AGENTCO-OP: Retrieval-Based Synthesis of Interoperable Multi-Agent Workflows

Shuaike Shen[1,*], **Wenduo Cheng**[1,*], **Shike Wang**[1],
Mingqian Ma[2], **Jian Ma**[1†]
[1]Ray and Stephanie Lane Computational Biology Department,
School of Computer Science, Carnegie Mellon University
[2]Machine Learning Department,
School of Computer Science, Carnegie Mellon University
[*]Equal contribution, [†]Correspondence: jianma@cs.cmu.edu

Abstract

Designing multi-agent workflows is especially difficult in open-ended scientific settings where tasks lack curated training sets, reliable scalar evaluation metrics, and standardized interfaces between existing tools and agents. We propose AGENTCO-OP, a retrieval-based synthesis framework that composes reusable skills, tools, and external agents into executable workflows through typed artifact handoffs, then applies bounded self-guided local repair to implicated components when execution evidence indicates failure. In two open-world genomics case studies, AGENTCO-OP composes independently developed scientific agents and external tool repositories into auditable workflows without redesigning them or running global topology search. It coordinates specialized agents for spatial transcriptomics and gene-set interpretation to enable collaborative discovery from spatial transcriptomics data, and builds a parallel workflow for cross-modality marker analysis on single-cell multiome data. AGENTCO-OP can also import a searched workflow as a structural prior and improve it by grounding nodes with retrieved components and applying local repair, showing that synthesis and search are complementary. On six coding, math, and question-answering benchmarks, AGENTCO-OP achieves the best result on four benchmarks and the best average score under a unified backbone setting, while consistently reducing per-task cost relative to multi-agent baselines. Together, these results suggest that retrieval-based synthesis can extend automated agentic workflow design beyond benchmark-optimized agent graphs to open-world workflows built from existing agents, tools, and typed artifacts.

1 Introduction

Multi-agent LLM systems decompose complex tasks across specialized roles, tools, and prompts, and have shown strong results on reasoning, coding, question answering (QA), and scientific analysis [Tran et al., 2025, Hong et al., 2023, Wu et al., 2023]. As these systems mature, the bottleneck has shifted from constructing individual agents to designing interoperable workflows among them. Recent automated methods such as ADAS [Hu et al., 2024], AFlow [Zhang et al., 2024], and AgentSquare [Shang et al., 2024] frame this design problem as searching over candidate topologies, prompts, operators, or workflow programs, optimizing against a training set with a scalar evaluation function. This formulation is powerful when representative tasks and reliable scalar signals are available, and has shown strong results on standard QA, math, and coding benchmarks.

However, this search-based formulation becomes limiting for a broad class of real-world tasks. In scientific domains, problems are often open-ended and rarely come with curated training sets,

Preprint.

standardized test cases, or automatic evaluation functions that reflect scientific utility. In genomics, for example, marker-gene interpretation through pathway or gene-set enrichment has no single ground-truth answer; the same gene list can support multiple plausible interpretations depending on tissue, cell type, disease context, database choice, and statistical threshold [Subramanian et al., 2005, Wang et al., 2025c]. Such tasks are judged through heterogeneous intermediate evidence such as statistical significance, biological plausibility, consistency with known markers, and provenance of the analysis, which are hard to compress into a single reward, making repeated scoring of candidate workflows expensive and often impractical.

A second challenge concerns interoperability rather than optimization. Many scientific domains already have tool-augmented agents that experts have built and validated for specialized tasks, so a substantial part of the challenge is coordinating independently developed systems rather than creating new capabilities from scratch [Wei et al., 2025]. Such agents typically rely on incompatible environments, expose different interfaces, and maintain separate provenance states, so simply placing multiple agents together does not yield a coherent workflow. What is needed is a mechanism for retrieving relevant components, aligning their interfaces, passing typed artifacts between them, and repairing failed components using execution evidence.

We propose AGENTCO-OP, a framework that reframes automated multi-agent workflow design as retrieval-based synthesis. Given a task specification, AGENTCO-OP retrieves relevant resources, skills, tools, and external agents from curated libraries or user-provided repositories, assigns them to specialized roles, aligns their input-output interfaces through typed artifacts, and synthesizes them into an executable workflow as a directed graph. During execution, AGENTCO-OP monitors heterogeneous evidence such as execution traces, validation checks, tool errors, and cost signals, and triggers bounded evidence-guided local repair on implicated components rather than restarting synthesis. This synthesis-first view produces workflows where scalar metrics are unavailable, reuses prior engineering effort by composing existing skills, tools, and entire repositories of independently developed agents, and confines local repair to failing components rather than repeating global search.

We evaluate AGENTCO-OP in both open-world scientific settings and standard benchmarks. The open-world setting motivates the synthesis-first design, since benchmark-driven search is often impractical when no curated benchmark or evaluation function is available. We study three representative applications. First, AGENTCO-OP coordinates independently developed domain agents through a serial repository handoff, composing TissueAgent and GeneAgent for differential expression and gene-set interpretation on a developing human heart MERFISH dataset. Second, AGENTCO-OP composes complementary domain workflows in parallel, integrating Seurat and Signac into a cross-modality marker-discovery pipeline on PBMC multiome data. Third, AGENTCO-OP reuses existing agent graphs by importing prior workflows, grounding their nodes with retrieved skills and tools, and applying bounded local repair during execution. On six standard QA, mathematical reasoning, and code generation benchmarks, AGENTCO-OP further achieves the best performance on four of the six benchmarks under a matched backbone setting and the lowest average cost.

Our contributions are as follows.

1. We formulate automated multi-agent workflow design as retrieval-based synthesis for settings where scalar rewards are weak or unavailable, and instantiate this view in AGENTCO-OP, a framework that dynamically composes resources, skills, tools, and external agents into executable workflows through typed artifact handoffs and bounded evidence-guided local repair.

2. We demonstrate that AGENTCO-OP can coordinate independently developed scientific agents and tool repositories in open-world genomics tasks. Given only a task specification and GitHub links to relevant repositories, AGENTCO-OP automatically synthesizes interoperable multi-agent workflows that support collaboration among heterogeneous methods.

3. We further show that synthesis and search are complementary by importing an AFlow-searched agentic workflow on MBPP and improving it through retrieval grounding and evidence-guided local repair.

4. On six coding, math, and QA benchmarks, AGENTCO-OP is competitive with search-based agentic workflow design methods, achieving the best performance on four of six benchmarks while consistently lowering test-time token cost.

2 Related Work

2.1 Multi-agent systems

Multi-agent LLM systems decompose tasks across agents with different roles, tools, and communication patterns. Role-based collaboration assigns agents complementary responsibilities, as in CAMEL [Li et al., 2023], MetaGPT [Hong et al., 2023], AutoGen [Wu et al., 2023], and AgentVerse [Chen et al., 2023]. Deliberation-based systems improve reasoning by having multiple agents propose, debate, or reconcile answers, as in LLM-Debate [Du et al., 2024] and ReConcile [Chen et al., 2024]. Practical guides further codify manager-style coordination, handoffs, guardrails, and subagents [OpenAI, 2025b,a, Anthropic, 2025b]. These works treat agents as composable building blocks, but their workflow structures are still largely manually designed or template-based, which limits their generalization to new tasks.

2.2 Automatic agentic workflow design

A growing line of work automates the design of agentic workflows. Early systems such as DyLAN optimize team participation and communication through dynamic selection [Liu et al., 2023], and GPTSwarm formulates agent collaboration as an optimizable graph [Zhuge et al., 2024]. More recent methods broaden the search space: ADAS searches over code-defined agents [Hu et al., 2024], AFlow uses Monte Carlo Tree Search over executable workflow graphs from execution feedback [Zhang et al., 2024], AgentSquare defines a modular space over planning, reasoning, tool use, and memory [Shang et al., 2024], and MaAS introduces an agentic supernet that samples query-dependent architectures [Zhang et al., 2025]. Related efforts further explore automatic workflow generation and evolution, including Flow [Niu et al., 2025], EvoAgentX [Wang et al., 2025b], SEW [Zhao et al., 2025], and AutoFlow [Li et al., 2024]. These methods typically rely on repeated proposal, execution, and evaluation under representative tasks and scalar feedback. AGENTCO-OP targets a complementary setting where such feedback is weak, costly, or inaccessible, compiling a coordinated workflow directly from available skills, prior agents, and task requirements, while limiting runtime adaptation to bounded evidence-guided local repair.

