as commercial LLM cost scales with the number of input tokens, this is not a feasible solution at scale. Additionally, large volumes of irrelevant information may 'distract' model reasoning and increase the likelihood of an inaccurate answer.

Method B: RAG

To balance cost and accuracy, approaches such as Retrieval-Augmented Generation (RAG) are selective about which context is provided to the model, using a cheaper and more scalable method to filter relevant context from the larger corpus. Typically, RAG systems chunk and embed the corpus with an embedding model. At question time, the system calculates the semantic similarity between each chunk and the query, and feeds the top N chunks to an LLM. The LLM then reads text from those chunks and extracts an answer.

Using RAG, the model doesn't need to read everything at question time, but still needs to interpret the top N chunks with every question. Document understanding happens at query time rather than at ingestion. The accuracy and completeness of the answer depend on how widely the evidence is distributed across the corpus, and on how well vector similarity can select relevant information.

Method C: Ontology-based structured retrieval (Onthos)

Onthos moves document understanding upstream using an ontology method. In our implementation, Onthos creates an ontology that reflects the information structure of the corpus. It then classifies, extracts and summarises all documents at ingestion, based on this ontology, and stores the information as structured and linked fact bases such as company activities, themes and summaries.

At question time, the question is translated into a structured query, which executes over the knowledge base and retrieves relevant facts across the whole universe. Because all documents are classified using

a shared ontology, fact retrieval is not limited to top N passages. Finding 1,000 companies with AI investment is nearly as fast and cheap as finding 10 companies, as long as "investment" and "AI" exist as labels in the ontology. The retrieved facts are then fed to an LLM to produce a final answer.

To compare the effectiveness of these three approaches, we conducted an experiment using a small corpus of public-company filings. The filings provide a controlled and auditable test, but the principles apply to internal and proprietary corpora as well.

We found that reading everything (method A) could produce a strong answer, but it was too slow and expensive to scale. RAG (method B) sharply reduced the amount of context processed compared to method A, but suffers in quality, especially in recall. Onthos (method C) produced the most complete answers with significantly lower cost and latency.

2 Experiment

2.1 Dataset and model setup

Corpus: we curated a corpus containing 59 annual reports from 30 US listed companies, covering approximately two fiscal years per company with a total of 7.2 million tokens of filing text. The 30 companies cover different industries including semiconductors, cloud infrastructure, nuclear generation, banks, energy, retail, distribution and industrial conglomerates.

Question bank: we created 84 questions representing four types of research commonly conducted by investment teams:

- **Financial screening & ranking (32):** for example, "How many companies increased capex by more than 50% year on year?"
- **Thematic screening & ranking (31):** for example, "Which companies have material exposure to both AI and power generation?"
- **Narrative questions (11):** for example, "Which companies increased data centre capex and cited power availability as a risk?"
- **Evidence & citation questions (10):** for example, "Which companies disclose customer concentration above 10%, and what filing text supports each answer?"

LLM models: we tested six open-weight and commercial frontier models: gpt-oss-120b, DeepSeek V4 Pro, Sonnet 5, Opus 5, GPT-5.6 Luna and GPT-5.6 Sol (models available as of August 2026). Where models were compared directly, they were tested on the same questions and universe. Not every model was tested in every experimental condition: method A was tested on representative questions; the narrower RAG condition (top 8) was tested on all models, whereas the wider RAG condition (top 40) was tested with a subset of models (see below).

2.2 Setup for the three methods

All three methods used the same question set, company universe, response format and scoring method. We recorded answer accuracy, input context, latency and cost. Every answer was required to use the context provided by the system instead of using an LLM's memory.

