NAVALIA: ADVANCING COWORK AGENTS THROUGH VERIFIABLE TASK SYNTHESIS AND LONG-HORIZON POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) agents are evolving from conversational assistants into Cowork Agents capable of autonomous planning, tool use, and complex task execution. Financial analysis requires these agents to integrate cross-document evidence and perform multistep numerical reasoning, but expert task authoring is costly, while direct LLM synthesis struggles to produce challenging tasks with reliable answers. We introduce Navalia, a framework for Cowork Agents that combines verifiable task synthesis with long-horizon post-training. Using corporate financial reports as its data source, Navalia semantically selects supporting evidence from complete fact sets and constructs executable computation chains to generate analytical tasks with traceable evidence and actual data dependencies between steps. We further use independent LLM review and programmatic checks to verify evidence consistency and answer correctness. The verified tasks are then used for trajectory collection, cold-start supervised fine-tuning (SFT), and reinforcement learning (RL) to improve long-horizon task execution. Navalia, initialized from Qwen3.6-35B-A3B, achieves consistent gains over the base model across all five benchmarks: OfficeQA Pro, FinSearchComp, Finance Agent Benchmark, APEX-Agents-AA, and Workspace-Bench-Lite. These results demonstrate that Navalia improves evidence integration, tool use, and long-horizon task execution capabilities and generalizes to broader Cowork settings, including general office tasks.

1 INTRODUCTION

Agents based on large language models (LLMs) (Wang et al., 2024) are evolving from conversational assistants (Bai et al., 2022; Touvron et al., 2023) into Cowork Agents that autonomously plan, invoke tools, and deliver verifiable results to achieve user goals (An et al., 2026; Vidgen et al., 2026; Lab et al., 2026). Financial analysis is an important application of such agents (Chen et al., 2022b; Zhu et al., 2021; Islam et al., 2023): agents must locate evidence in lengthy documents and complex tables, identify differences in metric definitions and reporting scope across companies and reporting periods, and perform multistep numerical analysis through repeated tool interactions. Document retrieval, evidence integration, and computation are interdependent, and errors at any stage may affect the final conclusions. Evaluation results on OfficeQA Pro (Opsahl-Ong et al., 2026) also reveal substantial limitations in agents' ability to retrieve evidence and perform numerical reasoning over large collections of financial documents.

Improving these capabilities requires the scalable construction of training tasks with both analytical depth and verifiability. A reliable financial analysis task must not only have sufficient analytical depth but also specify its source documents, supporting evidence, computational dependencies, and reference answer, while ensuring consistency in reporting periods, metric definitions, units, and algorithmic parameters. Relying on domain experts to construct and verify individual tasks is costly and difficult to scale to meet post-training data demands; allowing LLMs to freely generate question–answer pairs directly from raw document environments may lead them to mix data with inconsistent metric definitions or reporting scope, make calculation errors (Chen et al., 2022a), or favor simple lookup and basic arithmetic (Tao et al., 2026). Even when questions contain multiple steps, those steps may lack actual data dependencies and thus fail to provide effective training signals for

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

multistep analysis (Trivedi et al., 2022). In addition, systematic and stable post-training on these reliable tasks is needed to improve agents' cross-document evidence integration, multistep numerical reasoning, and long-horizon task execution capabilities (Liu et al., 2026). However, systematic public documentation covering data construction through complete post-training pipelines for Cowork Agents remains limited, making related methods difficult to reproduce and extend.

To this end, we introduce **Navalia**, a framework for Cowork Agents that combines verifiable task synthesis with long-horizon post-training. Figure 1 provides an overview of the pipeline. Navalia constructs a dual-view representation of documents and facts from corporate financial reports, using documents to provide the interactive environment and traceable facts to support task construction and verification. To reduce evidence omissions caused by rule-based prefiltering, an LLM semantically selects supporting evidence from the complete fact sets of candidate documents and checks metric definitions and reporting scope. Building on this evidence, Navalia constructs executable computation chains with actual data dependencies and combines deterministic execution of complex computations, independent LLM review, and programmatic checks to verify consistency among the question, evidence, computation process, and reference answer. This provides analytical tasks with traceable evidence, verifiable answers, and actual multistep dependencies for post-training.

Navalia adopts a two-stage post-training pipeline of cold-start supervised fine-tuning (SFT) followed by reinforcement learning (RL) (Shao et al., 2024; Guo et al., 2025). We collect teacher trajectories using a lightweight ReAct Harness (Yao et al., 2022) with FIFO context management (Seed, 2026) and filter them to construct the cold-start SFT dataset. After the cold start, we adapt the same Harness for RL, focusing training on solvable tasks that the model cannot yet solve reliably. Under FIFO context management, a complete interaction trajectory is divided into multiple segments. We compute trajectory-level advantages from final task outcomes and share them across the segments, enabling end-to-end optimization of the entire trajectory so that earlier retrieval and analysis behaviors can also learn from the final feedback.

