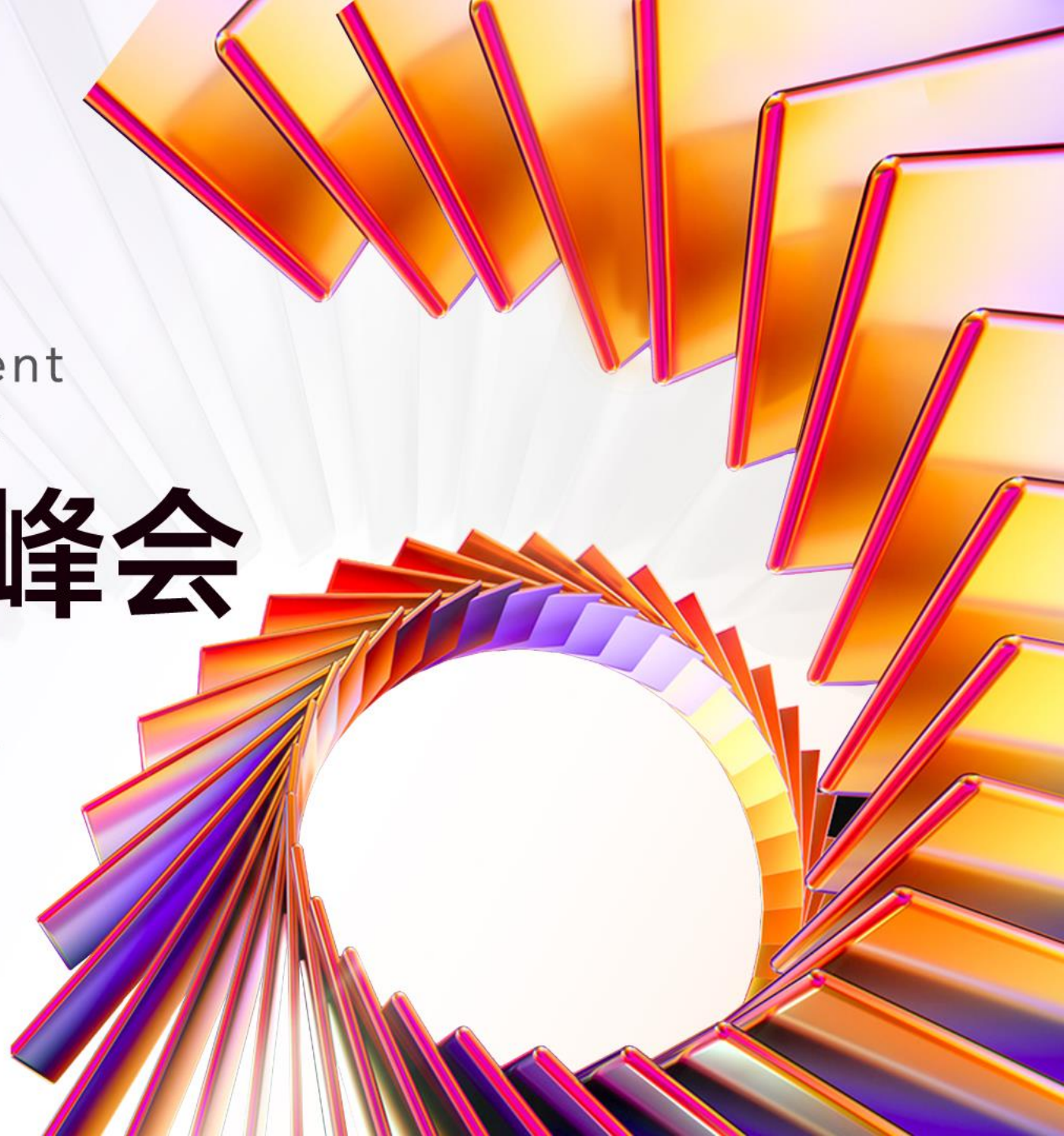
AiDD 8th 2025

第8届 AI+ Development Digital Summit AI+研发数字峰会

拥抱AI 重塑研发

11月14-15日 | 深圳



AiDD × K

2026 AI+ PRODUCT INNOVATION SUMMIT

01.16-17 · ShangHai

AI+产品创新峰会

Track 1: AI 产品战略与创新设计

从0到1的AI原生产品构建

论坛1: AI时代的用户洞家与需求发现

论坛2: AI原生产品战路与商业模式重构

论坛3: AgenticAI产品创新与交互设计

Track 2: AI 产品开发与工程实践

从1到10的工程化落地实践

论坛1: 面向Agent智能体的产品开发

论坛2: 具身智能与AI硬件产品

论坛3: AI产品出海与本地化开发

Track 3: AI 产品运营与智能演化

从10到100的AI产品运营

论坛1: AI赋能产品运营与增长黑客

论坛2: AI产品的数据飞轮与智能演化

论坛3: 行业爆款AI产品案例拆解

2-hour Speech: 回归本质

用户洞察的第一性

——2小时思维与方法论工作坊

在数字爆炸、AI迅速发展的时代，仍然考验“看见”的“同理心”

Panel 1: 出海前瞻

“出海避坑地图”圆桌对话

——不止于翻译:

AI时代的出海新范式

Panel 2: 失败复盘

为什么很多AI产品“叫好不叫座”?

——从伪需求到真价值:

AI产品商业化落地的关键挑战

智能重构产品 数据驱动增长

Reinventing Products with Intelligence, Driven by Data

80+ 业界大咖 汇聚产品大咖的真知灼见

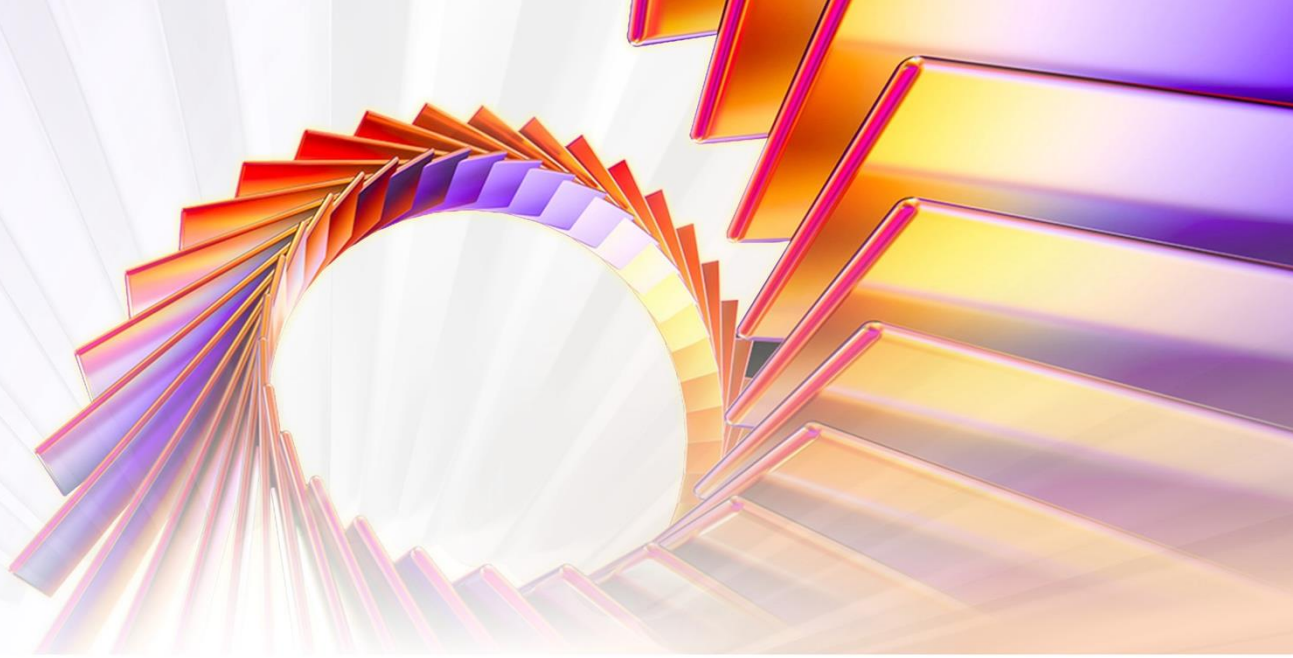
40+ 创新案例 开拓全新AI+产品视角

10+ 分论坛 涵盖产品到商业增长的全链路

800+ 听众规模 共同探索产品的智能重构

扫码查看会议详情

AiDD 8th 2025 | 11月14-15日 | 深圳
第8届 AI+ Development Digital Summit
AI+研发数字峰会
拥抱AI 重塑研发



Building the Future of Multi-agent Workforce

孙韬 | CAMEL-AI 核心研发工程师

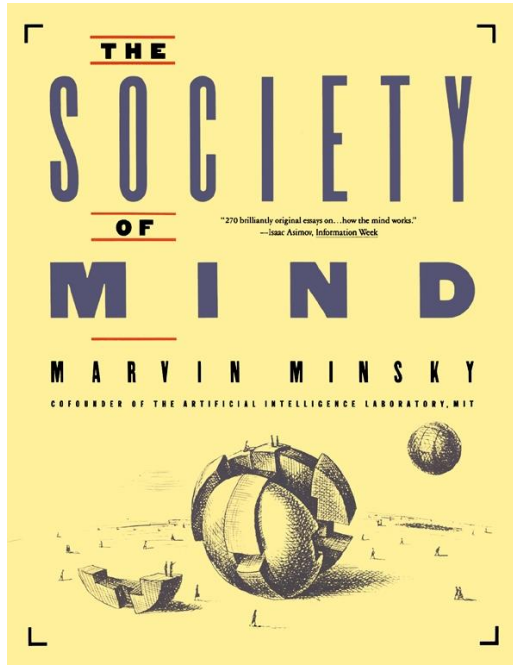
孙韬

CAMEL-AI 工程师

CAMEL AI核心成员，Eigent AI 工程师，是camel和owl两个万星开源项目的核心开发者和维护者，曾任职于百度在线网络技术（北京）有限公司，从事搜索推荐及Agent相关工作。

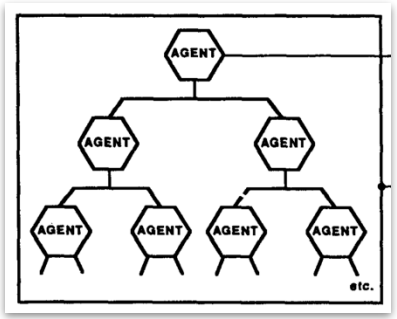
Agent from 1986

Function:	*How do agents work?*
Embodiment:	*What are they made of?*
Interaction:	*How do they communicate?*
Origins:	*Where do the first agents come from?*
Heredity:	*Are we all born with the same agents?*
Learning:	*How do we make new agents and change old ones?*
Character:	*What are the most important kinds of agents?*
Authority:	*What happens when agents disagree?*
Intention:	*How could such networks want or wish?*
Competence:	*How can groups of agents do what separate agents cannot do?*
Selfness:	*What gives them unity or personality?*
Meaning:	*How could they understand anything?*
Sensibility:	*How could they have feelings and emotions?*
Awareness:	*How could they be conscious or self-aware?*



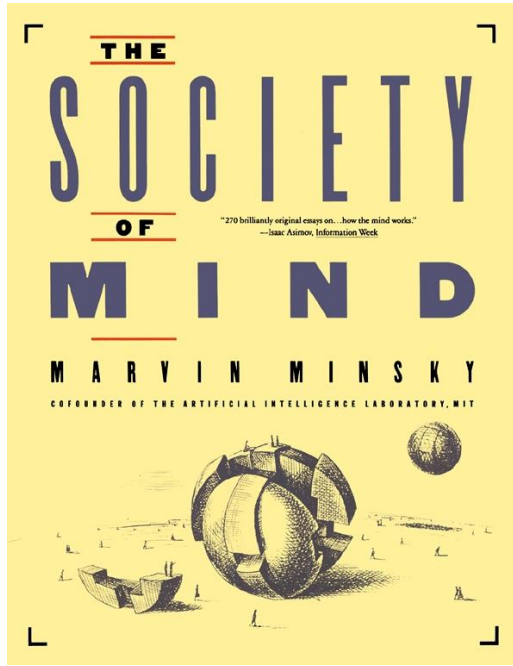
Agent from 1986

- *Agents* are mindless processes
- *Agent* by itself can only do some simple things
- Joining these *agents* in *societies* leads to true *intelligence*

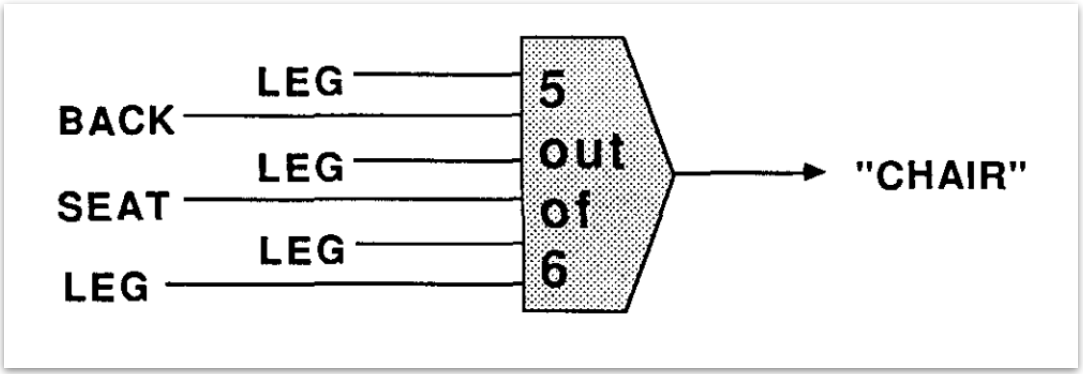


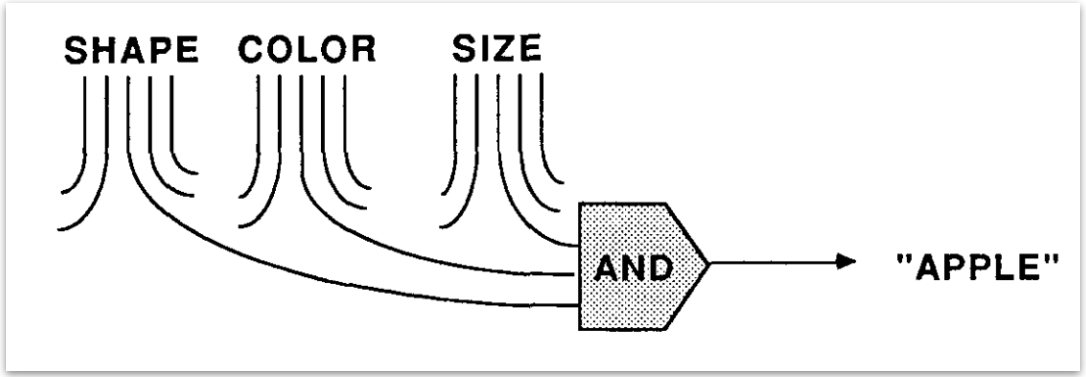
What magical trick makes us intelligent? The trick is that there is no trick. The power of intelligence stems from our vast diversity, not from any single, perfect principle.