- **Method A – Read everything.** Because the 7.2-million-token corpus exceeded the context window, it was divided into 8 or 9 size-balanced chunks. One model call read each chunk and a final call combined the partial answers.
- **Method B – RAG.** Filing text was split into passages of up to 1,200 characters with 200 characters of overlap, giving 34,821 indexed chunks. Splits were made at paragraph or sentence breaks where possible, so passages did not start or end mid-word. Each chunk was embedded with Cohere Embed v4. At query time we ran dense retrieval and BM25 keyword search side by side, taking 100 candidates from each, merged them with reciprocal rank fusion and passed the top 24 to Claude Haiku 4.5, which reranked them and kept 8 for the answer model. To test the effect of context size we also ran a wider condition that fused 80 candidates and kept 40.
- **Method C – Onthos.** The corpus is processed using Onthos' proprietary ontology, including multi-dimensional business activity summary creation (products & services, revenue, capex, partnerships, business models etc.) and industry and theme classification (Theia Insights Industry Classification). At query time, we tested two approaches: a full-context condition and a filter-first condition. In the full-context condition, we feed the full ontology fact base of all companies to the LLM. In the filter-first condition, a first model call mapped the question onto a selection of relevant ontology labels (e.g. the question of AI investments would select "capex" and "AI" and related themes); a second LLM call received only companies with a non-zero weight on the selected labels.

2.3 Evaluation

We evaluated each question type separately. Screening questions (in types 1 and 2) were measured using precision, recall and F1; ranking questions added Kendall's tau-b among shared companies; count questions required an exact answer. For evidence questions, we evaluated company selection separately from whether the cited passage supported the answer. Each question was executed three times to check model consistency even at temperature zero. The result tables show the average score from the three runs.

Reference answers: financial reference answers were manually checked against as-reported XBRL values from the audited filings. Narrative questions were evaluated against manually curated reference answers. Evidence questions required the answer to be directly supported by raw text from the source filing. The thematic reference answers were derived from the taxonomy underlying the Theia Insights Industry Classification. Because the ontology-based method uses the same taxonomy, these scores measure agreement with the TIIC classification rather than agreement with a fully independent thematic ground truth. We address this limitation when interpreting the results.

3 Results

The results address two aspects: cost and quality. Which methods scale economically, and how does answer quality in terms of accuracy and recall compare?

3.1 Cost and scalability

We tested method A (read-everything) using two representative questions on DeepSeek V4 Pro, costing ∼$5 at ∼200 seconds per question. The first question processed 6.0 million input tokens across ten calls, cost $5.44 and took 146 seconds with four calls running in parallel. The second processed 5.1 million tokens across nine calls, cost $4.64 and took 221 seconds. The number of tokens scales linearly with the number of documents/companies for each question. Table 2 extrapolates tokens and cost to larger universes for each method; Table 3 estimates the dollar cost across different models for the 30-company corpus. Cost is also affected by model choice (Table 3), where at the high end, Opus 5 would cost ∼$132–$146 for a single question on 30 companies for method A.

In comparison, Onthos and RAG cost a fraction of method A. Filter-first Onthos averaged 2,720 input tokens per question (∼0.04% of cost compared to method A), similar to top-8 RAG at 2,433 and below top-40 RAG at 8,645 (Table 1).

Table 1 Mean input per question by method

Method	Mean input tokens	Cost per 1,000 questions
Read everything (method A)	∼5.5 million	∼$3,000
Onthos, full context	25,590	$5.2
RAG, top 40	8,645	$2.1
Onthos, filter-first	2,720	$0.7
RAG, top 8	2,433	$0.8

Token counts are means over all models and question groups. Costs are for GPT-5.6 Luna on the 31 thematic questions, as the set of models tested differed by condition; the read-everything cost is the estimate from Table 3.

Table 2 Estimated cost per question by company universe and method (GPT-5.6 Luna)

	Read everything		RAG, top 8	Onthos, filter-first
Company universe	Input tokens	Cost	Cost	Cost
30	∼5.5 million	∼$3	<$0.001	<$0.001
500	∼92 million	∼$50	<$0.001	<$0.001
3,000	∼550 million	∼$300	<$0.001	<$0.001
50,000	∼9.2 billion	∼$5,000	<$0.001	<$0.001

Read-everything is extrapolated linearly from the measured 30-company runs, assuming similar filing length per company. RAG cost is fixed by the top-N budget. Filter-first Onthos cost depends on the number of companies matching the query rather than the size of the universe; shown here for a screen returning a similar number of matches.