We build Navalia on Qwen3.6-35B-A3B and evaluate it on five benchmarks covering document analysis, financial search, and general office tasks. Navalia achieves clear improvements over the base model on all five benchmarks and reaches performance levels comparable to those of leading open-source and closed-source models on several evaluations. Although Navalia's training tasks and interaction environments are all constructed from publicly disclosed corporate financial reports, the trained model still achieves substantial gains on open-domain financial search benchmarks (Hu et al., 2026; Bigeard et al., 2025) and long-horizon task benchmarks for general office settings. On APEX-Agents-AA (Vidgen et al., 2026), which covers investment banking, management consulting, and corporate legal tasks, and Workspace-Bench-Lite (Tang et al., 2026), which focuses on cross-file operations and long-horizon task execution, Navalia improves by 13.20 and 13.07 percentage points, respectively. These results indicate that the retrieval planning, evidence integration, tool use, and long-horizon task execution capabilities strengthened through training on financial-report tasks can transfer to a broader range of Cowork tasks.

Our main contributions are as follows:

- We propose an automated task synthesis pipeline that combines a dual-view representation of documents and facts with semantic evidence selection, executable computation chains, and multi-stage verification to construct analytical tasks with traceable evidence, actual multi-step dependencies, and verifiable answers.
- We develop a complete post-training recipe covering trajectory collection and filtering, cold-start SFT, and RL within a unified interaction framework. For long trajectories segmented under FIFO context management, we share trajectory-level advantages across segments to optimize the entire trajectory using final task outcomes.
- We plan to release model weights and open-source key components of our task construction, post-training, and evaluation pipelines to support reproducibility and further research on Cowork Agents.

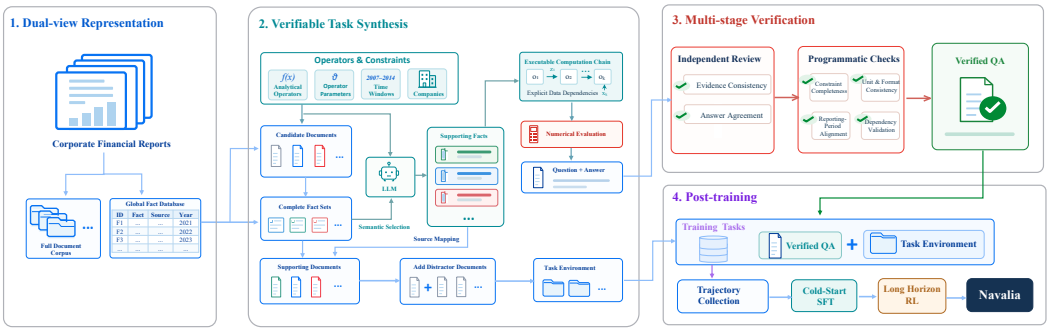


Figure 1: **Overview of Navalia's task synthesis and post-training pipeline.**

2 Verifiable Task Synthesis

Given a collection $\mathcal{D}$ of financial reports from public corporate disclosures, Navalia constructs analytical tasks with traceable evidence and verifiable answers, enforcing actual data dependencies in multistep tasks. Each task is represented as

$$\mathcal{T} = (q, y, \mathcal{C}, \mathcal{F}^+, \mathcal{G}, \mathbf{a}, \boldsymbol{\theta}), \tag{1}$$

where q is a natural-language question and y is the reference answer. The interactive task environment $\mathcal{C}$ is constructed from corporate financial reports and supports evidence retrieval and numerical analysis through multi-turn tool use. The set $\mathcal{F}^+$ contains traceable supporting facts, and $\mathcal{G}$ is an executable computation chain comprising these facts, ordered computational steps, and their intermediate results. The task anchors $\mathbf{a}$ specify the companies, reporting periods, and financial metrics involved, including metric definitions and reporting scope. The operator parameters $\boldsymbol{\theta}$ specify how each operator in the chain is evaluated.

The construction pipeline first samples analytical operators and task anchors, identifies candidate documents, and semantically selects supporting facts from their complete fact sets. It then constructs an executable computation chain with actual data dependencies and computes the reference answer. Finally, it generates the question according to the task constraints and verifies consistency through independent review and programmatic checks. By constraining question generation using traceable supporting facts and explicit computations, the pipeline produces financial analysis tasks with traceable evidence and reproducible answers.