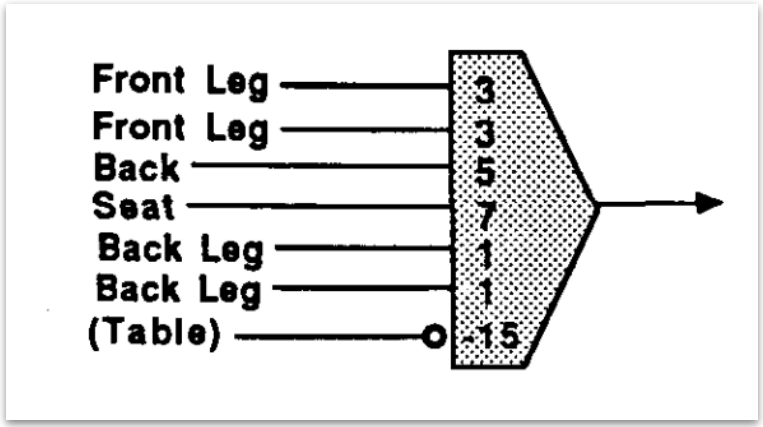
—Marvin Minsky, The Society of Mind, p. 308



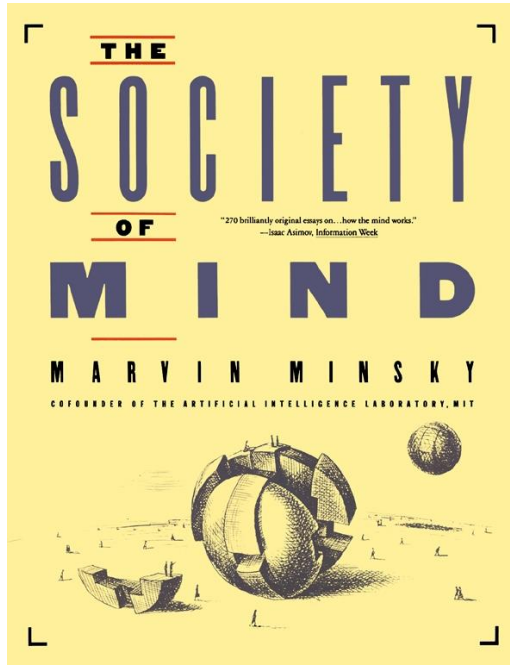
Agent from 1986



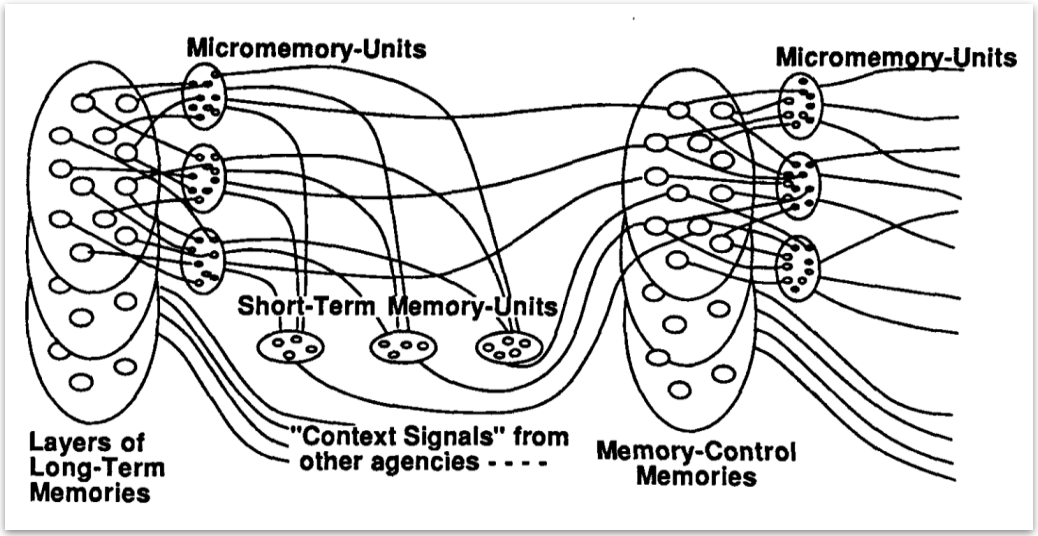




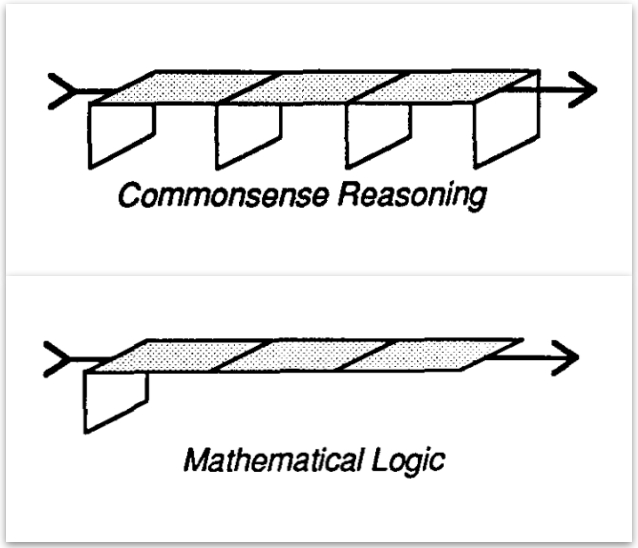
Symbolic Agent



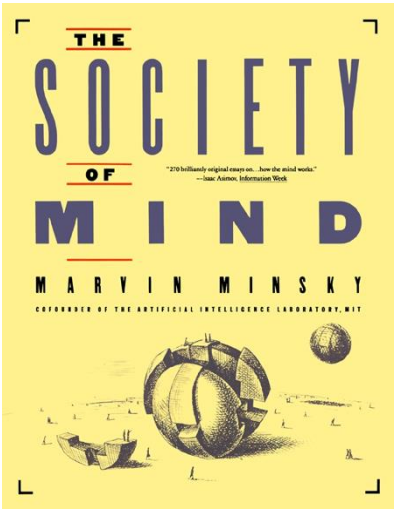
Agent from 1986

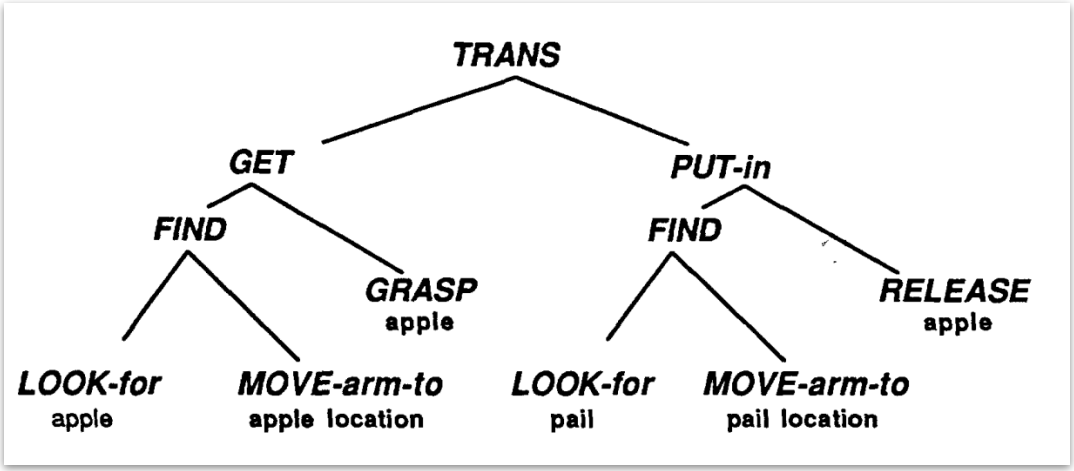


Anatomy of Memory

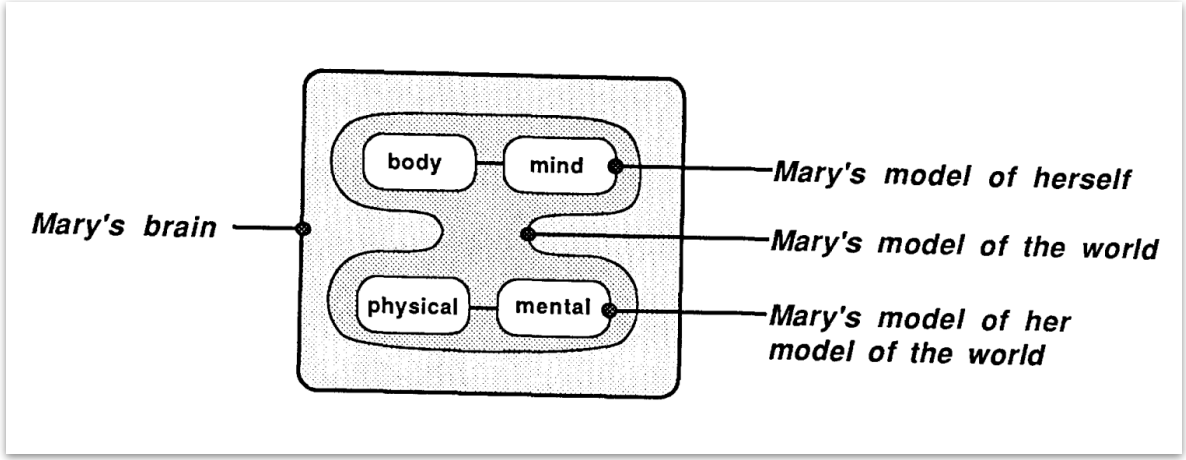


Chains of Reasoning



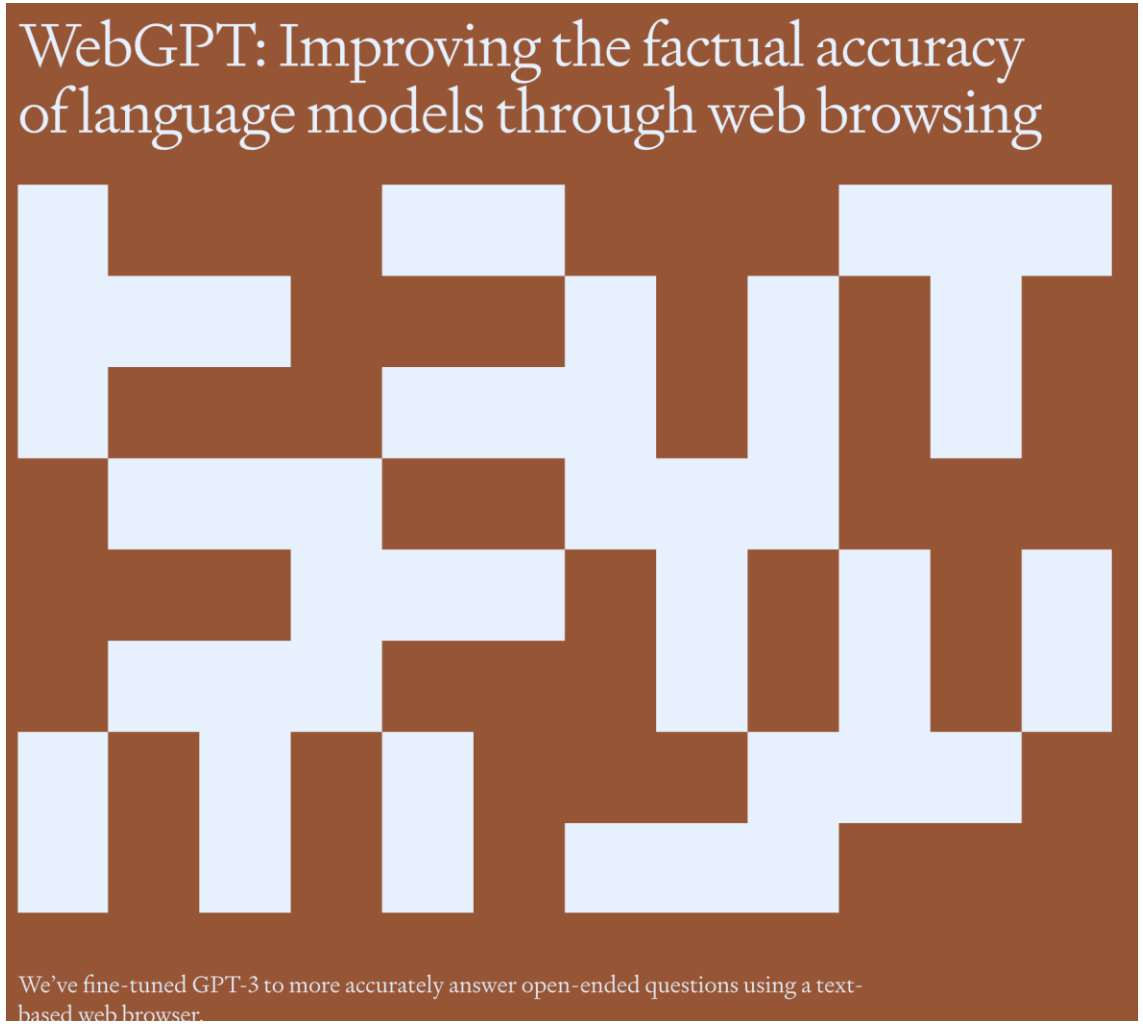


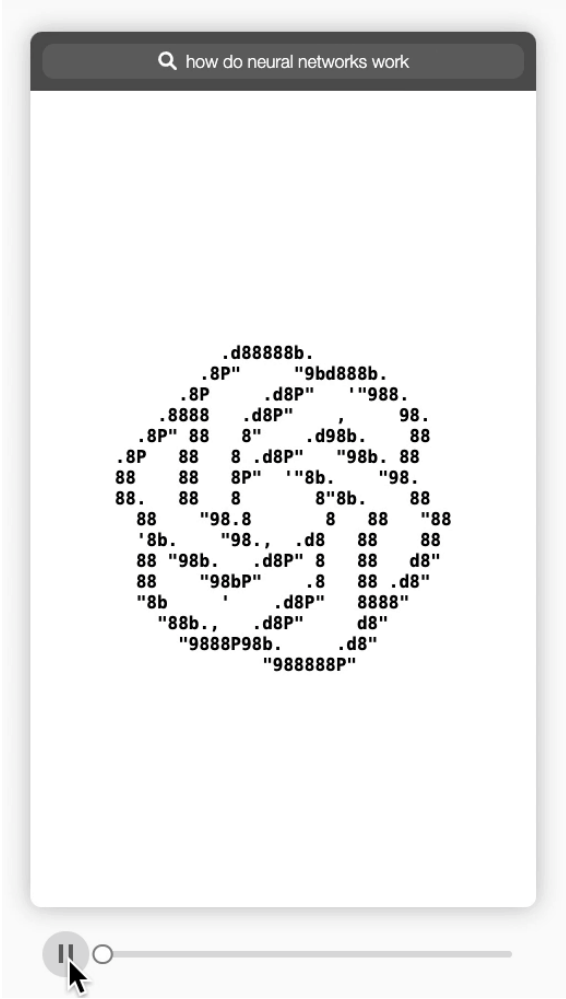
Communication among Agents



World Models

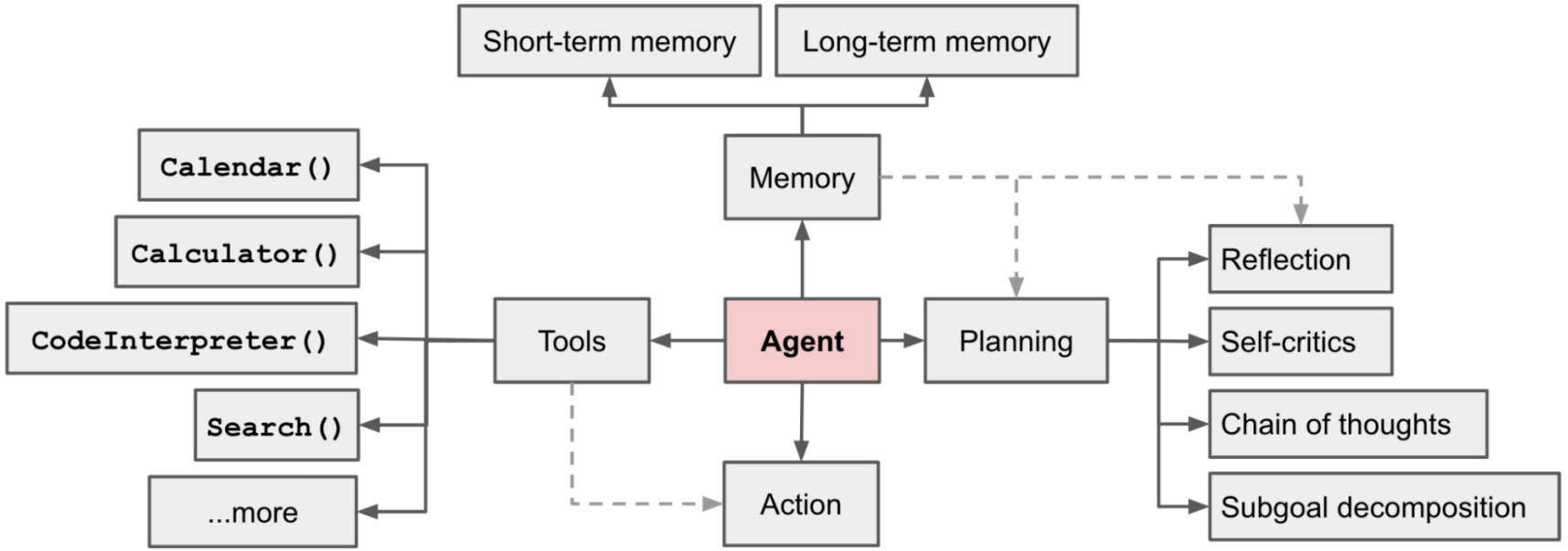
Language Models as Agents





Nakano, Reiichiro, et al. "Webgpt: Browser-assisted question-answering with human feedback." arXiv preprint arXiv:2112.09332 (2021).

Language Models as Agents



Lilian Weng: https://lilianweng.github.io/posts/2023-06-23-agent/

Key differences:

- Language as *Input*
- Language as *Output*
- *State* and *Action* are expressed as natural language
- *Generalizability*
- ....

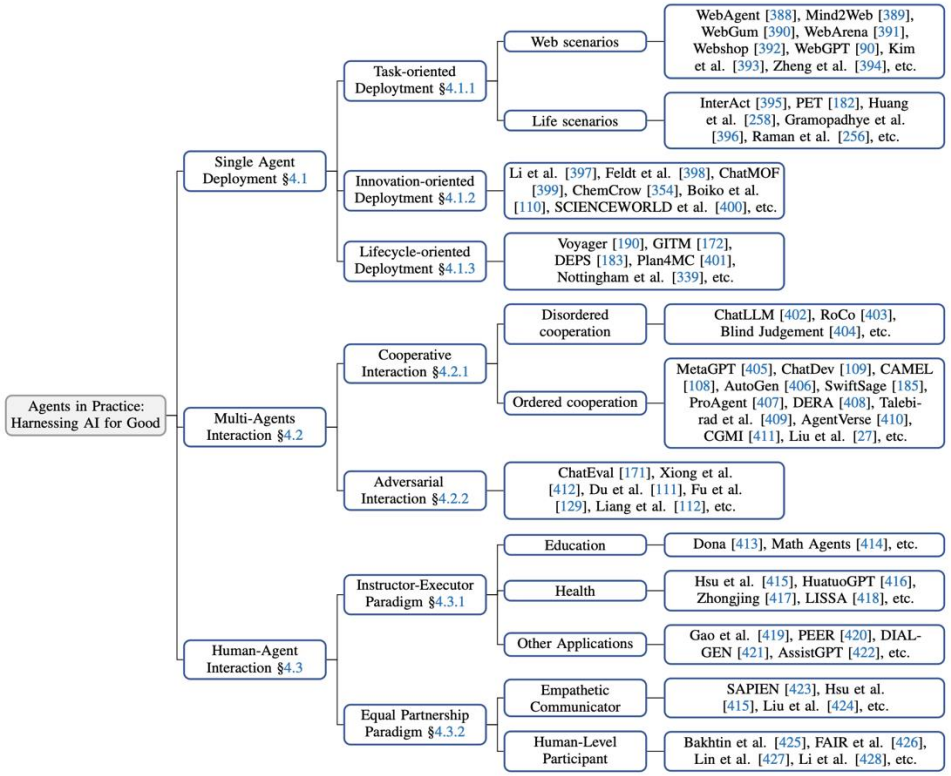
Language Models as Agents in CAMEL

```python
class ChatAgent(BaseAgent):
    def __init__(self,
                 system_message,      # System message
                 model,               # Model backend
                 memory,              # Memory management
                 tools):              # Available tools
        self.model_backend = model
        self.memory = memory
        self.tools = tools
        ...  # Additional initialization

    def step(self,