The contrast in Table 2 is the central point. Read-everything cost grows with the universe. RAG cost does not, because input is fixed by the top-N budget, but the price is that the model sees an ever smaller share of the universe. Filter-first Onthos cost also stays flat, since it grows with the number of companies that match the selected labels rather than with the universe: a screen that returns 50 companies from 50,000 costs about the same as one that returns 50 from 30.

Table 3 Estimated cost per question by model for the 30-company corpus without filtering (read-everything)

Model	Estimated cost per 30-company question
GPT-5.6 Luna	~$3
DeepSeek V4 Pro	$5.04
Sonnet 5	~$26–$29
GPT-5.6 Terra	~$30
GPT-5.6 Sol	~$60
Opus 5	~$132–$146

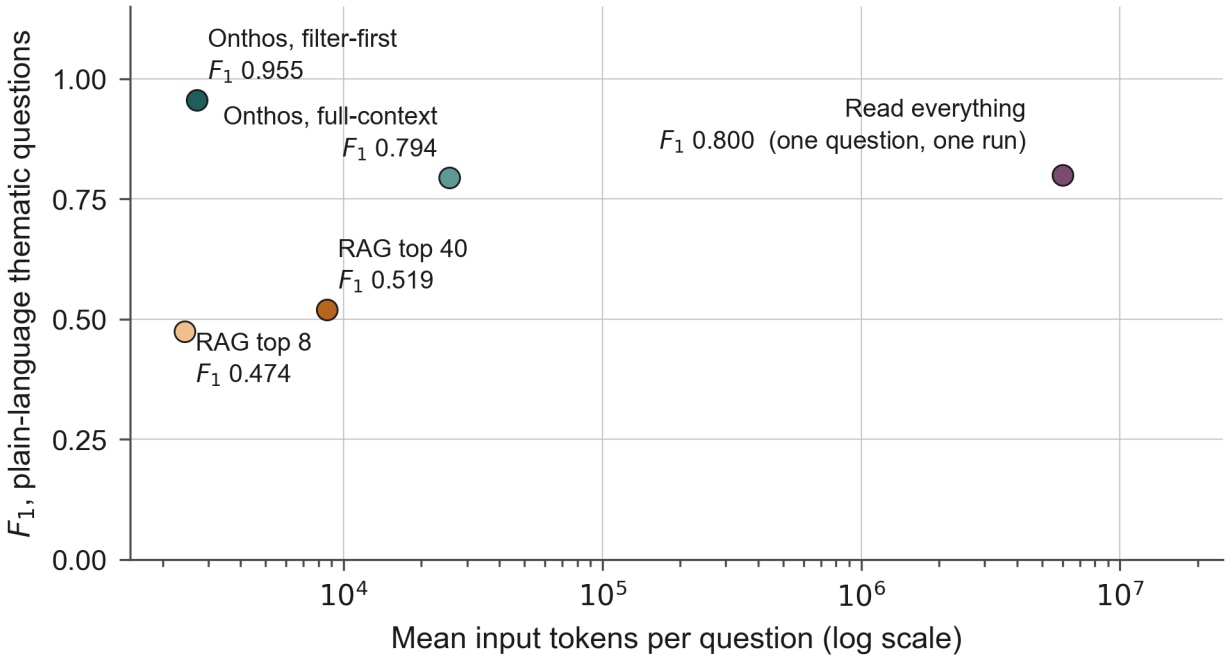

Figure 1 Input tokens and F1. RAG and filter-first Onthos use GPT-5.6 Luna on the same 31 thematic questions. Read-everything is based on two representative questions.