2.1 Dual-View Representation of Documents and Facts

For each financial report $d \in \mathcal{D}$, Navalia constructs two complementary representations: a document view that supports autonomous retrieval and analysis by the agent, and a fact view for task construction and validation:

$$d \longmapsto \big(r(d), \mathcal{F}(d)\big). \tag{2}$$

We parse financial reports from different sources and formats into Markdown to obtain the document view $r(d)$, preserving body text, headings, and table structure. The resulting documents form a corpus that provides the document resources the agent accesses through tools in the interactive task environment $\mathcal{C}$. The fact view $\mathcal{F}(d)$ consists of structured facts extracted from the same document. Each fact records a numerical value, unit, source location, table context, reporting period, metric definition, reporting scope, and version information. Records with incomplete numerical, temporal, or unit information are excluded. The fact sets from all documents are aggregated into a global fact database for task construction and validation:

$$\mathcal{F} = \bigcup_{d \in \mathcal{D}} \mathcal{F}(d). \tag{3}$$

2.2 Operator-Driven Task Design and Environment Construction

Navalia uses analytical operators, task anchors, and operator parameters to jointly determine the analytical objectives and data requirements of each task. Let $\mathcal{O}$ denote the collection of predefined

analytical operators and their composite templates, covering aggregation across reporting periods, trend comparison, statistical analysis, regression, numerical transformations, and cross-company analysis. Each operator or template $o \in \mathcal{O}$ specifies the required data types, computation rules, output format, and parameter space. Composite templates connect operators with compatible inputs and outputs to form multistep tasks with computational dependencies.

Navalia first samples an operator or composite template, then samples task anchors conditioned on the operator and fact database, and operator parameters conditioned on the operator:

$$o \sim p(o), \qquad \mathbf{a} \sim p(\mathbf{a} \mid o, \mathcal{F}), \qquad \boldsymbol{\theta} \sim p(\boldsymbol{\theta} \mid o). \tag{4}$$

Navalia then checks document coverage under the sampled configuration. Time-series tasks require coverage of the specified consecutive reporting periods, cross-company tasks require comparable data from different companies for the same periods, and multistep tasks require coverage of all steps' data requirements. Configurations that do not satisfy these requirements are resampled. For configurations that pass these checks, Navalia identifies the candidate document set $\mathcal{D}_{\text{cand}}$ and samples distractor documents from other reporting periods of the same companies and from peer companies. Together, the candidate and distractor documents form the document resources available in the interactive task environment $\mathcal{C}$.

2.3 Semantic Selection of Supporting Facts

Navalia reads all facts from the candidate documents $\mathcal{D}_{\text{cand}}$ to form the candidate fact set $\mathcal{F}_{\text{cand}} = \bigcup_{d \in \mathcal{D}_{\text{cand}}} \mathcal{F}(d)$. Navalia does not prefilter facts using keywords or fixed metric-based rules. Instead, an LLM determines whether each fact meets the computational requirements by jointly considering the analytical operator, task anchors, operator parameters, and the fact's reporting period, metric definition, reporting scope, unit, and context. This semantic selection aims to reduce evidence omissions caused by incomplete rule coverage.

When the candidate fact set is large, providing all records in a single model input may exceed the context window and introduce redundant information that interferes with evidence assessment. Navalia therefore first reads each document's complete fact set separately and performs within-document semantic preselection. The facts retained from individual documents are aggregated into $\widetilde{\mathcal{F}}_{\text{cand}}$, with all records retained for documents containing few facts. When preselection is not triggered, we set $\widetilde{\mathcal{F}}_{\text{cand}} = \mathcal{F}_{\text{cand}}$. A reasoning model then jointly assesses the aggregated candidate facts across documents to determine the final supporting fact set:

$$\mathcal{F}^{+} = \text{Select}\left(o, \mathbf{a}, \boldsymbol{\theta}; \widetilde{\mathcal{F}}_{\text{cand}}\right). \tag{5}$$

Each supporting fact retains its identifier, location in the original document, reporting period, unit, metric definition, and reporting scope, making each computational input traceable to its source. If $\mathcal{F}^{+}$ does not cover the raw data required by the target computation, or if the selected facts' metric definitions or reporting scopes do not satisfy the task requirements, we discard the current task instance and resample.

2.4 Executable Computation Chains and Numerical Evaluation

Navalia adopts different solution methods depending on the computational complexity of the analytical operator. For simple single-step tasks involving lookup, summation, rates of change, or averaging, an LLM derives the result directly from the supporting facts. For complex computations involving regression, forecasting, statistical analysis, or numerical transformations, the LLM extracts numerical values from the supporting facts and arranges them according to the operator's input requirements. Predefined numerical routines then execute the computations deterministically using the specified parameters to reduce intermediate errors that may arise when the model performs complex calculations directly. Fixed input formats and computation rules yield consistent results for identical data and parameters.

For multistep tasks, Navalia combines the sampled composite template with the supporting facts to construct an executable chain comprising multiple data extraction or computation steps. The template specifies each step's operator, input sources, and execution order. Task anchors impose consistent constraints on companies, reporting periods, metric definitions, and reporting scope throughout

the chain, while operator parameters specify the computational configuration. Data extraction steps read raw fact values, while computation steps take these values or intermediate results from earlier steps as inputs. Complex steps are executed by the corresponding numerical routines.