2.3 Agent skills and tool use

A complementary line equips agents with externally specified capabilities. The Model Context Protocol standardizes tool, resource, and prompt access across providers [Anthropic, 2024]. Building on this, the Anthropic Agent Skills package procedural knowledge as portable folders loaded on demand via progressive disclosure [Anthropic, 2025a], with a recent survey systematizing the paradigm [Bhardwaj, 2026]. SkillFoundry mines heterogeneous resources into self-evolving skill libraries with executable contracts [Shen et al., 2026], and EvoSkills evolves multi-file skill packages through co-evolutionary verification [Zhang et al., 2026]. Earlier tool-use work covers learned invocation and large API retrieval [Schick et al., 2023, Qin et al., 2024, Patil et al., 2024]. These works expose capabilities to agents but do not determine how they should be organized into task-specific workflows. AGENTCO-OP builds on this direction by treating skills as typed, testable units whose contracts are enforced during workflow synthesis and typed artifact handoff.

2.4 Scientific agents

LLM agents are increasingly applied to scientific discovery. SpatialAgent addresses spatial-biology pipelines from panel design to hypothesis generation [Wang et al., 2025a], and GeneAgent reduces hallucinations in gene-set analysis through database-grounded self-verification [Wang et al., 2025c]. The Virtual Lab orchestrates a principal investigator and specialist agents to design experimentally validated SARS-CoV-2 nanobodies [Swanson et al., 2025]. Biomni provides a generalist biomedical action space [Huang et al., 2025], and STELLA self-evolves its template library and tool ocean [Jin et al., 2025]. Other systems target gene editing, perturbation design, and chemistry, including CRISPR-GPT, BioDiscoveryAgent, and ChemCrow [Huang et al., 2024, Roohani et al., 2025, Bran et al., 2024]. These agents offer powerful specialized capabilities, but are typically built as standalone systems for specific task families. Composing them into multi-step, cross-modal, or interdisciplinary workflows remains difficult because their interfaces, environments, outputs, and assumptions are not aligned. AGENTCO-OP addresses this composition problem by wrapping specialized agents and domain workflows as executable graph nodes, aligning them through typed artifacts, and synthesizing coherent collaborative workflows.

3 Method

3.1 Problem formulation

We study the problem of automatically constructing multi-agent workflows for complex tasks. Given a task specification x, the goal is to produce an executable workflow W that decomposes the task, grounds each role in retrieved components, and coordinates communication through typed artifacts. We represent a task specification as

$$x = (g, c, r, \Omega), \tag{1}$$

where g is the user goal, c is the task context, r specifies operational constraints such as available data, budget, runtime, environment requirements, and desired output format, and Ω denotes task-specific resources provided or required by the user, including documents, datasets, repositories, tools, external agents, and existing agent graphs.

Traditional automated workflow design formulates the problem as search over a workflow space:

$$W^* = \arg\max_{W \in \mathcal{W}} \text{Eval}(W; D), \tag{2}$$

where $\mathcal{W}$ is the candidate space, D is a benchmark or training set, and Eval is a scalar evaluation function. This formulation is effective when representative tasks and reliable scalar metrics exist, but many real-world and scientific tasks have no curated benchmark, no ground-truth output, and no single scalar reward that captures workflow quality. Success instead depends on heterogeneous evidence such as the validity of intermediate artifacts, correctness of tool use, scientific plausibility, and ability to recover from failures.

We therefore formulate automated workflow design as a *retrieval-based synthesis* problem, in which a workflow is composed from retrieved resources, skills, tools, and external agents. Let S denote a global library of reusable artifacts, partitioned into reference resources such as papers and documentation, procedural agent skills, callable tools, and wrapped external agent repositories. Each artifact carries a description and an I/O interface or typed artifact schema so that retrieved items can be composed.

Then AGENTCO-OP synthesizes a workflow based on the retrieved artifacts:

$$W = \textsc{Synthesize}(x, \mathcal{S}) \triangleq (R, G, \phi, \Pi), \tag{3}$$

where R is a set of agent roles, G is a dependency graph over roles, $\phi : R \to 2^{\mathcal{S}}$ attaches a set of artifacts to each role, and Π specifies the interface protocol for communication between agents.

3.2 Method Details

Graph representation and workflow synthesis. Following prior work such as GPTSwarm [Zhuge et al., 2024] and Flow [Niu et al., 2025], we represent a multi-agent workflow as a directed graph $G = (V, E)$ in which each node is an agent with an assigned role belongs to R, and each edge represents the direction in which information and intermediate artifacts flow between agents. The interface protocol is governed by Π in Eq. 3. AGENTCO-OP extends the graph representation in two ways. First, a node can be an external agent or an end-to-end method wrapped in a Docker container, so the graph can incorporate heterogeneous components without imposing a uniform native execution environment. Second, every agent node is equipped with a set of skills and tools matched to its role, and the mapping relationship is defined by ϕ in Eq. 3. Thus, a node carries not only an instruction but also the procedural knowledge and callable operations needed to execute it.

To synthesize the workflow, AGENTCO-OP analyzes the input task specification and formulates a retrieval plan. AGENTCO-OP then retrieves a set of task-relevant artifacts, including related materials that inform the choice of workflow topology, agent skills that encode procedural knowledge, tools that expose callable operations, and metadata and documentation from the external GitHub repository if a GitHub repository URL is provided. AGENTCO-OP analyzes these artifacts to synthesize an initial multi-agent workflow as a directed graph.

Evidence-guided local repair. During workflow execution, AGENTCO-OP continuously monitors execution evidence such as logs, intermediate outputs, validation signals, tool errors, and cost signals. A Reviewer triggers local repair when this evidence indicates failure or uncertainty. Local repair consults a small set of repair policies and revises only the implicated nodes, attached skills and tools, or communication edges, so AGENTCO-OP produces a patched graph $G' = (V', E')$ rather than