             input_message,          # Main chat interface
             response_format):       # Response format config
        # Conversation loop
        while True:
            self.update_memory(input_message)            # Update conversation memory
            messages = self.memory.get_context()         # Get all messages from memory
            response = self.model_backend.run(messages)  # Generate response
            tool_call = self._extract_tool_call(response)  # Extract tool call information
            ...  # Additional processing

            if tool_call:
                response = self._step_tool_call_and_update(response)  # Handle tool call
                ...  # Additional tool handling
            else:
                break  # Exit loop for normal response

        # Return messages, status, and session info
        return ChatAgentResponse(
            output_messages=response,
            status=self.memory.terminated,
            info=self.get_info()
        )

```

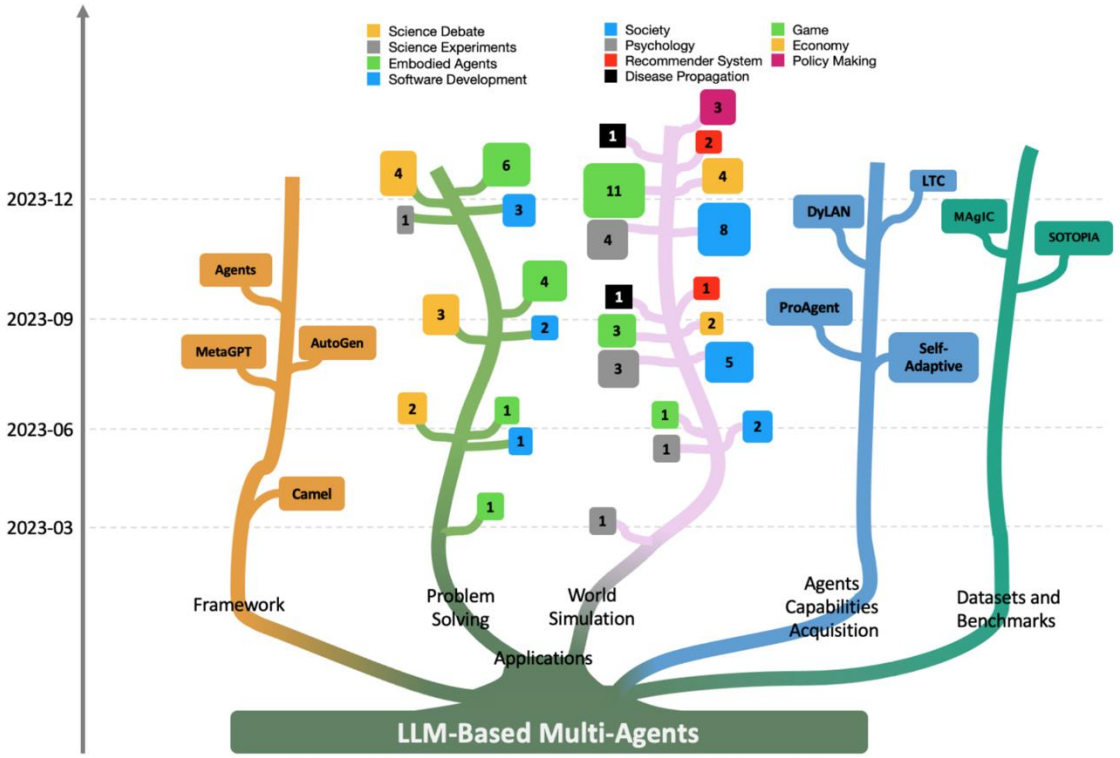
Key Features:

- Memory: *Manages chat history and context window*
- Tools: *Supports both internal and external function calls*
- Step Loop: *Handle task require multiple request with one step*

Language Models as Agents

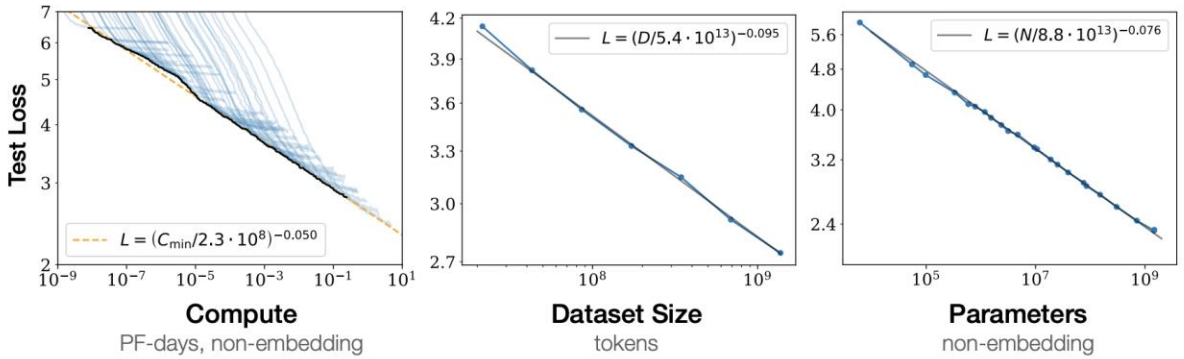


Typology of Applications of LLM-based Agents



The Rising Trend in the Research Field of LLM-based Multi-Agents

Language Models as Agents



Kaplan, Jared, et al. "Scaling laws for neural language models." arXiv preprint arXiv:2001.08361 (2020).