For a linear chain containing K computation steps, we write

$$\begin{aligned} z_1 &= o_1\left(\mathbf{x}_1; \boldsymbol{\theta}_1\right), \\ z_k &= o_k\left(z_{k-1}, \mathbf{x}_k; \boldsymbol{\theta}_k\right), \qquad k = 2, \ldots, K. \end{aligned} \tag{6}$$

Here, $\mathbf{x}_k$ denotes the raw supporting data required at step k, o_k and $\boldsymbol{\theta}_k$ denote its operator and parameters, and z_k denotes its output. Later steps must use earlier outputs, excluding nominally multistep tasks without actual data dependencies. The computation chain $\mathcal{G}$ records each step's supporting facts, operator, parameters, input sources, and intermediate results, allowing the final answer to be traced back to the original evidence. Instances are discarded if the data are incomplete, types are incompatible between steps, or numerical evaluation fails. The final chain output undergoes unit conversion, precision adjustment, and format normalization to produce the reference answer y for subsequent question generation and validation.

2.5 Question Generation and Quality Control

Given the computation chain $\mathcal{G}$ and reference answer y, Navalia generates the question q under the constraints imposed by the analytical operators, task anchors, operator parameters, and computation chain. Single-step questions specify the analytical objective, data scope, and computational requirements, while multistep questions describe the required analysis in the execution order of the computation chain and request only the final result. All constraints that affect the answer, including companies, reporting periods, financial metrics, reporting scope, and algorithmic parameters, must be preserved in the question. Requirements concerning unit conversion, precision, or specific output formats must also be explicitly stated.

After question generation, an independent review model derives an answer from q and $\mathcal{F}^+$ without reusing the reasoning produced during generation. The reviewer checks consistency between the question and supporting evidence and compares its independently derived result with the reference answer y according to the requirements stated in the question, identifying errors in evidence identification and numerical computation. Tasks passing this review undergo further programmatic checks, such as verifying that the output of every non-final step in a multistep task is actually used by a subsequent step and checking that the reporting periods specified in the question match those recorded in the supporting facts. Only tasks that pass both independent review and programmatic checks are retained as final task instances $\mathcal{T}$.

3 Long-Horizon Agent Post-Training

3.1 Training Framework Overview

Navalia adopts a two-stage post-training pipeline comprising cold-start supervised fine-tuning (SFT) and reinforcement learning (RL). SFT trajectory collection and online RL rollouts use the same lightweight ReAct harness with FIFO context management. Training samples reproduce the context visible during trajectory generation, with the loss computed only on newly generated model responses, enabling training over long-horizon interactions within a limited context window.

3.2 Cold-Start Supervised Fine-Tuning

For the analytical tasks constructed and verified by the preceding pipeline, we use a strong teacher model to autonomously retrieve evidence, perform numerical analysis, and produce answers in the corresponding document environments, recording the complete interaction trajectories. We first apply trajectory-level filtering based on final-answer correctness, tool-call validity, and a minimum number of interaction turns. We then group trajectories into capability pools according to their analytical operators or composite templates and sample evenly across these pools. For the retained trajectories, we further assess quality at the turn level, masking supervision for low-quality model responses while retaining them as context for subsequent interactions.

Training samples are constructed using FIFO context management with a stride of 10 turns. Each sample contains the task instructions, up to 20 turns of interaction history, and up to 10 new interaction turns. SFT computes the cross-entropy loss only on model responses that pass quality assessment within the new interaction turns, normalized by the total number of supervised tokens. The supervision intervals do not overlap across samples, so each model response that passes quality filtering contributes to the loss in exactly one training sample.

3.3 Long-Horizon Agentic Reinforcement Learning

Navalia uses the cold-start SFT model as its initial policy, reuses the ReAct harness and FIFO context management mechanism from SFT trajectory collection, samples rollouts online, and optimizes the policy using group-relative policy gradients. Before RL, we select training tasks based on the success rates of the cold-start SFT model on sampled rollouts for candidate tasks. We retain tasks with both successful and failed rollouts and discard those for which all sampled rollouts succeed. Tasks for which all sampled rollouts fail are retained only if a stronger model successfully solves them, focusing training on solvable tasks that the current policy cannot yet solve reliably.

For any question-answer pair, the policy model samples G trajectories. The g-th trajectory can be represented as

$$\tau_g = \left(s_{g,1}, \ldots, s_{g,J_g}\right), \qquad g = 1, \ldots, G. \tag{7}$$

Here, J_g denotes the number of segments in the trajectory. We designate the final segment s_{g,J_g} as the Anchor and use this segment to compute the final task reward R_g for the entire trajectory. Group-relative advantages are computed exclusively from the Anchor rewards of the trajectories and shared across all segments of the same trajectory. Through advantage sharing, document retrieval, tool use, and analytical actions in earlier segments also receive learning signals based on the task outcome.