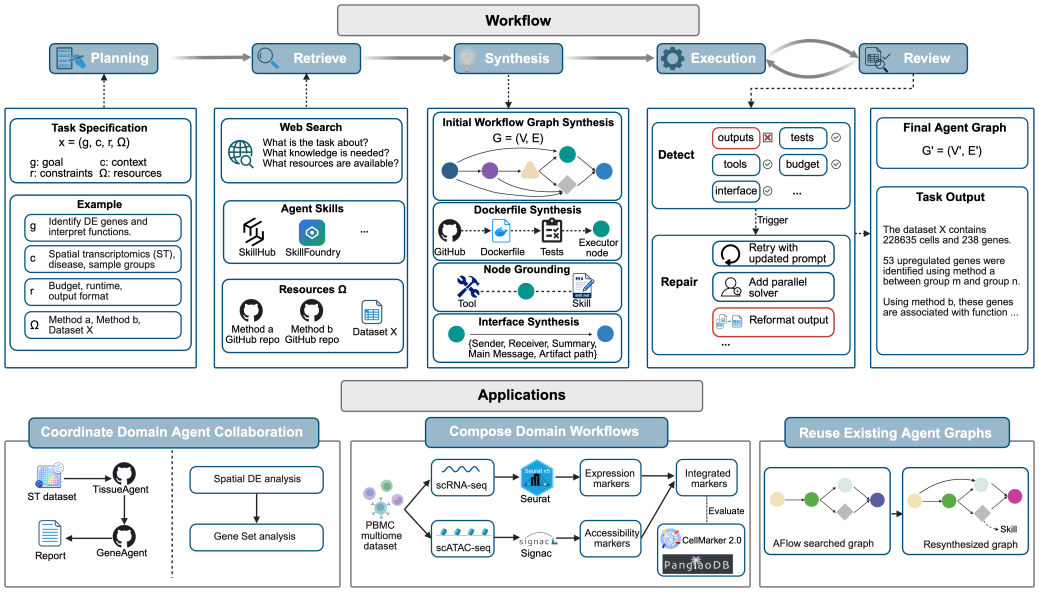

Figure 1: Overview of AGENTCO-OP. AGENTCO-OP synthesizes multi-agent workflows through five main stages: Planning, Retrieval, Synthesis, Execution, and Review. Given a typed task specification $x = (g, c, r, \Omega)$, the system retrieves relevant knowledge, skills, tools, repositories, and datasets, then synthesizes an executable workflow graph $G = (V, E)$. The synthesis stage includes initial graph construction, Dockerfile or executor wrapping, node grounding with skills and tools, and interface alignment through standardized message and artifact schemas. During execution, the reviewer monitors signals such as outputs, tests, tool behavior, budget, and interfaces. When failures or uncertainty arise, AGENTCO-OP performs bounded local repair, producing a patched graph $G' = (V', E')$ and the final task output. AGENTCO-OP supports three representative applications: coordinating collaboration among domain-specific agents, composing domain workflows, and reusing existing agent graphs.

restarting the entire synthesis pipeline. Repair stops when validation succeeds, the repair budget is exhausted, or the maximum number of repair rounds is reached. This bounded, evidence-guided adaptation allows the workflow to recover from issues that emerge only at execution time and would be hard to anticipate during the initial synthesis.

Composition with domain workflows and search-based workflows. As shown in Fig. 1, AGENTCO-OP accepts a GitHub repository URL and wraps the external workflow into a Docker container. Following the approach of Repo2Run [Hu et al., 2025], AGENTCO-OP builds the Docker image, uses build feedback and available tests to revise the container specification, and synthesizes or updates the Dockerfile together with accompanying documentation. The Docker container is then plugged into the graph as an external agent node, with its inputs and outputs aligned through the typed interface protocol. The same wrapping procedure applies to end-to-end methods, which can be attached to agent nodes as tools. Together, these mechanisms let AGENTCO-OP reuse prior engineering, resolve environment dependency conflicts through Docker, and enable independently developed agents to collaborate on tasks that none of them could solve alone.

AGENTCO-OP can also use an agent graph produced by a search-based agentic workflow design method as an additional resource in Ω. Rather than executing this graph directly, AGENTCO-OP treats it as a reference structure that guides workflow synthesis. The framework then follows the same synthesis process illustrated in Fig. 1, grounding graph nodes with retrieved skills and tools, enforcing typed interface and execution constraints, and applying bounded evidence-guided local repair during runtime.

4 Experiments

We evaluate AGENTCO-OP in two complementary regimes. First, we study three open-world workflow composition tasks that motivate the synthesis-first design. The first task tests whether AGENTCO-OP can coordinate independently developed domain agents by composing TissueAgent and GeneAgent [Wang et al., 2025c] for differential expression and gene-set interpretation on a

developing human heart MERFISH dataset [Farah et al., 2024]. The second task examines parallel composition of complementary domain workflows, integrating Seurat [Butler et al., 2018] and Signac [Stuart et al., 2021] into a cross-modality marker-discovery pipeline on PBMC multiome data, with marker quality evaluated against CellMarker 2.0 [Hu et al., 2023] and PanglaoDB [Franzén et al., 2019]. The third task evaluates whether AGENTCO-OP can reuse and improve a searched workflow by importing a multi-agent graph produced by AFlow [Zhang et al., 2024] on MBPP [Austin et al., 2021], then refining it through retrieval-based synthesis and bounded evidence-guided local repair.

We further evaluate AGENTCO-OP on six standard benchmarks spanning question answering (HotpotQA [Yang et al., 2018], DROP [Dua et al., 2019]), code generation (HumanEval [Chen et al., 2021], MBPP [Austin et al., 2021]), and mathematical reasoning (GSM8K [Cobbe et al., 2021], MATH [Hendrycks et al., 2021]). Following prior work [Zhang et al., 2024], we use GPT-4o-mini as the base model for our method and matched-backbone baseline runs. We compare AGENTCO-OP against three categories of baselines, including single-agent methods such as CoT [Wei et al., 2022], CoT SC [Wang et al., 2022], Self Refine [Madaan et al., 2023], and MedPrompt [Nori et al., 2023], search-based multi-agent design methods such as ADAS [Hu et al., 2024] and AFlow [Zhang et al., 2024], and predefined multi-agent collaboration methods such as MultiPersona [Wang et al., 2024], LLM-Debate [Du et al., 2024], and Reconcile [Chen et al., 2024]. Beyond task performance, we also analyze test-time and aggregate token cost of AGENTCO-OP against multi-agent collaboration baselines to demonstrate its efficiency.

4.1 Coordinate Domain Agent Collaboration

First, we evaluate whether AGENTCO-OP can coordinate independently developed domain agents to solve a collaborative scientific analysis task. The task asks whether aFibro cells in the AVN/AV ring cellular community exhibit a distinct transcriptional program compared with aFibro cells in the left atria and right atria communities in a spatial transcriptomics dataset [Farah et al., 2024]. Solving this task requires spatial transcriptomics analysis to process the dataset and identify differentially expressed marker genes, followed by gene-set interpretation to characterize the resulting transcriptional program. The task naturally requires collaboration between a spatial transcriptomics agent and a gene-set analysis agent, making it a direct test of whether AGENTCO-OP can synthesize a coherent workflow across independently developed scientific agents with different inputs, outputs, and execution environments.

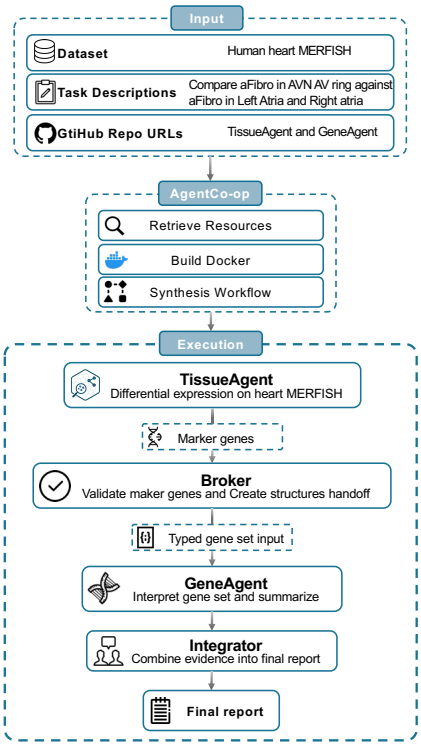

Figure 2: AGENTCO-OP orchestrates domain agents for collaborative biological analysis. Given a developing human heart MERFISH dataset and a task description, AGENTCO-OP prepares the domain-agent environment by profiling repositories and building containers, then coordinates a collaborative workflow for TissueAgent and GeneAgent.

We leverage AGENTCO-OP to compose TissueAgent [macompbio, 2025] and GeneAgent [Wang et al., 2025c], which are specialized agents for spatial transcriptomics analysis and gene-set analysis, respectively. Given only the task description and the GitHub repository URLs for the two agents, AGENTCO-OP profiles both repositories, builds isolated Docker containers, registers each container as an external workflow node, and synthesizes the collaborative workflow shown in Fig. 3 with an explicit broker-mediated typed handoff between marker discovery and gene-set interpretation.