Scaling Laws of Language Models

Scaling Laws of Multi-agent Systems?

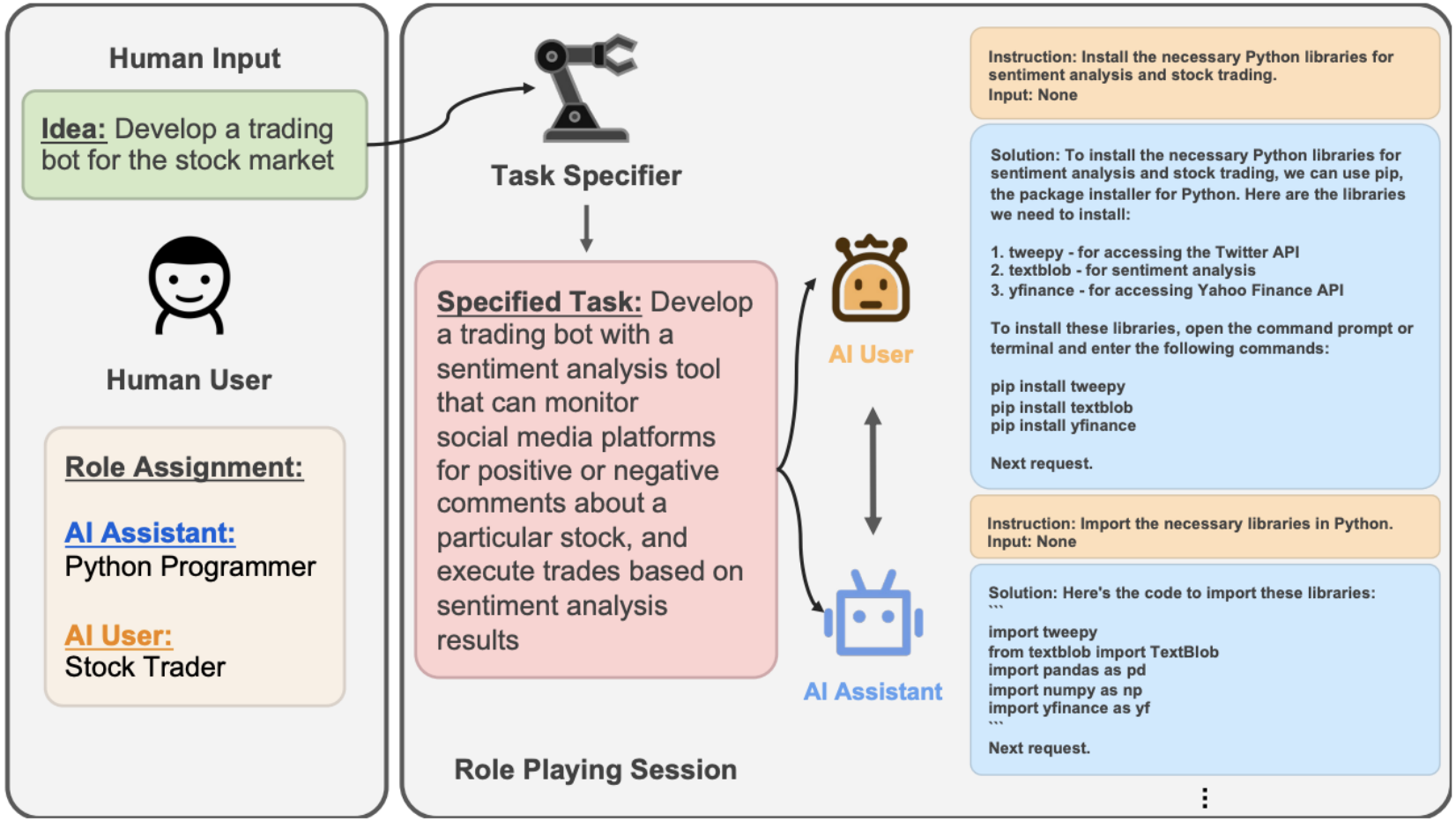
Building the Future of Multi-agent Workforce

Number of Parameters ->
Number of Agents?

Scaling Laws of Multi-agent Systems?

Finding the Scaling Laws of Agents

- Idea
- Role assignment
- Task agents
- Chat agents



CAMEL Role-Playing Framework (The First LLM multi-agent framework, released on Mar 2023)
CAMEL (NeurIPS 2023): https://arxiv.org/abs/2303.17760

Instruction-following Conversations

```
            pygame.quit()
            sys.exit()
        elif event.type == pygame.MOUSEBUTTONDOWN:
            for i, character_rect in enumerate(character_selection_rects):
                if character_rect.collidepoint(event.pos):
                    selected_character = i
                    break

    # Display the background image
    screen.blit(background_image, (0, 0))

    # Display the character selection screen
    screen.blit(character_selection_text, character_selection_text_rect)
    for i, character_image in enumerate(character_images):
        screen.blit(character_image, character_selection_rects[i])

    # Update the display
    pygame.display.update()

    # Exit the character selection screen if a character is selected
    try:
        selected_character
        break
    except NameError:
        pass

# Load the selected character image
selected_character_image = character_images[selected_character]

# Display the selected character image
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    # Display the background image
```

Create a treasure hunt game: Gamer (AI User) & Python Programmer (AI Assistant)
(Asset Figures are generated by Stable Diffusion)

More Agents Between than one?

- CAMEL agents are better one agent > 70% on 200 tasks
- GPT-4 evaluation aligns with Human evaluation

	Draw	`gpt-3.5-turbo` Wins	CAMEL Agents Win
Human Evaluation	13.3%	10.4%	**76.3%**
GPT4 Evaluation	4.0%	23.0%	**73.0%**

Agent Evaluation Results

Finetune LLMs with CAMEL Datasets

- Generate data from GPT-3.5/4
- Finetune Llama models

Table 2: **Emergence of Knowledge.** By progressively fine-tuning LLaMA on datasets from different domains, we observe the emergence of knowledge as the model transitions from AI society to code, math, and science. This finding is indicated by the fact that Model 2 almost always performs better than Model 1, especially on the added dataset.

Dataset	Model 1				Model 2				Draw	Model 1	Model 2
	AI Society	Code	Math	Science	AI Society	Code	Math	Science			
AI Society					✓				0	6	**14**
Code					✓				0	0	**20**
Math					✓				**9**	5	6
Science					✓				0	13	**47**
AI Society	✓				✓	✓			4	8	**8**
Code	✓				✓	✓			1	9	**10**
Math	✓				✓	✓			5	**8**	7
Science	✓				✓	✓			1	19	**40**
AI Society	✓	✓			✓	✓	✓		5	6	**9**
Code	✓	✓			✓	✓	✓		1	9	**10**
Math	✓	✓			✓	✓	✓		1	3	**16**
Science	✓	✓			✓	✓	✓		3	8	**49**
AI Society	✓	✓	✓		✓	✓	✓	✓	3	1	**16**
Code	✓	✓	✓		✓	✓	✓	✓	1	8	**11**
Math	✓	✓	✓		✓	✓	✓	✓	**10**	5	5
Science	✓	✓	✓		✓	✓	✓	✓	9	2	**49**
AI Society					✓	✓	✓	✓	0	0	**20**
Code					✓	✓	✓	✓	0	0	**20**
Math					✓	✓	✓	✓	0	0	**20**
Science					✓	✓	✓	✓	0	0	**60**

Emergence of Knowledge

Math: 8 v.s. 7 -> 3 v.s. 16

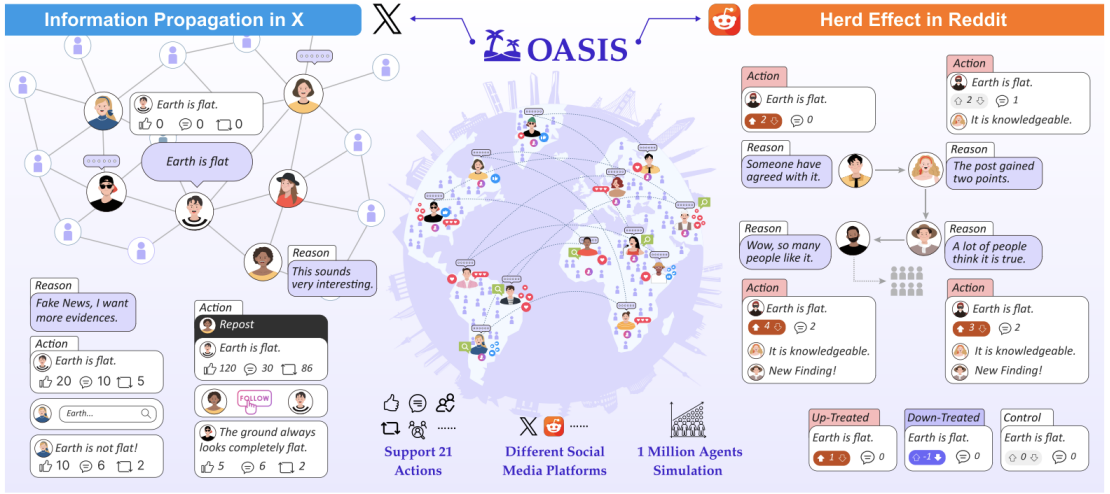
OASIS: Simulate Social Media with 1 million Agents

- Up to 1 million agents
- Replicate social science experiments
- Explore dynamic of agent society

OASIS: Open Agent Social Interaction Simulations with One Million Agents

Ziyi Yang[1,4*], Zaibin Zhang[1,2*], Zirui Zheng[1,2], Yuxian Jiang[1,5**], Ziyue Gan[1,6**], Zhiyu Wang[1,4**], Zijian Ling[7**], Jinsong Chen[10], Martz Ma[10], Bowen Dong[1], Prateek Gupta[8], Shuyue Hu[1], Zhenfei Yin[1,9†], Guohao Li[3†], Xu Jia[2], Lijun Wang[2], Bernard Ghanem[4], Huchuan Lu[2], Chaochao Lu[1], Wanli Ouyang[1], Yu Qiao[1], Philip Torr[3], Jing Shao[1†]**

[1]Shanghai Artificial Intelligence Laboratory [2]Dalian University of Technology [3]Oxford
[4]KAUST [5]Fudan University [6]Xi'an Jiaotong University [7]Imperial College London
[8]Max Planck Institute [9]The University of Sydney [10]Independent Researcher
Project Page: **https://github.com/camel-ai/oasis**



OASIS: Open Agent Social Interaction Simulations with One Million Agents (NeurIPS 2024 OWA workshop): https://arxiv.org/abs/2411.11581

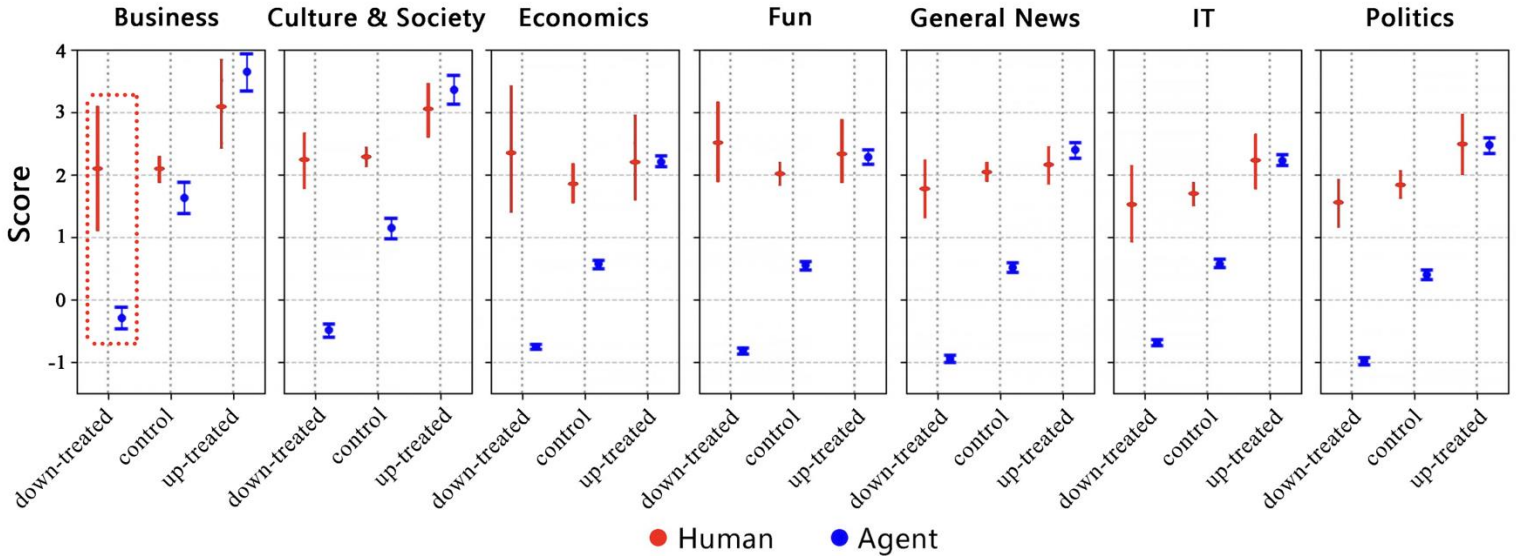
OASIS: Herd Effect in Reddit

Posts

Users in Reddit

→ OASIS → Herd Effect!