We further adopt a dynamic sampling strategy that selects training data according to the Anchor rewards of trajectories sampled for the same task. Only task groups with nonzero within-group reward variance are retained, increasing the proportion of examples in the training batch that provide effective learning signals. As in SFT, the policy loss applies only to newly generated model responses in each segment, while reused historical context and environment observations are excluded through loss masking. For trajectories terminated because a single model response reaches the generation length limit, their Anchor rewards still contribute to group-relative advantage computation and dynamic sampling statistics, while all generated tokens in the entire trajectory are masked out of the loss, preventing truncation failures from unduly penalizing earlier effective exploration.

Navalia adopts on-policy sampling, with each rollout batch used for only one policy update. Let $\mathcal{M}$ denote the index set of valid tokens in the batch. The policy loss is normalized by the total number of valid tokens $|\mathcal{M}|$:

$$\begin{aligned}\mathcal{L}_{\mathrm{RL}}(\theta) = -\frac{1}{|\mathcal{M}|} \sum_{u \in \mathcal{M}} \min\Big\{ & r_u(\theta)\widehat{A}_u, \\ & \mathrm{clip}(r_u(\theta), 1-\epsilon_{\mathrm{low}}, 1+\epsilon_{\mathrm{high}})\, \widehat{A}_u \Big\}.\end{aligned} \tag{8}$$

Here, y_u is the target token generated by the model, x_u is its generation context, and $r_u(\theta) = \frac{\pi_\theta(y_u|x_u)}{\pi_{\theta_{\mathrm{old}}}(y_u|x_u)}$ is the importance sampling ratio for token y_u. The advantage $\widehat{A}_u$ is the shared Anchor advantage of the trajectory containing token y_u. Token-level normalization across the batch avoids the additional segment weighting introduced by averaging losses separately over segments.

4 Experiments

4.1 Setting

Implementation Details. We initialize Navalia from Qwen3.6-35B-A3B and perform cold-start supervised fine-tuning followed by reinforcement learning. The base model has a native context length of 262,144 tokens. SFT trajectory collection and RL rollouts share a lightweight ReAct harness (Yao et al., 2022) with FIFO context management (Seed, 2026), using a window of up to 30 interaction turns and an update stride of 10 turns. Within each benchmark, models evaluated by us use the same agent harness, system prompt, tool interfaces, and interaction budget.

Evaluation Benchmarks. We evaluate Navalia on five benchmarks: OfficeQA Pro (Opsahl-Ong et al., 2026), which assesses evidence retrieval and numerical reasoning over financial documents (using 133 questions under the Full Corpus setting); FinSearchComp (Hu et al., 2026) and Finance Agent Benchmark (Bigeard et al., 2025), which assess financial search and research capabilities (using the T2/T3 subsets and the 77 public questions from v1.1 and v2, respectively); and APEX-Agents-AA (Vidgen et al., 2026) and Workspace-Bench-Lite (Tang et al., 2026), which assess long-horizon professional task execution and general office tasks involving cross-file operations, respectively. Scoring protocols and harness configurations are detailed in Appendix C.

Comparison Models. We compare Navalia against two groups of models to assess its training gains and relative performance: frontier closed-source models, including GPT-5.4, Claude Opus 4.6, Gemini 3.1 Pro Preview, and Qwen3.7-Plus; and large open-weight models, including DeepSeek-V4-Flash, Kimi-K2.6, GLM-5.2, and Qwen3.8-Flash-Next (Qiu et al., 2026). We also include Qwen3.6-35B-A3B without our post-training as the baseline to quantify the performance gains achieved by Navalia. The comparison includes both results obtained by us and publicly reported results; the corresponding markers are explained in the caption of Table 1.

4.2 Main Results

Table 1 presents the overall performance of the evaluated models on benchmarks covering document-grounded analysis, open-web financial research, and professional work across multiple domains.

Navalia outperforms the baseline on all five benchmarks. Navalia achieves scores of 62.66, 55.80, 57.10, 25.44, and 59.27 on OfficeQA Pro, FinSearchComp, Finance Agent Benchmark, APEX-Agents-AA, and Workspace-Bench-Lite, respectively, improving over Qwen3.6-35B-A3B by 23.06, 14.10, 13.91, 13.20, and 13.07 percentage points. The improvement is most pronounced on OfficeQA Pro, indicating that our task synthesis and post-training pipeline can effectively enhance the model's document retrieval, tool use, and financial analysis capabilities.