The synthesized workflow executes in four steps. First, the upstream TissueAgent node loads the MERFISH object, identifies 576 target aFibro cells in the AVN/AV ring community and 5,685 control aFibro cells in the left and right atria communities, and performs differential expression analysis, yielding 53 upregulated markers. Second, the Broker node validates the marker table and converts it into a structured input while preserving all 53 genes, ensuring that the downstream GeneAgent node receives typed evidence rather than an unstructured free-text list. Third, GeneAgent interprets the 53 markers without re-running differential expression and annotates them as an AV canal- and

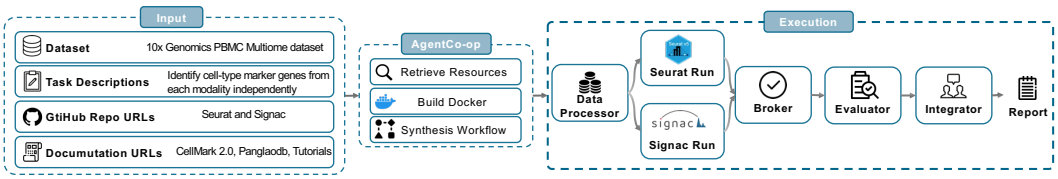

Figure 3: AGENTCO-OP coordinates external tools for cross-modal marker discovery. AGENTCO-OP registers external Seurat and Signac tool nodes, runs parallel RNA and ATAC marker-discovery branches, validates typed artifacts, evaluates marker support against CellMarker 2.0 and PanglaoDB, and integrates the evidence into a final report.

node-associated fibroblast program. Finally, the Integrator combines the differential expression evidence with the GeneAgent interpretation and concludes that AVN/AV ring aFibro cells represent a developmentally specialized, ECM-rich, conduction-niche-associated state rather than generic atrial stroma. This result demonstrates that AGENTCO-OP can coordinate independently developed scientific agents into a coherent collaborative workflow to solve an end-to-end scientific analysis task without requiring global workflow search or redesigning either repository.

4.2 Compose Domain Workflows

We then evaluate whether AGENTCO-OP can compose independently developed domain workflows to solve a collaborative cross-modality analysis task. The task asks whether integrating marker signals from paired RNA and chromatin accessibility modalities can improve cell-type marker identification on multiome data. Solving this task requires scRNA-seq analysis to identify expression-based markers and scATAC-seq analysis to identify accessibility-based markers, since the two modalities provide partially overlapping but non-identical evidence of cell identity. It naturally requires coordination between RNA and ATAC analysis workflows. As such, this task provides a direct test of whether AGENTCO-OP can synthesize a coherent cross-modality workflow from modality-specialized domain repositories and integrate complementary evidence across modalities.

We leverage AGENTCO-OP to compose Seurat [Butler et al., 2018] and Signac [Stuart et al., 2021], which are widely used workflows for single-cell RNA-seq and ATAC-seq analysis respectively. Given the task description, the PBMC multiome dataset from 10x Genomics, and the GitHub repositories and tutorials of the two workflows, AGENTCO-OP builds two separate Docker containers from the Seurat and Signac GitHub repositories, registers them as execution nodes, and synthesizes a parallel workflow followed by a join step as illustrated in Fig. 3. The Seurat node runs the FindAllMarkers function on the gene expression assay, and the Signac node runs the GeneActivity function followed by the FindAllMarkers function on the chromatin accessibility assay. The evaluator collects identified marker sets from both nodes, computes their intersection and union per cell type, and evaluates them against CellMarker 2.0 and PanglaoDB [Hu et al., 2023, Franzén et al., 2019], two established cell-type marker databases. The intersection captures jointly supported markers and is evaluated for precision, while the union captures all recovered markers and is evaluated for recall.

The results are shown in Tab. 1. Across both reference databases, combining the two modalities improves both macro precision and recall over either single modality, and this trend holds at the per-cell-type level for the majority of cell types. This shows that AGENTCO-OP can compose existing domain workflows to integrate complementary evidence. More details are illustrated in App. A.2.3.

Table 1: Macro precision and recall of cross-modality marker integration on the PBMC multiome dataset, evaluated against two marker gene databases. Precision is computed on the intersection and recall on the union of the two modalities. RNA and ATAC report each modality alone. Combined reports the cross-modality result. Bold marks the best result.

Database	Metric	RNA	ATAC	Combined
CellMarker 2.0	Precision	0.195	0.110	**0.303**
	Recall	0.102	0.061	**0.124**
PanglaoDB	Precision	0.231	0.131	**0.333**
	Recall	0.097	0.054	**0.117**

4.3 Reuse Existing Agent Graphs

We further show that AGENTCO-OP can import an existing predefined agent graph as a reference to synthesis the workflow. We take the multi-agent graph produced by a trained AFlow search and feed it into AGENTCO-OP as additional resources in Ω, then AGENTCO-OP can retrieve the artifacts, resynthesize the agent graph and interface protocol, attach retrieved skills and tools to the agent nodes and apply bouneded local repair during execution. We evaluate the resulting workflow on the MBPP benchmark [Austin et al., 2021]. The results are

reported in Tab. 2. AFlow + AGENTCO-OP outperforms both AFlow alone and AGENTCO-OP built from scratch, which confirms that synthesis and search are complementary. The imported graph contributes a strong reference resource, while AGENTCO-OP contributes resource grounding and runtime adaptability that pure search does not provide.

4.4 Benchmarks

The benchmark results are shown in Tab. 3. For GSM8K and MATH, we report the Solve Rate (%)as the primary metric. For HumanEval and MBPP, we report the pass@1 metric to assess code accuracy. For HotpotQA and DROP, we report the F1 Score. Without any training or workflow-search stage, AGENTCO-OP achieves the best performance on four of the six benchmarks and ranks first on the average score under the matched backbone setting. This competitiveness is consistent with AGENTCO-OP's synthesis-first design: It starts from reusable workflow priors, including skills, roles, motifs, typed handoffs, and repair policies, which already encode many of the structures search methods have to discover through expensive trial and error. Search-based methods typically optimize one persistent workflow for a task distribution, whereas different benchmark instances often require different checks, decompositions, or repair actions. Instead of searching for one workflow that works best on average, AGENTCO-OP adapts the workflow locally for each problem when execution evidence indicates failure or uncertainty.

Table 2: MBPP performance of different agentic workflow design strategies. Bold indicates the best pass@1 score. Initializing AGENTCO-OP from AFlow searched graph improves performance compared with initializing it from scratch.

Strategy	pass@1
AFlow	78.2
AGENTCO-OP (From Scratch)	87.1
AFlow + AGENTCO-OP	**87.5**

Table 3: Performance comparison of different methods on six benchmarks spanning QA, code, and math, using GPT-4o-mini as the backbone model. Bold indicates the best result. The original AFlow paper mixes multiple backbone models, and AFlow* denotes the results reported by its authors. For a fair comparison, we rerun AFlow using GPT-4o-mini only, reported as AFlow (GPT-4o-mini).

Method	Benchmarks						Avg.
	HotpotQA	DROP	HumanEval	MBPP	GSM8K	MATH	
IO (GPT-4o-mini) [Hurst et al., 2024]	68.1	68.3	87.0	71.8	92.7	48.6	72.8
CoT Wei et al. [2022]	67.9	78.5	88.6	71.8	92.4	48.8	74.7
CoT SC (5-shot) Wang et al. [2022]	68.9	78.8	91.6	73.6	92.7	50.4	76.0
MedPrompt Nori et al. [2023]	68.3	78.0	91.6	73.6	90.0	50.0	75.3
MultiPersona Wang et al. [2024]	69.2	74.4	89.3	73.6	92.8	50.8	75.0
Self Refine Madaan et al. [2023]	60.8	70.2	87.8	69.8	89.6	46.1	70.7
ADAS Hu et al. [2024]	64.5	76.6	82.4	53.4	90.8	35.4	67.2
AFlow* Zhang et al. [2024]	73.5	80.6	**94.7**	83.4	93.5	56.2	80.3
LLM-Debate Du et al. [2024]	71.8	81.4	91.4	70.7	92.4	50.0	76.3
ReConcile Chen et al. [2024]	73.8	**82.1**	89.3	70.3	93.7	44.1	75.6
AFlow (GPT-4o-mini) Zhang et al. [2024]	71.4	68.9	89.3	78.2	86.8	53.1	74.3
AGENTCO-OP (GPT-4o-mini)	**76.5**	77.2	90.2	**87.1**	**94.4**	**58.2**	**80.6**

4.5 Cost Analysis

We further record the token cost of each method on every benchmark, with results shown in Tab. 4. AGENTCO-OP is substantially more efficient than the multi-agent baselines, and its test-time cost is lower than ReConcile on all six benchmarks and lower than LLM Debate on five of six benchmarks. Search-based methods such as AFlow consume additional tokens and time exploring, evaluating, and optimizing candidate workflows on training set before producing a final design. Discussion-based methods such as LLM-Debate and ReConcile incur repeated rounds of inter-agent communication for every task instance, which compounds quickly as the number of tasks grows. AGENTCO-OP avoids both of these patterns. AGENTCO-OP separates workflow synthesis from workflow repair. Synthesis produces a reusable initial workflow, while repair performs bounded, instance-specific local repair during execution time. AGENTCO-OP modifies the workflow graph only for the current instance and the modifications will be discarded afterward. This design allows AGENTCO-OP to adapt to heterogeneous problem instances without overfitting the global workflow or requiring expensive resynthesis.