3.2 Answer accuracy and recall

On the 31 plain-language thematic questions, filter-first Onthos achieved F1 scores of 0.900–0.977 across five models, against 0.432–0.533 for top-8 RAG. With GPT-5.6 Luna, F1 was 0.955 for filter-first Onthos, 0.794 for full-context Onthos, 0.519 for top-40 RAG and 0.474 for top-8 RAG. Filter-first matched or improved on full-context Onthos in four of five models; the exception was Sol, by 0.008. It was also the most repeatable condition: the share of questions where three repeats disagreed was 0.00–0.26 for

filter-first, against 0.03–0.58 for full-context Onthos and 0.03–0.52 for RAG. A smaller, more relevant context appears to reduce variance as well as cost.

Filter-first was evaluated on the thematic questions only. The ontology snapshot built for this study carries thematic and narrative fields but not fiscal-year financial figures or source-passage links, so the financial and evidence groups fall outside what this snapshot can express. This is a property of the study build rather than of the architecture; the conclusions return to it.

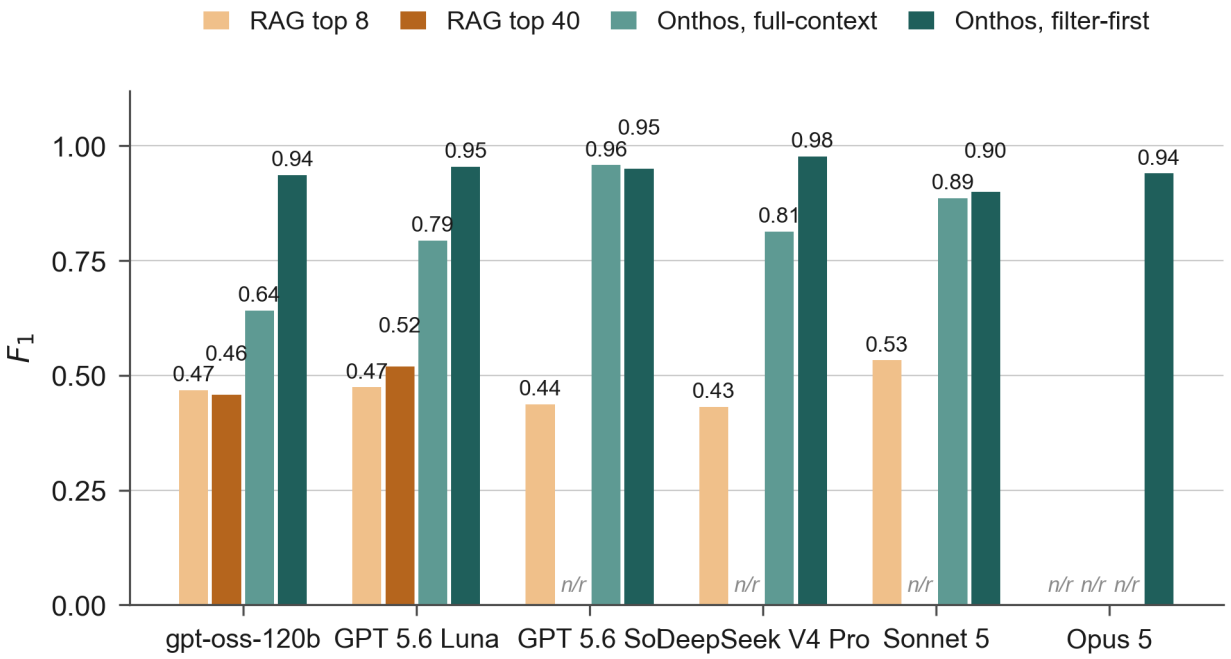

Figure 2 F1 on the 31 plain-language thematic questions. Filter-first Onthos outperformed top-8 RAG for every paired model.

The difference lies in how many companies the model actually sees. Top-8 RAG exposed the model to 4.0 of 30 companies on average. Raising the budget to top 40 worked on its own terms: visible companies rose to 11.7, and the share of correct answer companies present in the context nearly doubled, from 0.424 to 0.803. The answers barely moved. Recall rose from 0.317 to 0.432, precision fell from 0.763 to 0.719, and input grew 3.6 times. The models also stayed conservative, naming 3.6 companies on average against a true mean of 5.5. In the clearest case, the same question produced an identical three-company answer at top 8 and at top 40, across all three repeats, while the context grew from roughly 2,000 to nearly 8,000 tokens.