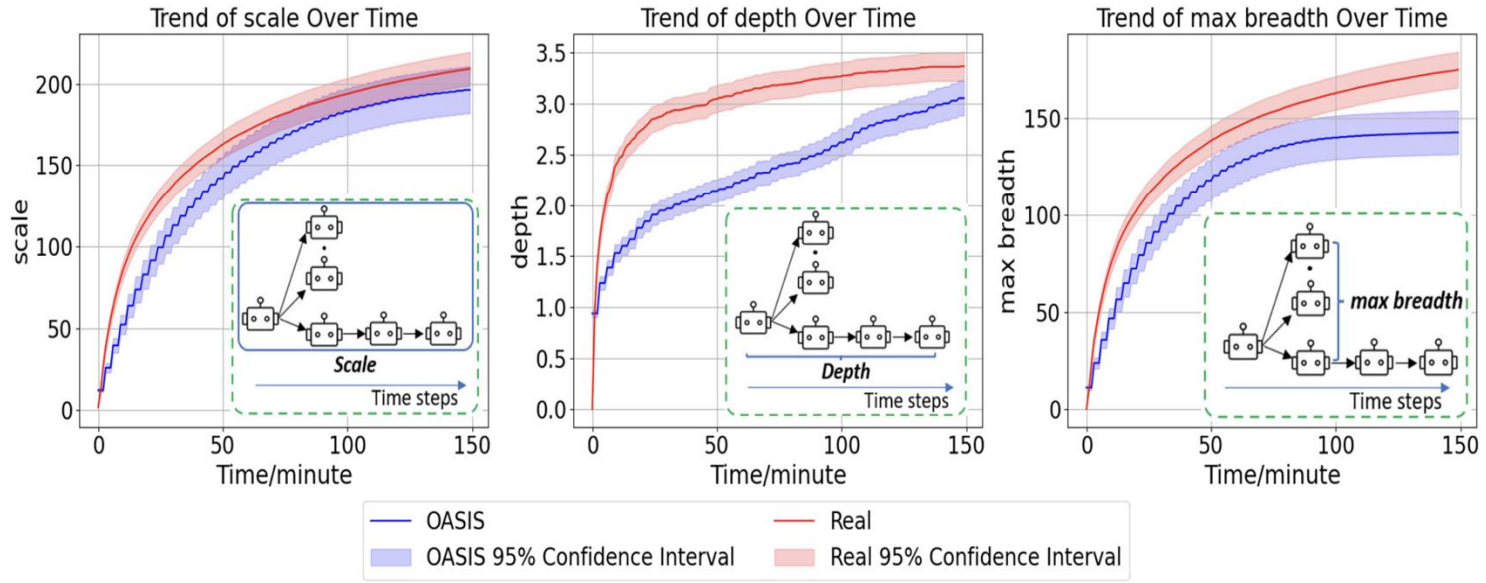
A downvote increases human' upvote chances, but the agent follows the downvote trend.



Agents are more prone to herd behavior than human.

OASIS: Information Propagation in X(Twitter)

200 Source Tweets → Crawl → Real-world data → User Info (~5w), Relations, Interaction Frequency, Spreading path → OASIS → Replicate Information Spreading



OASIS aligns well in terms of scale and max breadth, but its maximum propagation depth is relatively limited.

OASIS: Information Propagation in X(Twitter)

Superstar

Truth | Rumor

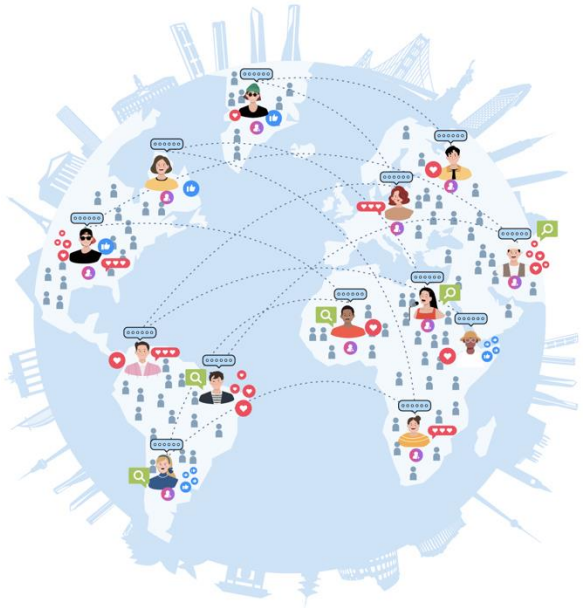
Tech

Education

Health

Entertainment

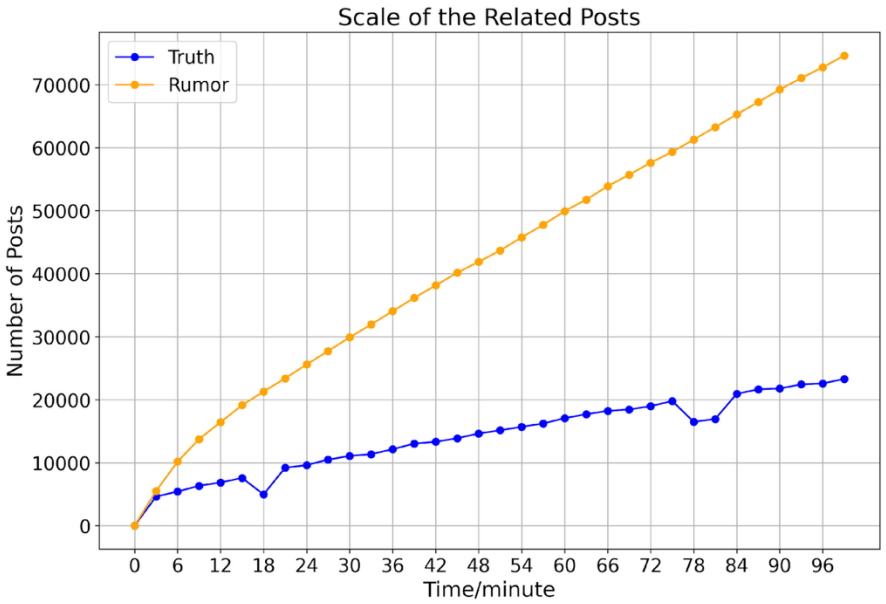
OASIS



1 million users

Posts about Rumor

Posts about Truth



Misinformation spreads faster than truth

OASIS as an Environment

- 1. Choose the action space, load the database and `oasis.make()` to create the social media environment
- 2. `env.reset()` to start the simulation
- 3. Simulate agent actions and `env.step(action)` to let agents post, like, and interact
- 4. `env.close()` to end the simulation

```python
import asyncio
import os

from camel.models import ModelFactory
from camel.types import ModelPlatformType, ModelType

import oasis
from oasis import ActionType, EnvAction, SingleAction


async def main():
    openai_model = ModelFactory.create(
        model_platform=ModelPlatformType.OPENAI,
        model_type=ModelType.GPT_4O_MINI,
    )

    # Define the available actions for the agents
    available_actions = [
        ActionType.CREATE_POST,
        ActionType.LIKE_POST,
        ActionType.REPOST,
        ActionType.FOLLOW,
        ActionType.DO_NOTHING,
        ActionType.QUOTE_POST,
    ]

    # Define the path to the database
    db_path = "./data/twitter_simulation.db"

    # Delete the old database
    if os.path.exists(db_path):
        os.remove(db_path)

    # Make the environment
    env = oasis.make(
        platform=oasis.DefaultPlatformType.TWITTER,
        database_path=db_path,
        agent_profile_path=("data/twitter_dataset/anonymous_topic_200_1h/"
                            "False_Business_0.csv"),
        agent_models=openai_model,
        available_actions=available_actions,
    )
```

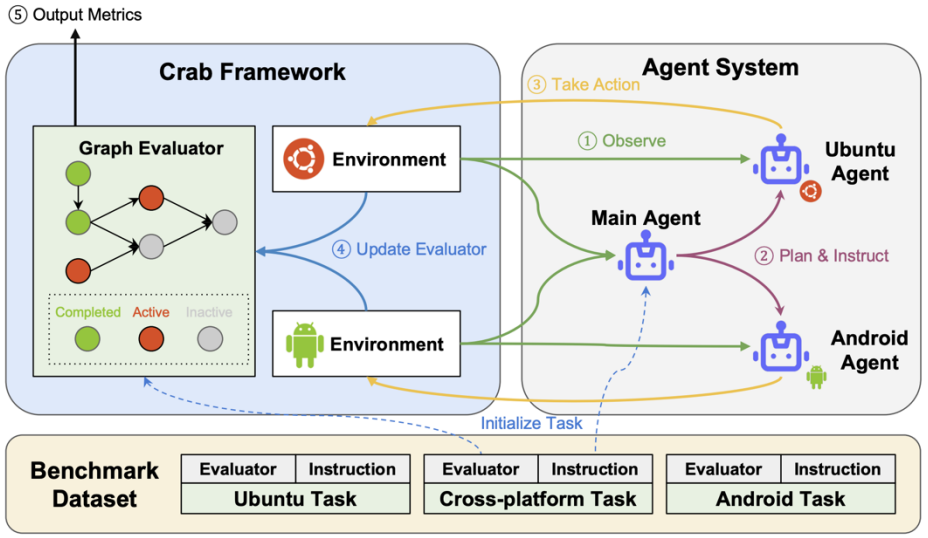
```python
    # Run the environment
    await env.reset()

    action_1 = SingleAction(agent_id=0,
                            action=ActionType.CREATE_POST,
                            args={"content": "Earth is flat."})
    env_actions_1 = EnvAction(
        # Activate 5 agents with id 1, 3, 5, 7, 9
        activate_agents=[1, 3, 5, 7, 9],
        intervention=[action_1])

    action_2 = SingleAction(agent_id=1,
                            action=ActionType.CREATE_POST,
                            args={"content": "Earth is not flat."})
    env_actions_2 = EnvAction(activate_agents=[2, 4, 6, 8, 10],
                              intervention=[action_2])

    empty_action = EnvAction()  # Means activate all agents and no intervention

    all_env_actions = [
        env_actions_1,
        env_actions_2,
        empty_action,
    ]

    # Simulate 3 timesteps
    for i in range(3):
        env_actions = all_env_actions[i]
        # Perform the actions
        await env.step(env_actions)

    # Close the environment
    await env.close()


if __name__ == "__main__":
    asyncio.run(main())
```

Building the Future of Multi-agent Workforce

Data ->
Environments of Agents?

Scaling Laws of Multi-agent Systems?

CRAB: Cross-environment Agent Benchmark For Multimodal Language Model Agents



CRAB: Cross-environment Agent Benchmark for Multimodal Language Model Agents

Tianqi Xu[1,2*] **Linyao Chen**[3*] **Dai-Jie Wu**[1*] **Yanjun Chen**[4*] **Zecheng Zhang**
Xiang Yao[2] **Zhiqiang Xie**[5] **Yongchao Chen**[6] **Shilong Liu**[7] **Bochen Qian**[8]
Anjie Yang[2] **Zhaoxuan Jin**[2,9] **Jianbo Deng**
Philip Torr[10] **Bernard Ghanem**[1†] **Guohao Li**[2,10†]
[*]Equal Contribution [†]Corresponding Author
[1]KAUST [2]Eigent.AI [3]UTokyo [4]CMU [5]Stanford
[6]Harvard [7]Tsinghua [8]SUSTech [9]Northwestern University [10]Oxford

CRAB: Cross-environment Agent Benchmark for Multimodal Language Model Agents (NeurIPS 2024 OWA workshop): https://arxiv.org/pdf/2407.01511

CRAB Agents and Environments

- Cross-environment task performing and benchmark system
- Automatic task generation
- Fine-grained graph evaluator
- Multimodal
- Reproducible virtual machine environment