Generalization to Broader Cowork Tasks. Navalia's training tasks are constructed from corporate financial reports and require the model to locate information in corporate documents, integrate cross-document evidence, and perform multi-step numerical analysis through repeated tool use. These tasks jointly strengthen retrieval planning, evidence integration, tool use, and long-horizon task execution capabilities. On APEX-Agents-AA, which covers professional work across multiple domains, and Workspace-Bench-Lite, which emphasizes cross-file operations and long-horizon task execution, Navalia improves over the baseline by 13.20 and 13.07 percentage points, respectively. These results indicate that the retrieval planning, evidence integration, tool use, and long-horizon task execution capabilities strengthened by this pipeline generalize across tasks and can transfer to general-purpose Cowork settings.

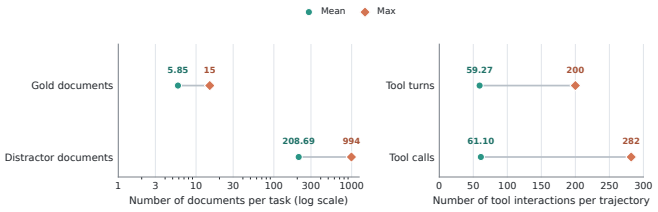
Comparison with Larger and Frontier Models. Despite its smaller parameter scale, Navalia outperforms Qwen3.5-397B-A17B on all five benchmarks and surpasses other strong comparison models on selected benchmarks. Specifically, Navalia outperforms Kimi-K2.6 by 4.01 and 2.61 percentage points on OfficeQA Pro and Finance Agent Benchmark, respectively. It also exceeds Qwen3.5-397B-A17B by 5.40 percentage points on FinSearchComp, Qwen3.7-Plus by 3.02 percentage points on APEX-Agents-AA, and DeepSeek-V4-Flash by 1.19 percentage points on Workspace-Bench-Lite. These results indicate that high-quality task synthesis and systematic post-training can improve the competitiveness of smaller models and narrow the performance gap with larger and frontier models.

4.3 Analysis

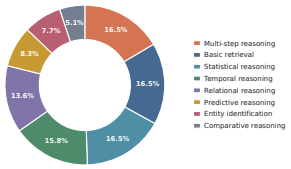
As shown in Figure 2(a), each task in the training data contains an average of 5.85 supporting documents and 208.69 distractor documents, with maxima of 15 and 994, respectively. These statistics indicate that the model must locate and integrate evidence from multiple supporting documents within large, noisy document environments. The corresponding trajectories contain an average of 59.27 tool-use turns and 61.10 tool calls. Such noisy retrieval environments and long-horizon tool interactions help develop the model's cross-document retrieval, multi-turn tool use, and evidence integration capabilities.

Table 1: **Main results on financial research and professional workplace tasks.** All benchmark scores are reported as percentages. The final row reports gains over Qwen3.6-35B-A3B in percentage points. Results marked with * are taken directly from the original papers, technical reports, or official websites; all other results are obtained by us under the unified evaluation setup. When the marker follows a model name, it applies to the entire row; when it follows an individual score, it applies only to that entry. Scores marked with † were obtained using a different document representation or Agent Harness and are included for reference only.

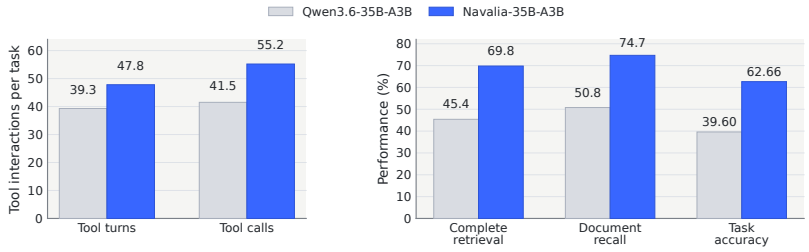
Model	OfficeQA Pro Correctness ↑	FinSearchComp T2&T3 Acc. ↑	Finance Agent Public-77 CB All-Pass ↑	APEX-Agents-AA Pass@1 ↑	Workspace-Bench-Lite Rubric Pass Rate ↑
Proprietary Frontier Models					
GPT-5.4*	50.40	–	–	33.26	52.9
Claude Opus 4.6*	55.60	–	–	33.04	–
Claude Opus 4.7*	46.6†	–	–	–	66.6
Gemini 3.1 Pro Preview*	39.10	–	–	32.01	38.7
Seed2.1 Pro*	70.9†	–	–	–	53.0†
Qwen3.7 Plus	55.39	61.1	56.94	22.42*	55.01
Large Open-Weight Models					
Kimi-K3	67.92	68.8	65.02	37.13	60.56
DeepSeek-V4 Flash	65.66	65.2	58.87	36.43	58.08
Qwen3.8 Flash Next	64.04	70.3	58.48	38.86	61.96
GLM-5.2	63.41	65.7	63.1	33.89	59.34
Kimi-K2.6	58.65	59.1	54.49	29.28	54.96
Qwen3.5-397B-A17B	47.62	50.4	51.82	14.60	50.51
Baseline					
Qwen3.6-35B-A3B	39.6	41.7	43.19	12.24	46.2
Navalia-35B-A3B	**62.66** (+23.06)	**55.80** (+14.10)	**57.10** (+13.91)	**25.44** (+13.20)	**59.27** (+13.07)