The top-N design of RAG means the system can never know whether an answer to a screening question is complete. Screening needs a judgement on all the companies in the corpus. When a company does not appear in the retrieved passages, the model cannot tell a true negative from a retrieval miss. Meanwhile the extra passages introduce near-matching distractors, which is why precision fell as recall rose. Onthos structured retrieval avoids this by construction. Because the whole corpus was classified upstream, a company absent from the filtered context is absent because its exposure on the selected labels is zero.

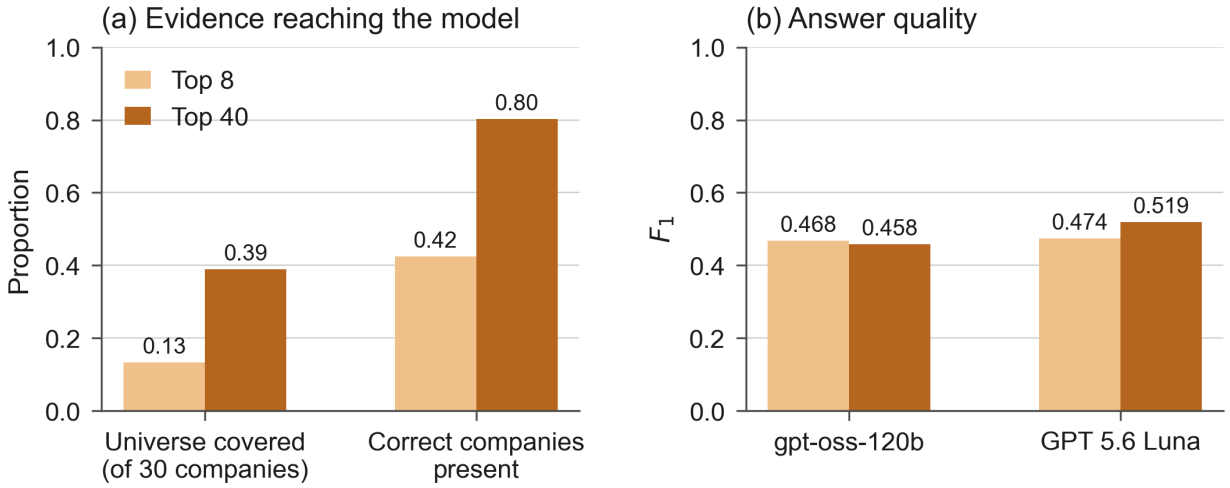

Figure 3 Top-40 RAG exposed more companies and more correct evidence, but answer recall improved modestly and precision fell.

4 Discussion

This study focused on cost at query time and did not consider ingestion cost. Both RAG and Onthos have an ingestion cost, and both are low. For RAG, embedding the 30-company corpus for retrieval cost about $0.82, and ontology extraction over the same 59 filings is of the same order: using models such as Amazon Nova or Anthropic Haiku it is under 10 cents per filing, excluding the cost of building and maintaining the extraction pipeline and the ontology.

The ontology should not be fixed. In the Onthos design, the system learns from new questions and documents to update the label sets or even the ontology structure. Once added at ingestion, every future question of a similar shape can be answered cheaply and reliably, and the cost of the gap is paid once rather than with every new question.

5 Conclusions

This study evaluated different methods for question answering for financial research: read-everything, RAG and ontology. Using a corpus of 59 filings from 30 companies, we found that read-everything for each question is operationally impractical due to its high cost and low speed. RAG reduced the amount of text sent to the model, but top-N passage retrieval limits answer completeness. Filter-first Onthos produced the strongest results with higher F1 and lower cost.

Overall, we argue that information that will be reused across questions should be understood once, represented in an ontology and applied consistently across the universe. At question time, the system should translate the user's intent into validated filters rather than start again from raw documents. RAG remains valuable for retrieving source passages, but it should be used as a second-stage tool rather than for context selection from the full universe.