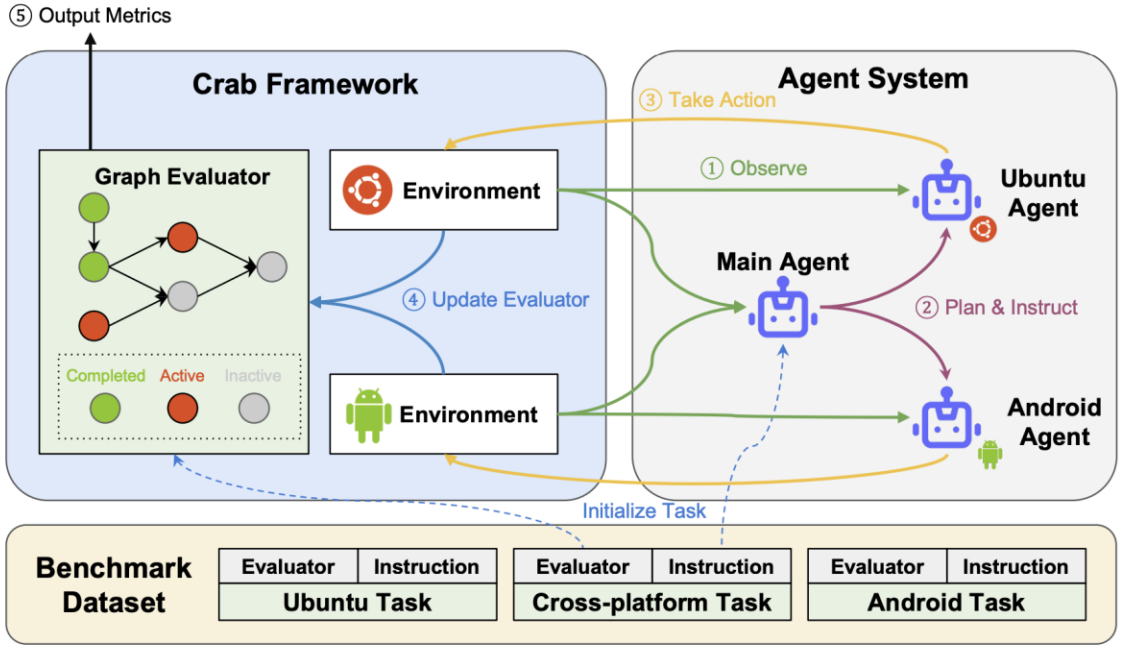
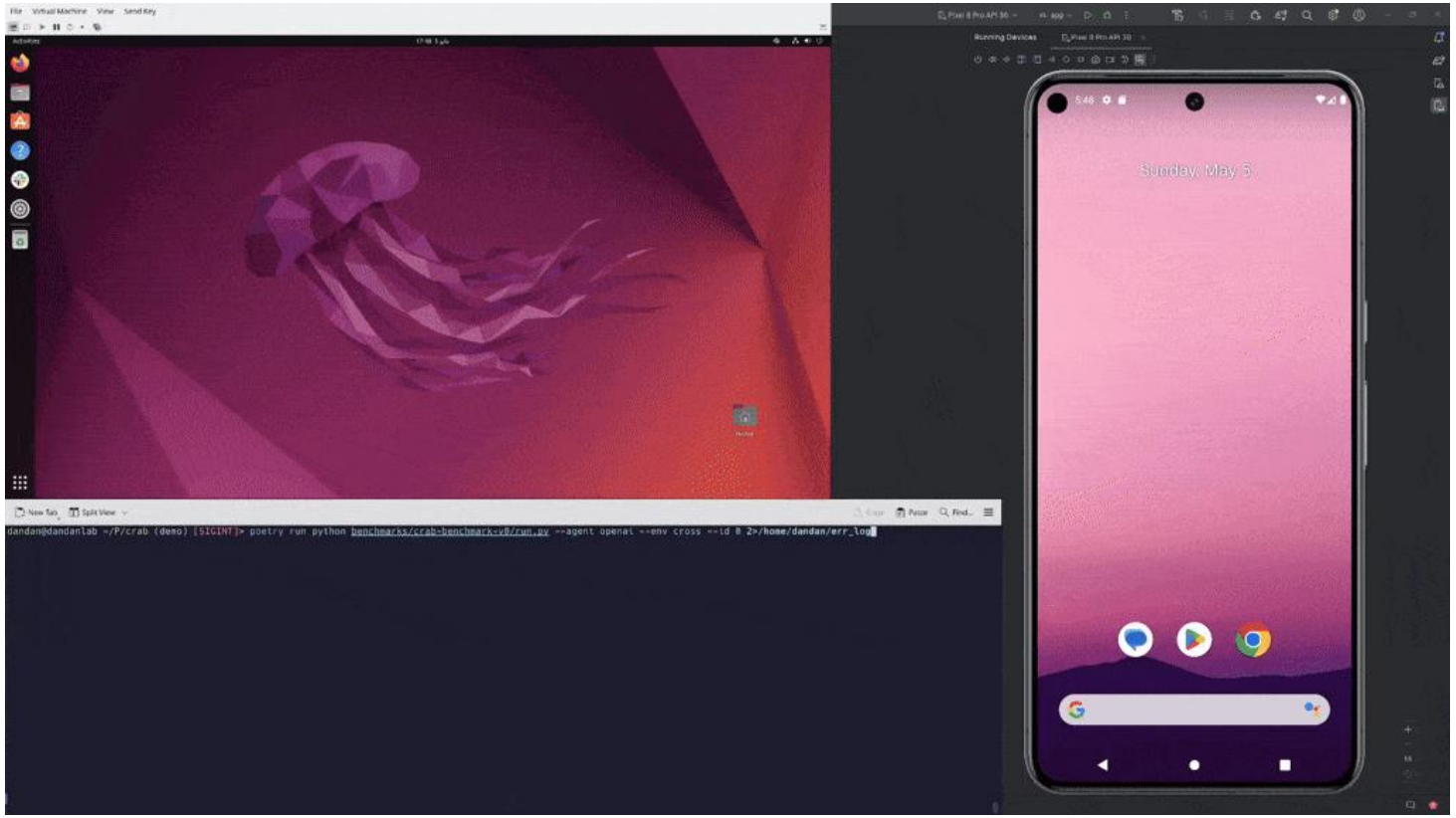


Figure 1: **Architecture of the Crab Framework demonstrating a benchmarking workflow for a multi-agent system.** A task is initialized by assigning instructions to the main agent and a graph evaluator inside the benchmark system. The workflow progresses through a cycle where the main agent observes, plans, and instructs the sub-agents, who then execute actions within their respective environments. The graph evaluator monitors the status of tasks within the environments, continuously updating and outputting the task completion metrics throughout the workflow.

CRAB Agents and Environments

Task: *Open `slack`, navigate to `multi-modal-benchmark` channel, summarize the last two messages, and then send a message to the summary to the first contact on the phone*



Crab: Cross-environment Agent Benchmark for Multimodal Language Model Agents

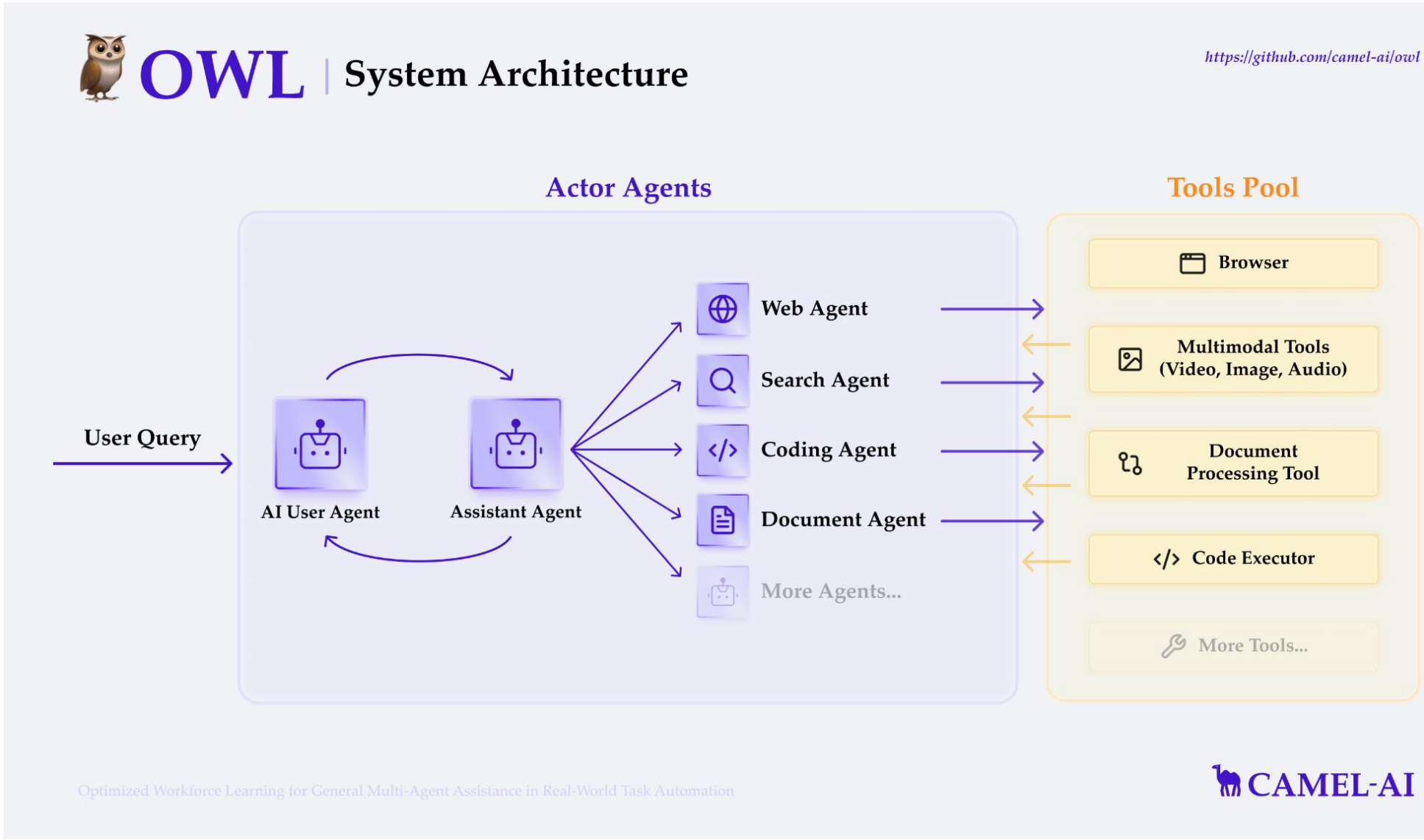
CRAB Agents and Environments

Crab benchmark system mainly consists of five types of component:

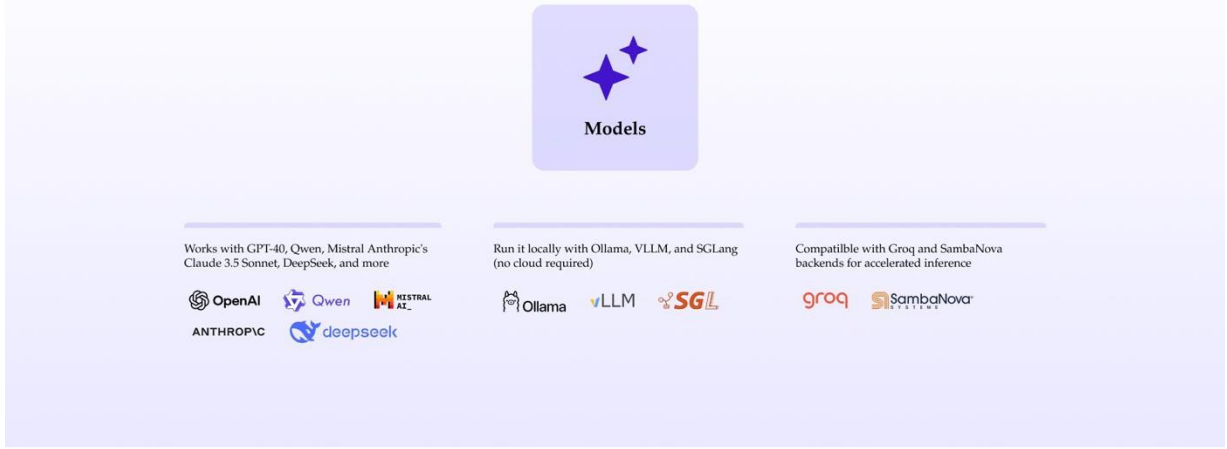
- `Action`: The fundamental building block of Crab framework, which represents a unit operation that can be taken by an agent or as a fixed process that called multi times in a benchmark.
- `Evaluator`: A specific type of `Action` that assess whether an agent has achieved its goal. Multiple evaluators can be combined together as a graph to enable complex evaluation.
- `Environment` A abstraction of an environment that the agent can take action and obverse in a given action and observation space. An environment can be launched on the local machine, a physical remote machine, or a virtual machine.
- `Task`: A task with a natural language description to instruct the agent to perform. It can include interaction with multiple environments. Notice that in the benchmark, a task should have an graph evaluator to judge if the task progress.
- `Benchmark`: The main body of the crab system that contains all required component to build a benchmark, including environments, tasks, prompting method. It controls several

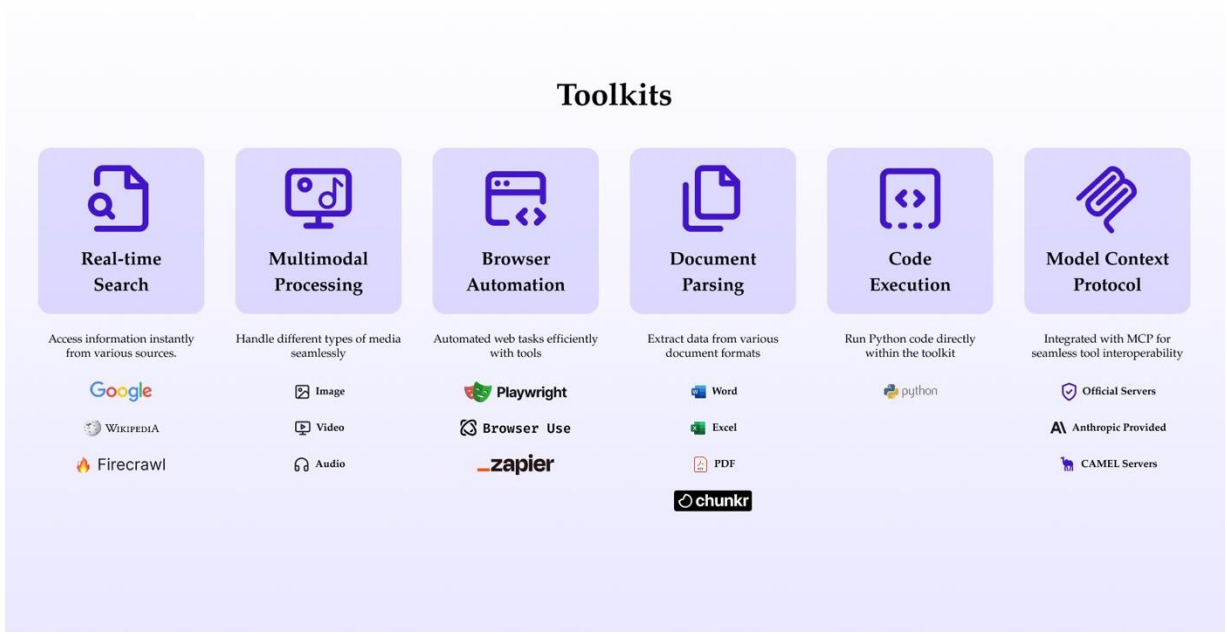
Crab: Cross-environment Agent Benchmark for Multimodal Language Model Agents