(a) Retrieval environment and tool-use statistics



(b) Training capability distribution



(c) Tool use, evidence retrieval, and task performance

Figure 2: **Training data characteristics and model behavior.** The constructed training data is characterized by retrieval-intensive tasks, long tool-use trajectories, and diverse reasoning capabilities. We further compare the tool-use behavior, evidence retrieval, and task performance of Qwen3.6-35B-A3B and Navalia-35B-A3B.

Figure 2(b) presents the capability distribution of the training data. The distribution is relatively balanced, with no single category accounting for more than 16.5% of the data. This helps prevent any particular task category from dominating training and provides diverse training coverage for financial information retrieval, quantitative analysis, and multi-step reasoning.

Table 2: **Training efficiency and downstream performance of trajectory segmentation strategies.** FIFO substantially reduces sample expansion while achieving higher accuracy on OfficeQA Pro.

Segmentation strategy	Training-set statistics			OfficeQA Pro
	Trajectories	Samples ↓	Samples/trajectory ↓	Accuracy (%) ↑
Per-turn sliding window	6,826	276,598	40.52	49.96
FIFO	6,031	**23,523**	**3.90**	**51.56**

Figure 2(c) compares tool use, evidence retrieval, and task performance on OfficeQA Pro before and after training. Following training, Navalia's task accuracy increases from 39.60% to 62.66%. The proportion of tasks for which all supporting documents are retrieved increases from 45.4% to 69.8%, and supporting document recall improves from 50.8% to 74.7%. Meanwhile, the average numbers of tool-use turns and tool calls increase from 39.3 and 41.5 to 47.8 and 55.2, respectively. These results indicate that our task synthesis and post-training pipeline effectively improves the model's long-horizon tool use, evidence retrieval, and financial task-solving capabilities.

To evaluate the effects of different context management methods on training efficiency and downstream performance, we compare per-turn sliding window and FIFO context management under the same limit of 30 visible interaction turns. The former advances the window by one turn and constructs a training sample for each target turn, whereas FIFO updates the window with a fixed stride, reducing context overlap between adjacent samples. Table 2 shows that FIFO reduces the average number of samples per trajectory from 40.52 to 3.90, a reduction of 90.4%, while improving OfficeQA Pro accuracy from 49.96% to 51.56%. These results indicate that FIFO substantially reduces sample expansion while maintaining comparable downstream performance. We therefore adopt FIFO context management for subsequent SFT trajectory collection and reinforcement learning.

5 Limitations and Future Work

We use a lightweight single-agent ReAct harness for SFT trajectory collection and RL rollouts. Context is managed solely through a fixed-window FIFO mechanism, without introducing more complex mechanisms such as trajectory compression, long-term memory, or multi-agent collaboration. The fixed window removes earlier interactions from the visible context, while our current advantage-sharing mechanism, based on final task rewards, does not explicitly distinguish the contributions of actions at different stages to task completion. Nevertheless, Navalia outperforms the baseline on Workspace-Bench-Lite using the Claude Code Harness and APEX-Agents-AA using the Stirrup Agent Harness. These results indicate that our task synthesis and post-training pipeline strengthens the model's document retrieval, evidence integration, and long-horizon tool-use capabilities, and that these improvements transfer across evaluation harnesses. Future work will explore more comprehensive agent harnesses and finer-grained reinforcement learning signals to further improve long-horizon Cowork task execution.

6 Conclusion

We introduced Navalia, a framework for advancing Cowork Agents through verifiable task synthesis and long-horizon post-training. Based on a dual-view representation of documents and facts, Navalia constructs analytical tasks with traceable evidence, actual multistep dependencies, and verifiable answers. We combine this task synthesis pipeline with cold-start SFT and RL. Built on Qwen3.6-35B-A3B, Navalia consistently improves over the base model across all five benchmarks. These results indicate that training on financial-report tasks improves evidence integration, tool use, and long-horizon task execution, with benefits extending to open-domain financial search and broader Cowork tasks.

AI USE STATEMENT

We used LLMs for task synthesis and verification, teacher trajectory generation, and benchmark evaluation, as described in the paper, as well as manuscript translation and language polishing. The authors reviewed the manuscript and take full responsibility for its final content.

REFERENCES

B An, B Li, B Wang, B Zhang, BL Wang, C Feng, C Wei, C Xue, C Zhang, D Ng, et al. Apodex 1.1: Scaling agentic intelligence for complex work. *arXiv preprint arXiv:2608.23283*, 2026.