Table 4: Per-dataset performance and cost comparison across methods. Costs are aggregated over the entire benchmark dataset. A dash in the Train Cost column indicates that the method requires no workflow search or training stage.

Dataset	Method	Score	Train Cost	Test Cost	Total Cost
HotpotQA	LLM Debate	71.8	–	$1.5200	$1.5200
	ReConcile	73.8	–	$3.7600	$3.7600
	AFlow	20.0	$4.6104	$1.3398	$5.9502
	AGENTCO-OP	**76.5**	–	$0.4284	$0.4284
DROP	LLM Debate	81.4	–	$0.7200	$0.7200
	ReConcile	**82.1**	–	$1.6800	$1.6800
	AFlow	68.9	$1.6798	$0.3235	$2.0033
	AGENTCO-OP	77.2	–	$0.3853	$0.3853
HumanEval	LLM Debate	**91.4**	–	$0.1572	$0.1572
	ReConcile	89.3	–	$0.4061	$0.4061
	AFlow	89.3	$0.2258	$0.0371	$0.2629
	AGENTCO-OP	90.2	–	$0.1062	$0.1062
MBPP	LLM Debate	70.7	–	$0.1705	$0.1705
	ReConcile	70.3	–	$0.7502	$0.7502
	AFlow	72.4	$0.3475	$0.1152	$0.4627
	AGENTCO-OP	**87.1**	–	$0.1791	$0.1791
GSM8K	LLM Debate	92.4	–	$1.6880	$1.6880
	ReConcile	93.7	–	$1.8990	$1.8990
	AFlow	86.8	$0.0469	$0.2000	$0.2469
	AGENTCO-OP	**94.4**	–	$0.2537	$0.2537
MATH	LLM Debate	50.0	–	$1.7982	$1.7982
	ReConcile	44.1	–	$1.6038	$1.6038
	AFlow	53.1	$0.0781	$0.2691	$0.3472
	AGENTCO-OP	**58.2**	–	$0.3670	$0.3670

5 Limitations and Future Work

We acknowledge several limitations of this study. First, our current evaluation of domain-agent collaboration remains within a single scientific domain and two genomics-centered case studies. Future work could extend this setting to interdisciplinary synthesis and cross-domain agent collaboration. Second, the framework still depends on the quality of the available domain resources, including specialized agents, skills, and tools. Coordination failures may occur when agents produce poorly specified outputs, expose incompatible interfaces, or generate intermediate artifacts that are difficult to validate. Third, bounded local repair improves robustness but does not guarantee global optimality; a locally repaired workflow may still miss a better global organization. Fourth, the biological case studies demonstrate auditable workflow composition, but their scientific conclusions should be interpreted as computational analyses that require expert review and, where appropriate, orthogonal validation. In future work, we plan to extend AGENTCO-OP toward more adaptive organization discovery, stronger verification of intermediate outputs, richer memory and provenance tracking, more explicit typed artifact schemas, and broader integration with domain-specific agent libraries.

6 Conclusion

In summary, AGENTCO-OP introduces a retrieval-based synthesis paradigm for automatic multi-agent workflow design. Rather than relying on benchmark-driven global search, AGENTCO-OP dynamically composes skills, tools, roles, and external agents into task-specific workflows, coordinates them through typed artifact handoffs, and refines implicated components through bounded evidence-guided local repair. Our results show that workflow synthesis can remain competitive with search-based design methods on standard benchmarks while reducing test-time token cost relative to discussion-based multi-agent baselines. More importantly, this paradigm is better aligned with real-world scientific domains, where benchmarks are often inaccessible, prior workflows already exist, and interoperability among heterogeneous agents is essential. In two biological analysis case studies, AGENTCO-OP demonstrates how independently developed agents and domain methods can be coordinated to answer biologically meaningful questions that require multiple complementary forms of expertise without redesigning the underlying repositories or performing global topology search.

More broadly, this work suggests that progress in scientific agents will depend not only on building stronger specialized agents, but also on developing methods for organizing, composing, and adapting heterogeneous agents into reliable collaborative systems. Scientific workflows are rarely solved by a single capability; they require coordinated expertise across multiple tasks and domains. By

treating workflow construction as an organization synthesis problem, AGENTCO-OP provides a practical path toward a reusable, cooperative, and extensible ecosystem of specialized scientific agents, where existing expertise can be composed, verified, and reused to address increasingly complex cross-domain discovery challenges.

References

Anthropic. Introducing the model context protocol. `https://www.anthropic.com/news/model-context-protocol`, 2024.

Anthropic. Agent skills. `https://docs.anthropic.com/en/docs/claude-code/skills`, 2025a.

Anthropic. Create custom subagents. `https://code.claude.com/docs/en/sub-agents`, 2025b.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Varun Pratap Bhardwaj. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*, 2026.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. ChemCrow: Augmenting large-language models with chemistry tools. *Nature Machine Intelligence*, 6:525–535, 2024.

Andrew Butler, Paul Hoffman, Peter Smibert, Efthymia Papalexi, and Rahul Satija. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature biotechnology*, 36(5):411–420, 2018.

Justin Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7066–7085, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first international conference on machine learning*, 2024.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Elie N Farah, Robert K Hu, Colin Kern, Qingquan Zhang, Ting-Yu Lu, Qixuan Ma, Shaina Tran, Bo Zhang, Daniel Carlin, Alexander Monell, et al. Spatially organized cellular communities form the developing human heart. *Nature*, 627(8005):854–864, 2024.

Oscar Franzén, Li-Ming Gan, and Johan L. M. Björkegren. PanglaoDB: a web server for exploration of mouse and human single-cell rna sequencing data. *Database*, 2019:baz046, 2019. doi: 10.1093/database/baz046.

Yuhan Hao, Stephanie Hao, Erica Andersen-Nissen, William M. Mauck, Shiwei Zheng, Andrew Butler, Maddie J. Lee, Aaron J. Wilk, Charlotte Darby, Michael Zager, Paul Hoffman, Marlon Stoeckius, Efthymia Papalexi, Eleni P. Mimitou, Jaison Jain, Avi Srivastava, Tim Stuart, Lamar B. Fleming, Bertrand Yeung, Angela J. Rogers, M. Juliana McElrath, Catherine A. Blish, Raphael Gottardo, Peter Smibert, and Rahul Satija. Integrated analysis of multimodal single-cell data. *Cell*, 184(13):3573–3587.e29, 2021. doi: 10.1016/j.cell.2021.04.048.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Advances in Neural Information Processing Systems*, 2021.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. Metagpt: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

Congxue Hu et al. CellMarker 2.0: an updated database of manually curated cell markers in human and mouse and web tools based on scrna-seq data. *Nucleic Acids Research*, 51(D1):D870–D876, 2023. doi: 10.1093/nar/gkac947.