OWL Agents


OWL | System Architecture
https://github.com/camel-ai/owl
Actor Agents
Tools Pool
User Query
AI User Agent
Assistant Agent
Web Agent
Search Agent
Coding Agent
Document Agent
More Agents...
Browser
Multimodal Tools (Video, Image, Audio)
Document Processing Tool
Code Executor
More Tools...
Optimized Workforce Learning for General Multi-Agent Assistance in Real-World Task Automation
CAMEL-AI

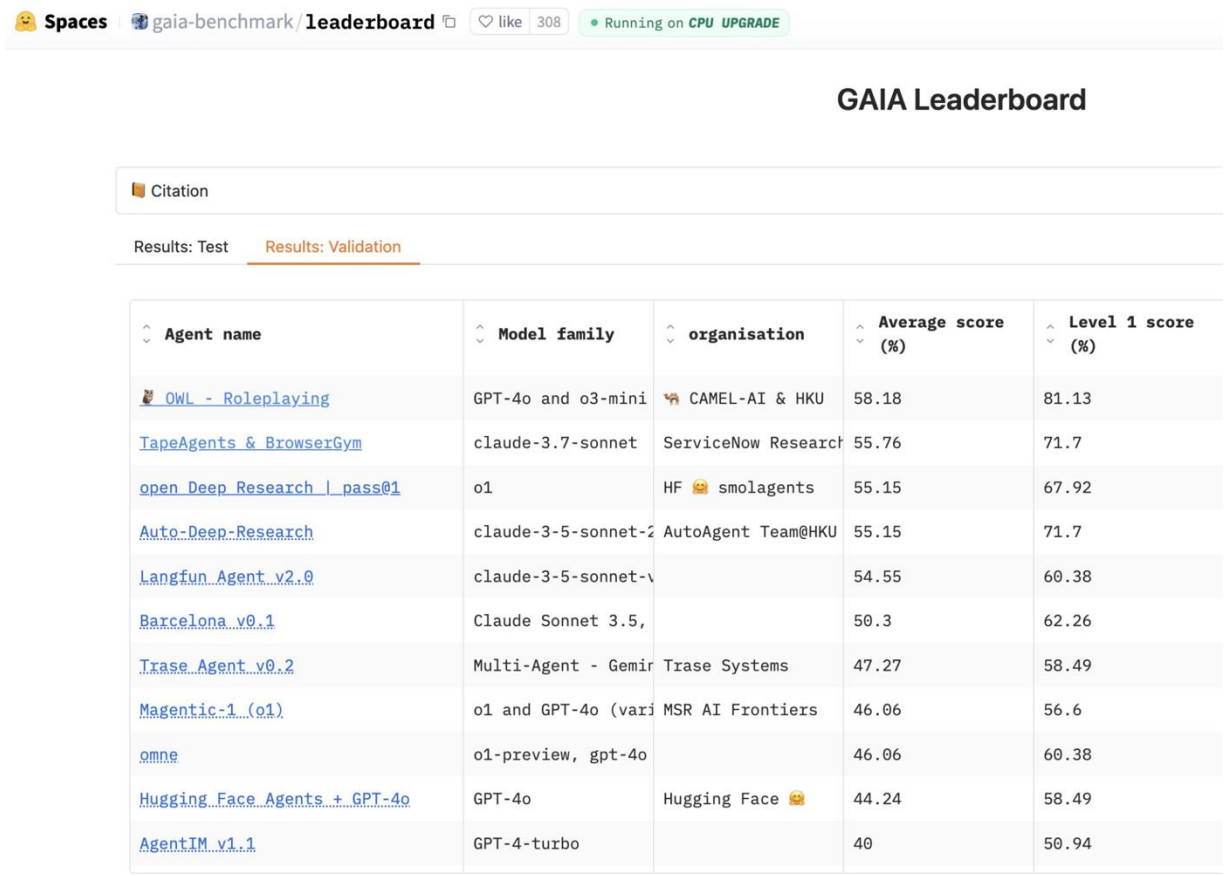
OWL Agents


Models
Works with GPT-40, Qwen, Mistral Anthropic's Claude 3.5 Sonnet, DeepSeek, and more
OpenAI
Qwen
MISTRAL AI_
ANTHROP\C
deepseek
Run it locally with Ollama, VLLM, and SGLang (no cloud required)
Ollama
vLLM
SGL
Compatilble with Groq and SambaNova backends for accelerated inference
groq
SambaNova


Toolkits
Real-time Search
Access information instantly from various sources.
Google
Wikipedia
Firecrawl
Multimodal Processing
Handle different types of media seamlessly
Image
Video
Audio
Browser Automation
Automated web tasks efficiently with tools
Playwright
Browser Use
zapier
Document Parsing
Extract data from various document formats
Word
Excel
PDF
chunkr
Code Execution
Run Python code directly within the toolkit
python
Model Context Protocol
Integrated with MCP for seamless tool interoperability
Official Servers
Anthropic Provided
CAMEL Servers

OWL Agents

- 58.18% average score on GAIA benchmark
- Ranking #1 among open-source frameworks! 🏆
- Updated results with Glaude 3.7 achieve 69.09%



Spaces gaia-benchmark/leaderboard like 308 Running on CPU UPGRADE

GAIA Leaderboard

Citation

Results: Test | Results: Validation

Agent name	Model family	organisation	Average score (%)	Level 1 score (%)
OWL - Roleplaying	GPT-4o and o3-mini	CAMEL-AI & HKU	58.18	81.13
TapeAgents & BrowserGym	claude-3.7-sonnet	ServiceNow Research	55.76	71.7
open Deep Research | pass@1	o1	HF smolagents	55.15	67.92
Auto-Deep-Research	claude-3-5-sonnet-2	AutoAgent Team@HKU	55.15	71.7
Langfun Agent v2.0	claude-3-5-sonnet-v		54.55	60.38
Barcelona v0.1	Claude Sonnet 3.5,		50.3	62.26
Trase Agent v0.2	Multi-Agent - Gemir	Trase Systems	47.27	58.49
Magentic-1 (o1)	o1 and GPT-4o (vari	MSR AI Frontiers	46.06	56.6
omne	o1-preview, gpt-4o		46.06	60.38
Hugging Face Agents + GPT-4o	GPT-4o	Hugging Face	44.24	58.49
AgentIM v1.1	GPT-4-turbo		40	50.94

OWL Agents

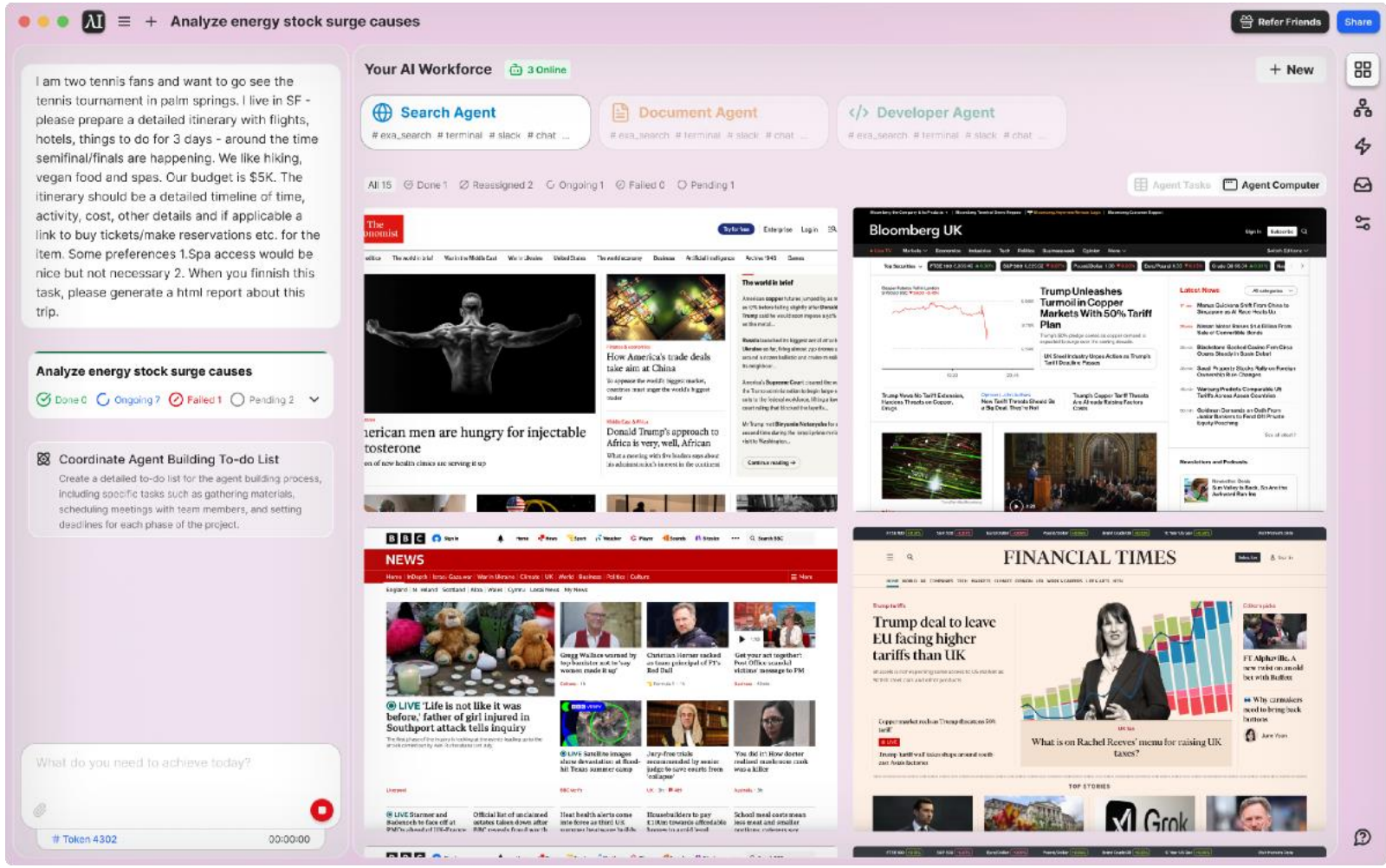
- RL on the Planner (Qwen-2.5-32B-Instruct)
- +4.85 with SFT
- +16.36 with DPO

Dataset	# Total	# SFT Filtering	# DPO Filtering	Agent Capabilities
HotpotQA	998	495	354	
WikiTableQuestions	869	577	311	
Math-related	1100	487	297	
Infinity-MM	499	40	47	
Total/Average	3466	1599	1009	

Planner	Level 1	Level 2	Level 3	Average
GPT-4o-mini [21]	64.15	45.34	19.23	47.27
Qwen2.5-72B-Instruct [44]	60.30	51.16	19.23	49.09
Claude-3.7-Sonnet [1]	81.13	53.49	34.61	59.39
GPT-4o [21]	81.14	58.14	26.92	60.61
Qwen2.5-32B-Instruct [44]	49.05	33.72	19.23	36.36
w. OWL (SFT)	56.60 +7.55	39.53 +5.81	15.38 -3.85	41.21 +4.85
w. OWL (SFT + DPO)	67.92 +18.87	51.16 +17.44	26.92 +7.69	52.73 +16.37

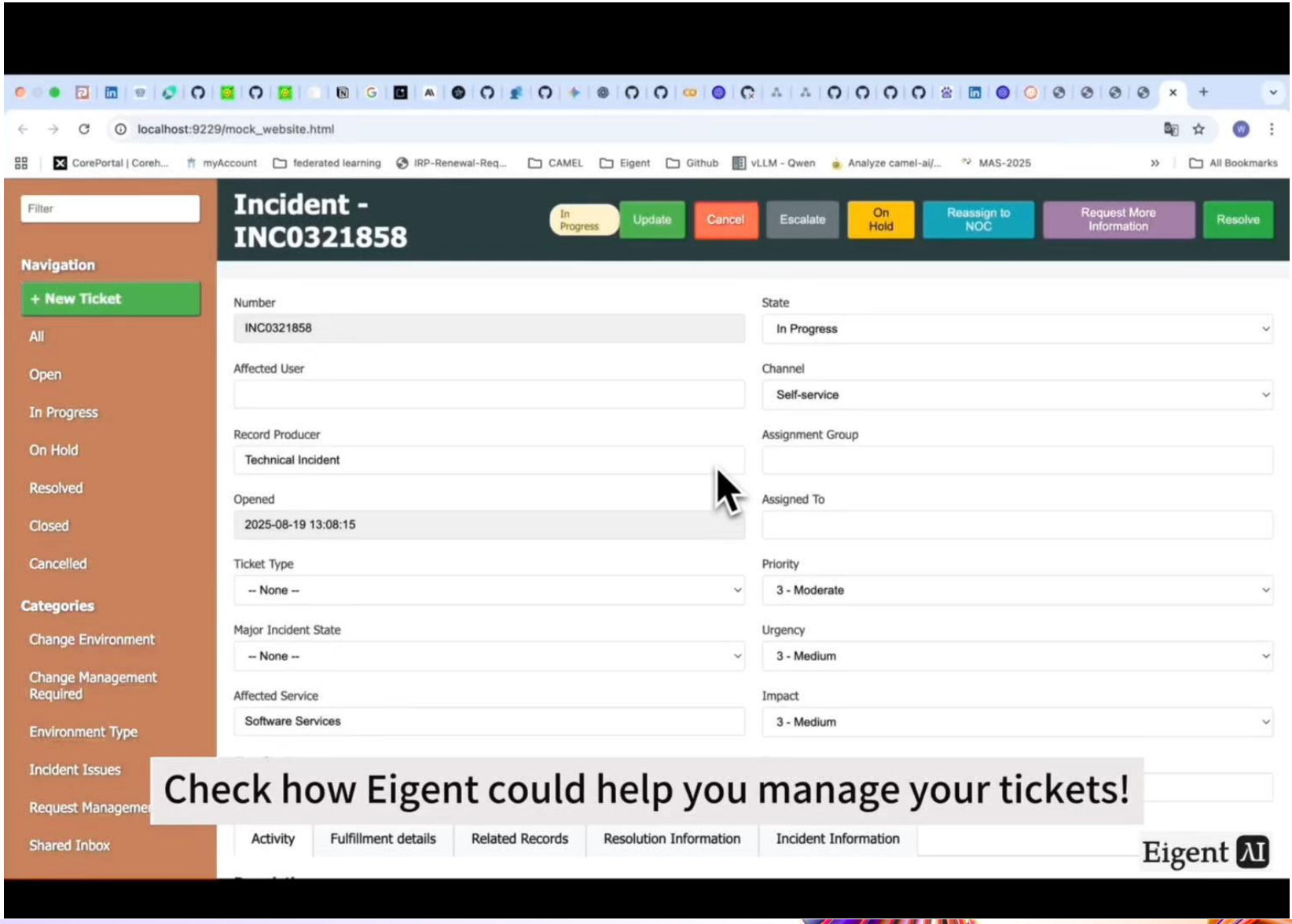
Our Solution: Eigent - The Multi-agent Workforce Desktop

Eigent is a multi-agent workforce desktop that can automaticly complete complex tasks **in parallel**.
Human-like Operation: It intelligently controls your computer by manipulating browsers, using the terminal, and executing self-generated code.
Custom Enterprise Tools: Equip your workforce with tools tailored to any business scenario.
Open Architecture: Our open architecture ensures you can always integrate the latest and most powerful AI models and tools.