Fei Bai, Huatong Song, Shuang Sun, Daixuan Cheng, Yike Yang, Chuan Hao, Renyuan Li, Feng Chang, Yuan Wei, Ran Tao, et al. Clawgym: A scalable framework for building effective claw agents. *arXiv preprint arXiv:2604.26904*, 2026.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Antoine Bigeard, Langston Nashold, Rayan Krishnan, and Shirley Wu. Finance agent benchmark: Benchmarking llms on real-world financial research tasks. *arXiv preprint arXiv:2508.00828*, 2025.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022a.

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

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 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*, pp. 6279–6292, 2022b.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Liang Hu, Jianpeng Jiao, Jiashuo Liu, Dongyuan Mutu, Yanle Ren, Zhoufutu Wen, Kaiyuan Zhang, Xuanliang Zhang, Xiang Gao, Tianci He, et al. Finsearchcomp: Towards a realistic, expert-level evaluation of financial search and reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 46402–46428, 2026.

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

Nanbeige Lab, Chen Yang, Chengrui Huang, Fufeng Lan, Hanhui Chen, Hao Zhou, Huatong Song, Jiaqi Cao, Jiaying Zhu, Jinlin Niu, et al. Nanbeige4. 2-3b: Unlocking agentic capabilities in a compact model. *arXiv preprint arXiv:2607.22083*, 2026.

Ziyuan Liu, Hengqi Liu, Zichuan Wang, Yang Qin, Jiachen Liang, Xu Chu, Shaowei Chen, Yuantao Gu, and Mu Chuan. Iris: Climbing to the search frontier. *arXiv preprint arXiv:2609.04304*, 2026.

Krista Opsahl-Ong, Arnav Singhvi, Jasmine Collins, Ivan Zhou, Cindy Wang, Ashutosh Baheti, Owen Oertell, Jacob Portes, Sam Havens, Erich Elsen, et al. Officeqa pro: An enterprise benchmark for end-to-end grounded reasoning. *arXiv preprint arXiv:2603.08655*, 2026.

Sungrae Park, Sanghoon Kim, Gyoungjin Gim, Jungho Cho, Hyunwoong Ko, Minbyul Jeong, Minjeong Kim, Keunwoo Choi, Chaehun Shin, Chanwoong Yoon, Dongjun Kim, Eunwon Kim, Gyungin Shin, Hyeonju Lee, Hyungkyu Kang, Inseo Song, Jisu Bae, Jiyoon Han, Jiyun Lee, Joonkee Kim, Junyeop Lee, Mikyoung Cha, Sangwon Yu, Sehwan Joo, Seokyoon Kang, Seonghoon Yang, Seung Shin, Seunghyun Lee, Seungseop Lim, Seungyoun Shin, Sukyung Lee, Taegyeong Eo, Taehwan Oh, Taewhoo Lee, Wonho Song, Wonjun Oh, Wonseok Hwang, Yunsu Kim, Yura Shim, Hwalsuk Lee, Sunghun Kim, Du-Seong Chang, Kyunghyun Cho, Seungju Han, Yejin Choi, Junsuk Choe, Hwaran Lee, Minjeong Ban, Yun Taewon, Hwanjun Song, Jae-Gil Lee, KyungTae Lim, and Alice Oh. Solar open 2 technical report, 2026. URL https://arxiv.org/abs/2607.20062.

Zihan Qiu, Zekun Wang, Xiao Li, Yanpeng Li, Yang Xu, Yixuan Wang, Huaqing Zhang, Rui Men, Bochao Mao, Chengruidong Zhang, et al. On the design of qwen3. 8-next architecture: Evaluation, efficiency, and training stability. *arXiv preprint arXiv:2608.30320*, 2026.

Bytedance Seed. Seed2. 0 model card: Towards intelligence frontier for real-world complexity. *arXiv preprint arXiv:2607.00248*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zirui Tang, Xuanhe Zhou, Yumou Liu, Linchun Li, Yukai Wu, Weizheng Wang, Hongzhang Huang, Wei Zhou, Jun Zhou, Jiachen Song, et al. Workspace-bench 1.0: Benchmarking ai agents on workspace tasks with large-scale file dependencies. *arXiv preprint arXiv:2605.03596*, 2026.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Pu Wu, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, et al. Webshaper: Agentically data synthesizing via information-seeking formalization. In *International Conference on Learning Representations*, volume 2026, pp. 101872–101889, 2026.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

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.

Bertie Vidgen, Austin Mann, Abby Fennelly, John Wright Stanly, Lucas Rothman, Marco Burstein, Julien Benchek, David Ostrofsky, Anirudh Ravichandran, Debnil Sur, et al. Apex-agents. *arXiv preprint arXiv:2601.14242*, 2026.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of computer science*, 18(6):186345, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 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)*, pp. 3277–3287, 2021.