Ruida Hu, Chao Peng, Xinchen Wang, Junjielong Xu, and Cuiyun Gao. Repo2run: Automated building executable environment for code repository at scale. *arXiv preprint arXiv:2502.13681*, 2025.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*, 2024.

Kaixuan Huang, Yuanhao Qu, Henry Cousins, William A. Johnson, Di Yin, Mihir Shah, Denny Zhou, Russ Altman, Mengdi Wang, and Le Cong. CRISPR-GPT: An LLM agent for automated design of gene-editing experiments. *bioRxiv 2024.04.25.591003*, 2024.

Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, Yusuf Roohani, Ryan Li, Lin Qiu, Gavin Li, Junze Zhang, Di Yin, Shruti Marwaha, Jennefer N. Carter, Xin Zhou, Matthew Wheeler, Jonathan A. Bernstein, Mengdi Wang, Peng He, Jingtian Zhou, Michael Snyder, Le Cong, Aviv Regev, and Jure Leskovec. Biomni: A general-purpose biomedical AI agent. *bioRxiv 2025.05.30.656746*, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Ruofan Jin, Zaixi Zhang, Mengdi Tang, Le Cong Wang, and Mengdi Wang. STELLA: Self-evolving LLM agent for biomedical research. *arXiv preprint arXiv:2507.02004*, 2025.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, 2023.

Zelong Li, Shuyuan Xu, Kai Mei, Wenyue Hua, Balaji Rama, Om Raheja, Hao Wang, He Zhu, and Yongfeng Zhang. AutoFlow: Automated workflow generation for large language model agents. *arXiv preprint arXiv:2407.12821*, 2024.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. *arXiv preprint arXiv:2310.02170*, 2023.

ma-compbio. TissueAgent: A role-based multi-agent framework for reproducible spatial transcriptomics workflows. `https://github.com/ma-compbio/TissueAgent`, 2025. Accessed: 2026-05-06.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Boye Niu, Yiliao Song, Kai Lian, Yifan Shen, Yu Yao, Kun Zhang, and Tongliang Liu. Flow: Modularized agentic workflow automation. In *International Conference on Learning Representations (ICLR)*, 2025.

Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolo Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, et al. Can generalist foundation models outcompete special-purpose tuning? case study in medicine. *arXiv preprint arXiv:2311.16452*, 2023.

OpenAI. Openai agents sdk documentation. `https://developers.openai.com/api/docs/guides/agents`, 2025a.

OpenAI. A practical guide to building agents. `https://openai.com/business/guides-and-resources/a-practical-guide-to-building-ai-agents/`, 2025b.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations (ICLR)*, 2024.

Yusuf Roohani, Andrew Lee, Qian Huang, Jian Vora, Zachary Steinhart, Kexin Huang, Alexander Marson, Percy Liang, and Jure Leskovec. BioDiscoveryAgent: An AI agent for designing genetic perturbation experiments. In *International Conference on Learning Representations (ICLR)*, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. Agentsquare: Automatic llm agent search in modular design space. *arXiv preprint arXiv:2410.06153*, 2024.

Shuaike Shen, Wenduo Cheng, Mingqian Ma, Alistair Turcan, Martin Jinye Zhang, and Jian Ma. Skillfoundry: Building self-evolving agent skill libraries from heterogeneous scientific resources. *arXiv preprint arXiv:2604.03964*, 2026.

Tim Stuart, Avi Srivastava, Shaista Madad, Caleb A. Lareau, and Rahul Satija. Single-cell chromatin state analysis with signac. *Nature Methods*, 18(11):1333–1341, 2021. doi: 10.1038/s41592-021-01282-5.

Aravind Subramanian, Pablo Tamayo, Vamsi K Mootha, Sayan Mukherjee, Benjamin L Ebert, Michael A Gillette, Amanda Paulovich, Scott L Pomeroy, Todd R Golub, Eric S Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the national academy of sciences*, 102(43):15545–15550, 2005.

Kyle Swanson, Wesley Wu, Nash L. Bulaong, John E. Pak, and James Zou. The Virtual Lab of AI agents designs new SARS-CoV-2 nanobodies. *Nature*, 646:716–723, 2025. doi: 10.1038/s41586-025-09442-9.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*, 2025.

H. Wang et al. Spatialagent: An autonomous ai agent for spatial biology. *bioRxiv*, 2025a. doi: 10.1101/2025.04.03.646459.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Yingxu Wang, Yuxuan Liu, Lu Tian, Wenkang Shen, Zixuan Tang, Tianqi Wang, Wenhao Wu, Wenjun Liu, and Quanjia Yu. EvoAgentX: An automated framework for evolving agentic workflows. *arXiv preprint arXiv:2507.03616*, 2025b.

Z. Wang, Q. Jin, C.-H. Wei, et al. Geneagent: Self-verification language agent for gene-set analysis using domain databases. *Nature Methods*, 2025c. doi: 10.1038/s41592-025-02748-6.

Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 257–279, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Jiaqi Wei, Yuejin Yang, Xiang Zhang, Yuhan Chen, Xiang Zhuang, Zhangyang Gao, Dongzhan Zhou, Guangshuai Wang, Zhiqiang Gao, Juntai Cao, et al. From ai for science to agentic science: A survey on autonomous scientific discovery. *arXiv preprint arXiv:2508.14111*, 2025.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*, 2025.

Hanrong Zhang, Shicheng Fan, Henry Peng Zou, Yankai Chen, Zhenting Wang, Jiayu Zhou, Chengze Li, Wei-Chieh Huang, Yifei Yao, Kening Zheng, et al. Evoskills: Self-evolving agent skills via co-evolutionary verification. *arXiv preprint arXiv:2604.01687*, 2026.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*, 2024.

Siwei Zhao, Xinyu Liu, Yifei Zhao, Tianyu Yang, Jiaqi Wang, Yong Bai, and Yang Liu. SEW: Self-evolving agentic workflows for automated code generation. *arXiv preprint arXiv:2505.18646*, 2025.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Language agents as optimizable graphs. *arXiv preprint arXiv:2402.16823*, 2024.

A Appendix

A.1 Methods

A.1.1 Planning and Retrieval

Planning. The Planning stage analyzes the typed task specification $x = (g, c, r, \Omega)$ and formulates a retrieval plan that determines what knowledge is needed to solve the task. Concretely, AGENTCO-OP decomposes the user goal g into sub-goals consistent with the task context c, identifies the operational constraints in r that must be respected during synthesis, including runtime, budget, environment requirements, and desired output format, and inspects the resources in Ω to decide which entries should be wrapped as execution nodes and which can be referenced as documents or datasets. The retrieval plan therefore acts as a specification of what AGENTCO-OP should look up before constructing the graph, separating the decision of *what to retrieve* from the actual retrieval calls.

Retrieval. Guided by the retrieval plan, AGENTCO-OP gathers task-relevant artifacts from heterogeneous sources. Reference resources, including research papers and curated documentation, inform the choice of workflow topology by providing concrete examples of how similar problems have been decomposed. Agent skill libraries, such as SkillHub and SkillFoundry, supply procedural knowledge encoded as portable skill packages. Tool registries expose callable operations such as database queries, plotting routines, or domain-specific functions. For each external repository URL provided in Ω, AGENTCO-OP additionally retrieves the repository metadata and documentation, including README files, tutorials, and example scripts, which are later used both for synthesizing the Docker container and for grounding the corresponding executor node. The retrieved artifacts populate the global library $\mathcal{S}$ from which the workflow is later composed.

A.1.2 Synthesis

Initial graph construction. The Synthesis stage begins by producing an initial directed graph $G = (V, E)$ from the retrieved artifacts and task specification. Topology decisions are informed by retrieved reference workflows for related problems and by the structure of any imported agent graph in Ω. AGENTCO-OP also decides whether the graph should be linear, parallel, or a mixed topology based on data dependencies in the task, for example assigning two modality-specific analyses to parallel branches when their inputs are independent and joining them at a downstream evaluator.

Node grounding. After the topology is fixed, AGENTCO-OP grounds each node by attaching a set of skills and tools matched to its role, defined by the mapping $\phi : R \to 2^{\mathcal{S}}$ in Eq. 3. Matching is performed by scoring candidate skills and tools against the role description and the upstream and downstream artifact types of the node, and selecting the top-ranked entries. As a result, every node carries not only an instruction but also the procedural knowledge and callable operations needed to execute it, which both reduces prompt-engineering load and enforces consistency across instances of the same role.

Dockerfile synthesis. For each external repository or end-to-end method that needs to be wrapped as an execution node, AGENTCO-OP builds an isolated Docker container following Repo2Run [Hu et al., 2025]. The procedure is iterative: AGENTCO-OP drafts a Dockerfile from the retrieved repository metadata, attempts to build the image, and on failure inspects the build log to revise the dependency list, base image, or build commands. Available repository tests and example scripts are then executed inside the container as a smoke check; recurrent failures trigger a further revision round. The same wrapping procedure also applies to end-to-end methods that are not full agentic workflows. Such methods are still packaged into Docker containers, but AGENTCO-OP attaches them to existing agent nodes as callable tools rather than instantiating them as standalone executor nodes, which avoids unnecessary inter-node communication for components that only expose a single entry point.

Interface synthesis. Finally, AGENTCO-OP synthesizes the interface protocol Π that governs communication along each edge of G. Every communication step is described by a structured message that records the sender, the receiver, a short summary, the main message body, and the path of any typed artifact passed between the two nodes. Typed artifacts include validated marker

tables, structured gene-set inputs, intermediate code files, and tool outputs serialized as JSON. The schema is enforced by the Broker nodes shown in Fig. 1, which validate that the artifact produced by an upstream node satisfies the expected schema before it is consumed downstream. This explicit schema is what allows independently developed components, including Docker-wrapped repositories, to exchange information through validated artifacts rather than free-form text.

A.1.3 Review Loop

Detect. AGENTCO-OP aggregates execution evidence into a small set of structured signals. Output signals capture node-level results and judge confidences, test signals capture pass and fail counts on validation cases, tool signals capture tool invocation errors and missing outputs, budget signals capture accumulated token cost relative to the per-task budget in r, and interface signals capture schema mismatches between artifacts produced by an upstream node and the schema expected by a downstream node. AGENTCO-OP flags a node as failing or uncertain when one or more of these signals cross a policy-specific threshold.

Decide. AGENTCO-OP then matches the observed evidence pattern against a small set of repair policies. Each policy maps an evidence pattern to a repair action. Examples include retrying with an updated prompt when a judge node returns low confidence, adding a parallel solver when a code node has persistent test failures, swapping the backend when tool errors recur on the same external service, and reformatting the output of an upstream node when a downstream artifact violates its schema. Policies are evaluated in priority order, and the first matching policy determines the action passed to the Repair step.

A.2 Experiments

A.2.1 Ablation Study

To verify the contribution of each component in AGENTCO-OP, we conduct an ablation study with results reported in Tab. 5. After removing the runtime local repair, most benchmarks show a drop in accuracy. The remaining two benchmarks fluctuate within a small margin, which suggests that local repair contributes most when tasks involve longer reasoning chains or precise generation. When we further remove agent skills and tools, the performance on most benchmarks remains close to the AGENTCO-OP without local repair. We attribute this to the nature of the standard benchmarks used here, which mainly require general reasoning and coding ability rather than specialized procedural knowledge or external operations. Agent skills and tools therefore have limited room to improve performance in this setting, although they are essential in the open-ended scientific scenarios.

Table 5: Ablation study on the components of AGENTCO-OP. **AC-Full** is the complete AGENTCO-OP system; **AC-NoLocalRepair** removes runtime gates and local repair; **AC-Minimal** removes agent skills, tools, runtime gates and local repair. All values are accuracy (%); bold marks the best result in each column.

Variant	Benchmarks						Avg.
	HotpotQA	DROP	HumanEval	MBPP	GSM8K	MATH	
AC-Full	76.5	77.2	**90.2**	**87.1**	**94.4**	**58.2**	**80.6**
AC-NoLocalRepair	**76.6**	77.4	87.9	86.8	93.2	56.6	79.8
AC-Minimal	76.0	77.0	88.6	86.0	93.9	51.7	78.9

A.2.2 Coordinate Domain Agent Collaboration

For this task, we evaluate whether AGENTCO-OP can coordinate independently developed domain agents to solve a collaborative scientific analysis problem. The biological question asks whether aFibro cells in the AVN/AV ring cellular community exhibit a distinct transcriptional program compared with aFibro cells in the left atria and right atria communities in a developing human heart MERFISH dataset [Farah et al., 2024]. Answering the question requires more than generic cell-type comparison: the workflow must identify spatially localized differential expression signals and then interpret the resulting marker genes to characterize the resulting transcriptional program.

Table 6: Summary of the serial collaboration case study.

Item	Result
External repositories	TissueAgent and GeneAgent
Workflow type	Linear handoff through a validated marker gene artifact
Dataset	Developing human heart MERFISH AnnData object
Dataset size	228635 cells and 238 genes
Target group	576 aFibro cells in the AVN/AV ring community
Control group	5685 aFibro cells in the Left Atria and Right Atria communities
Primary differential expression	Welch testing with Benjamini Hochberg correction
Primary marker rule	Adjusted $P < 0.05$ and positive $\log_2$ fold change
Primary marker count	53 AVN/AV ring upregulated markers
Sensitivity marker count	46 markers with a Mann Whitney test
Sanity check genes	DES, HAND2, IGFBP5, MYH6, MYH7, and NELL2 were all recovered
Broker output	Structured GeneAgent input with 53 genes and zero dropped genes
GeneAgent label	AV canal and node associated fibroblast program
Final interpretation	AVN/AV ring aFibro cells support a specialized developmental and conduction associated state rather than generic atrial stroma

This setting requires AGENTCO-OP to synthesize a collaborative workflow that composes two independently developed agents: TissueAgent and GeneAgent. TissueAgent is a role-based multi-agent framework that turns open-ended natural-language spatial transcriptomics requests and multimodal inputs into auditable, runnable workflows [ma-compbio, 2025]. GeneAgent is an agent for gene-set analysis that reduces hallucinations by autonomously interacting with biological databases to verify its own outputs [Wang et al., 2025c]. Together, these agents provide complementary expertise, but they were developed independently and expose different repositories, interfaces, execution environments, and output formats.

Therefore, this task is also significant from an agent-composition perspective. No single specialized agent is sufficient on its own: spatial transcriptomics analysis is needed to load the MERFISH data, select the relevant cellular communities, and perform differential expression, while gene-set interpretation is needed to determine whether the marker genes correspond to a meaningful biological program. The task naturally requires collaboration between a spatial transcriptomics agent and a gene-set analysis agent, making it a direct test of whether AGENTCO-OP can synthesize a coherent workflow across independently developed scientific agents with different inputs, outputs, and execution environments.

Given the task description, the two external GitHub repositories, and the public MERFISH dataset, AGENTCO-OP compiles a serial collaborative workflow. The synthesized workflow consists of repository profiling, sandbox construction, agent registration, TissueAgent execution, broker validation, GeneAgent execution, integration, and reporting. The workflow is linear because GeneAgent depends on the marker-gene artifact produced by the upstream differential expression analysis. To bridge this interface, the Broker validates the TissueAgent marker table and converts it into a structured JSON input containing all 53 human genes, with zero genes dropped. As a result, GeneAgent receives the marker set as a typed artifact rather than an unstructured free-text list.

The upstream execution loaded the MERFISH AnnData object without synthetic fallback. The object contained 228635 cells and 238 genes. The detected annotation fields were `populations`, `communities`, and `sample_id`. The target group was aFibro cells in the AVN/AV ring community. The control group was aFibro cells in the Left Atria and Right Atria communities. The target group contained 576 cells. The control group contained 5685 cells. The target cells were 295 in R78_4C12 and 281 in R77_4C4. The control cells were 2378 in R77_4C4, 1849 in R78_4C12, and 1458 in R78_4C15.

The primary analysis normalized expression to a total count of 10000 per cell and then applied log transformation. Differential expression used Welch testing across the 238 MERFISH genes. Multiple testing correction used Benjamini Hochberg adjustment. The resulting marker list began with DES, MYH6, IGFBP5, MYH7, HAND2, HCN4, TBX3, NELL2, COL9A2, and CD34. A sensitivity run used the Mann Whitney test and returned 46 markers. We report the Welch result as the primary analysis because it was the executed primary configuration. We report the Mann Whitney result as a sensitivity analysis because it shows that the exact marker count changes with the test choice.

Table 7: GeneAgent interpretation of the validated marker genes.

Subprocess	Supporting genes reported by GeneAgent
AV canal and conduction system specification	TBX3, TBX5, NKX2-5, HAND2, HCN4, HEY1, IRX4, SEMA6D
Extracellular matrix synthesis and remodeling	POSTN, FBLN2, FMOD, COL9A2, MMP11, CTSV, IGFBP5, TPBG
Epicardial and mesenchymal lineage identity	TCF21, WT1, PDGFRA, PRRX1, CD34, MECOM, TSHZ2
Contractile and ion channel signals	MYH6, MYH7, TTN, DES, CACNA1C, KCNH2, RRAD, CNN1
Neurogenic and axon guidance cues	NELL2, NRXN1, NEFL, NTS, PENK, SERPINI1, ADGRL1, BRINP3, SLC1A3, ADM
Developmental morphogen patterning	BMP2, INHBA, RSPO3, SFRP1, HHIP, BAMBI, CRABP2, MSX2

GeneAgent returned Markdown and JSON reports. Its self verification field listed Gene Ontology Biological Process, Reactome Pathways, UniProt and NCBI Gene summaries, Human Protein Atlas, and literature evidence on AV canal and AV node development. The final integrator combined the differential expression evidence and the GeneAgent report. It also retained caveats about cardiomyocyte-like transcripts, possible spatial admixture, and the need for orthogonal validation. These caveats are important because genes such as MYH6, MYH7, TTN, and DES may reflect proximity to nodal or transitional myocardium. The main systems conclusion is not affected by this caveat. AGENTCO-OP converted two external repositories into coordinated execution nodes and preserved an auditable handoff from marker discovery to gene set interpretation.

A.2.3 Compose Domain Workflows

For this task, we evaluate whether AGENTCO-OP can compose existing domain workflows to solve a collaborative cross-modality analysis problem. The biological question asks whether integrating cell-type marker signals from paired RNA and chromatin accessibility modalities can yield more reliable cell-type marker identification than either modality alone on a single-cell multiome dataset. Answering this question requires more than running a single analysis pipeline. The workflow needs to independently identify markers from gene expression and from chromatin accessibility, and then reconcile their outputs against curated cell-type marker references.

This setting requires AGENTCO-OP to synthesize a collaborative workflow that composes two domain workflows, Seurat and Signac. Seurat is a widely used analysis framework for single-cell RNA-seq that supports normalization, clustering, and differential expression on gene expression assays [Butler et al., 2018]. Signac extends single-cell analysis to chromatin accessibility, providing modality-specific quality control and a GeneActivity function that summarizes accessibility into per-gene scores [Stuart et al., 2021]. Together, the two packages provide complementary expertise on the two modalities, but they are maintained as separate repositories with distinct dependencies, and integrating them into an automated workflow typically requires manual scripting and environment management.

Therefore, this task offers a meaningful workflow-composition perspective. The two modality-specific workflows can be executed in parallel since neither workflow consumes the other's output, while their results still need to be reconciled at the cell-type level to evaluate the integrated marker set. The task therefore naturally requires parallel coordination of two domain workflows followed by a join step, making it a direct test of whether AGENTCO-OP can synthesize a coherent cross-modality workflow across existing domain repositories with different inputs and execution environments.

The inputs to the task include the Seurat and Signac GitHub repositories and official tutorials, the 10x PBMC multiome data files, the Hao PBMC reference data [Hao et al., 2021], and two marker databases, CellMarker 2.0 and PanglaoDB. AGENTCO-OP compiles a parallel-then-join workflow with repository profiling, sandbox preparation, agent registration, data inspection, independent Seurat and Signac execution, broker validation, marker set evaluation, integration, and reporting. The broker records two valid typed handoffs, so the cross-modality result is evaluated through structured artifacts rather than free text.

Table 8: PBMC workflow and marker discovery summary.

Quantity	Value
Dataset	10x Genomic multiome dataset for human PBMCs
Assays	Gene expression and chromatin accessibility
Raw cells	11909
Post-QC cells	11070
Annotated cells before marker filter	5000
Cells used for marker discovery	4777
Cell types used for marker discovery	22
RNA marker rows	26353
ATAC gene activity marker rows	45731
Top markers per modality and cell type	50
Mean RNA and ATAC intersection size	11.55
Mean RNA and ATAC union size	88.45
Mean Jaccard index	0.133

Table 9: Macro precision and recall results for the PBMC case study.

Database	Cell types	$P_{\cap}$	P_{RNA}	P_{ATAC}	$R_{\cup}$	R_{RNA}	R_{ATAC}
CellMarker 2.0	22	0.303	0.195	0.110	0.124	0.102	0.061
PanglaoDB	22	0.333	0.231	0.131	0.117	0.097	0.054

The intended annotation path transfers `celltype.l2` labels from the Hao reference. This path does not complete on the execution host, triggering AGENTCO-OP's self-correction, which switches to a SingleR-based fallback with the MonacoImmuneData reference. The fallback produces 5000 annotated cells with 27 fine immune labels. After applying a minimum of 30 cells per label to ensure stable marker discovery, 4777 cells across 22 labels are retained for marker discovery. As a result, the quantitative evaluation reflects Monaco-derived immune labels rather than Hao `celltype.l2` labels. A summary of the workflow and marker discovery statistics is provided in Tab. 8.

For each evaluated cell type, the evaluator constructs four marker sets, namely RNA markers, ATAC gene activity markers, their intersection, and their union. For a predicted marker set M and a database marker set D, precision is computed as $|M \cap D|/|M|$ and recall as $|M \cap D|/|D|$. The precision comparison uses the intersection, RNA, and ATAC marker sets, and the recall comparison uses the union, RNA, and ATAC marker sets. The macro-averaged scores across cell types are summarized in Tab. 9. At the per-cell-type level, on CellMarker 2.0, the intersection improves precision over both single modalities for 17 of 22 cell types, and the union improves recall over both single modalities for 16 of 22 cell types. On PanglaoDB, the intersection improves precision for 15 of 22 cell types, the union improves recall for 18 of 22 cell types. AGENTCO-OP thus composes two specialized domain repositories into coordinated parallel execution nodes and aligns their cross-modality marker outputs.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims of the paper are demonstrated by the theoretical and experimental results in Sections 3 and 4, which are consistent with the claims made in the abstract and introduction. The paper does not make any claims that are not supported by the results.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

Justification: The limitations are discussed in the Limitation and future works section.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

   Justification: The paper does not include any heoretical results.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Yes, we have provided detail method descriptions and figure of the workflow pipeline.

   Guidelines:

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

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

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

   Answer: [No]

   Justification: The data is open-sourced and the code will be released upon publication. The instructions for reproducing the main experimental results are provided in the paper.

   Guidelines:

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

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The datasets and benchmarks are fully open-sourced. We use the same data processing method and split as previous work.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [N/A]

   Justification: We do not show the error bars in the experiments.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide cost analysis in the experiment section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our research conforms to it in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We provide the broader impacts of our work in the discussion.

    Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper does not have any safety risk.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All the code, data, and models used in or related to this work are properly cited and discussed.

Guidelines:

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

13. **New assets**

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

Answer: [N/A]

Justification: This paper does not release new assets during the review process.944

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects. The datasets used in this paper are publicly available.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

Answer: [N/A]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs.

Guidelines:

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