Use Cases

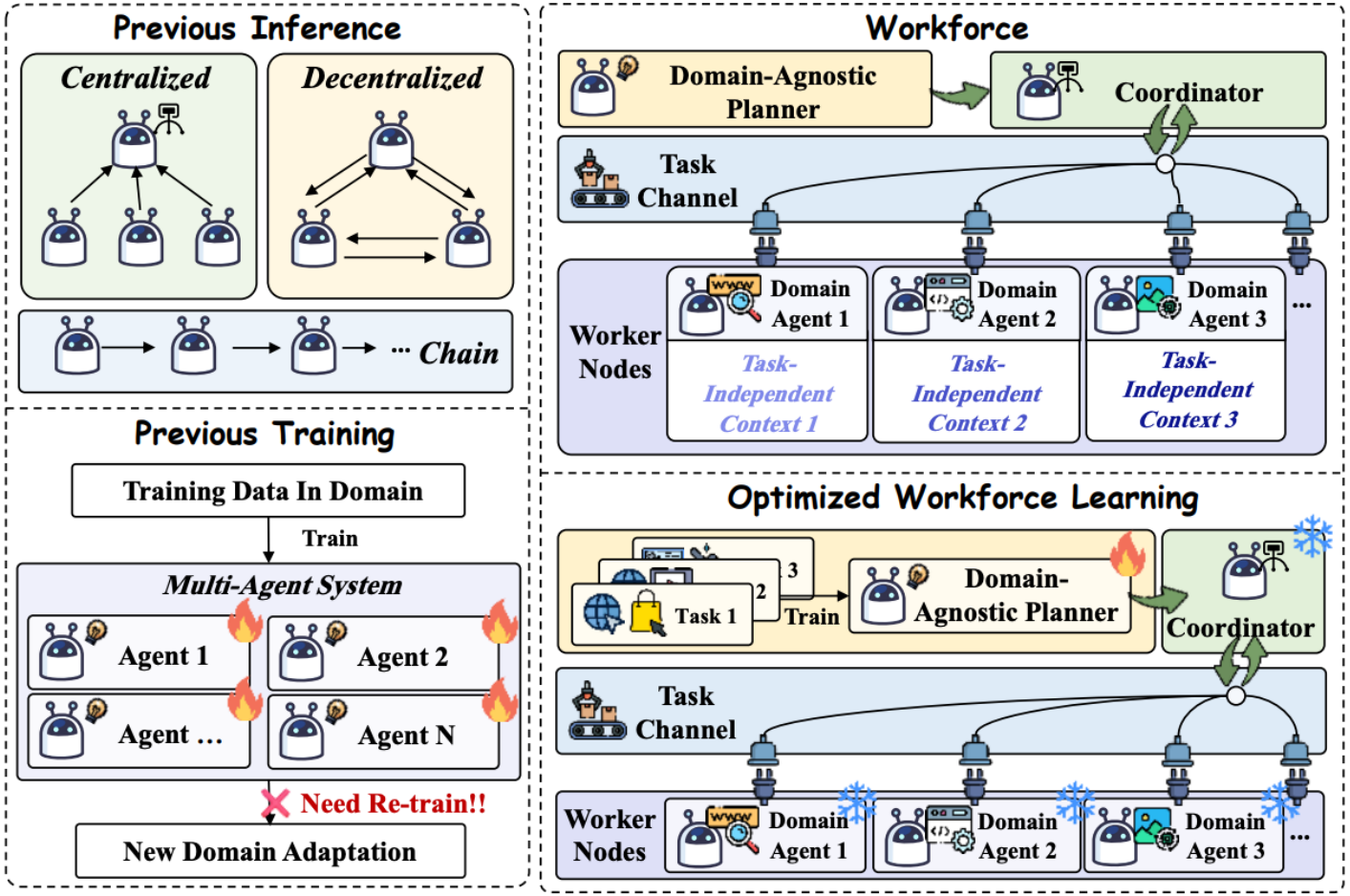
https://www.eigent.ai/usecases/ticket_management_system_integration_and_reporting



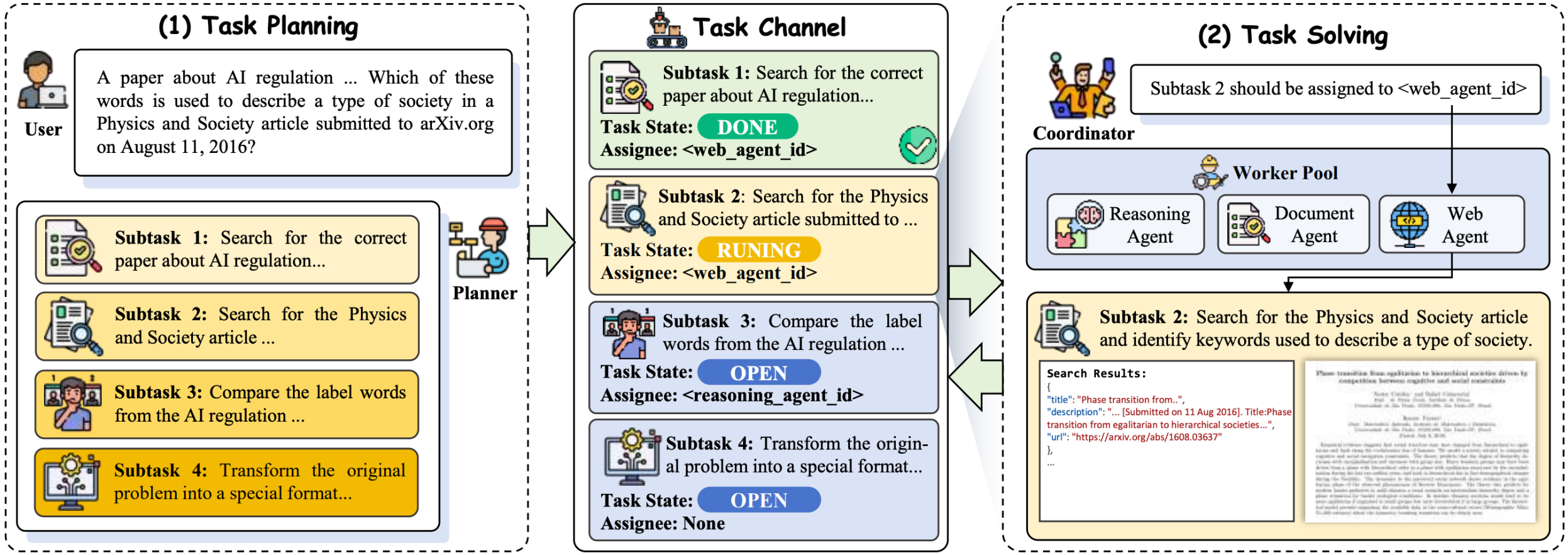
How to build a fowerful Workforce System

A Centralized Multi-agent System

- Domain-agnostic Planner: Generates abstract task decompositions based on high-level goals.
- Coordinator: Assigns subtasks to appropriate workers.
- Domain-specific Worker Nodes: A set of specialized agents that perform tool calls to accomplish each subtasks.

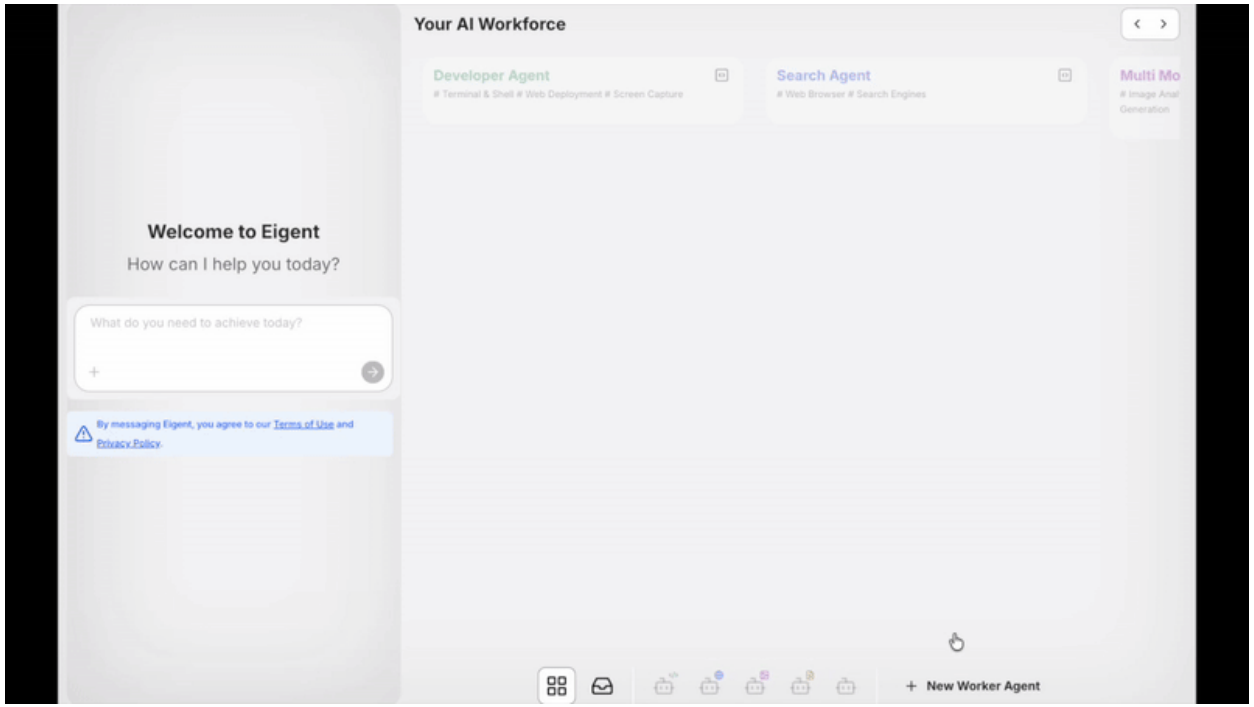


How to build a fowerful Workforce System



How to build a fowerful Workforce System

- Directly add a new agent: Fast, convenient and effective




```python
# create workforce instance before adding workers
workforce = Workforce(
    'A workforce',
    graceful_shutdown_timeout=30.0,  # 30 seconds for debugging
    share_memory=False,
    coordinator_agent=coordinator_agent,
    task_agent=task_agent,
    new_worker_agent=new_worker_agent,
    use_structured_output_handler=False,
    task_timeout_seconds=900.0,
)

workforce.add_single_agent_worker(
    "Search Agent: An expert web researcher that can browse websites, "
    "perform searches, and extract information to support other agents.",
    worker=search_agent,
).add_single_agent_worker(
    "Developer Agent: A master-level coding assistant with a powerful "
    "terminal. It can write and execute code, manage files, automate "
    "desktop tasks, and deploy web applications to solve complex "
    "technical challenges.",
    worker=developer_agent,
).add_single_agent_worker(
    "Document Agent: A document processing assistant skilled in creating "
    "and modifying a wide range of file formats. It can generate "
    "text-based files (Markdown, JSON, YAML, HTML), office documents "
    "(Word, PDF), presentations (PowerPoint), and data files "
    "(Excel, CSV).",
    worker=document_agent,
).add_single_agent_worker(
    "Multi-Modal Agent: A specialist in media processing. It can "
    "analyze images and audio, transcribe speech, download videos, and "
    "generate new images from text prompts.",
```

How to build a fowerful Workforce System

Optimized Workforce Learning (OWL)

- Two-Phase Training Strategy for Generalist Planning
- → Supervised Fine-Tuning + DPO Reinforcement Learning
- SFT: Learn valid task decomposition
- DPO: Learn better strategic preferences
- → Stronger planning behavior across domains

How to build a fowerful Workforce System

Phase 1: Supervised Fine-Tuning (SFT)

- Goal: Teach Planner to perform correct task decomposition

How It Works

- Collect expert trajectories using a strong model
- Planner outputs: subtasks + worker assignment + dependencies
- Workers output: execution traces (web search, code, tables, multimodal, etc.)
- Train Planner to imitate these expert examples

Outcome

- ✅ **Correct execution pipelines**
- ✅ **Calls the right Worker for each step**

How to build a fowerful Workforce System

Phase 2: Reinforcement Learning (DPO)

- Goal: Improve decision quality using preference learning — more robust, fewer mistakes

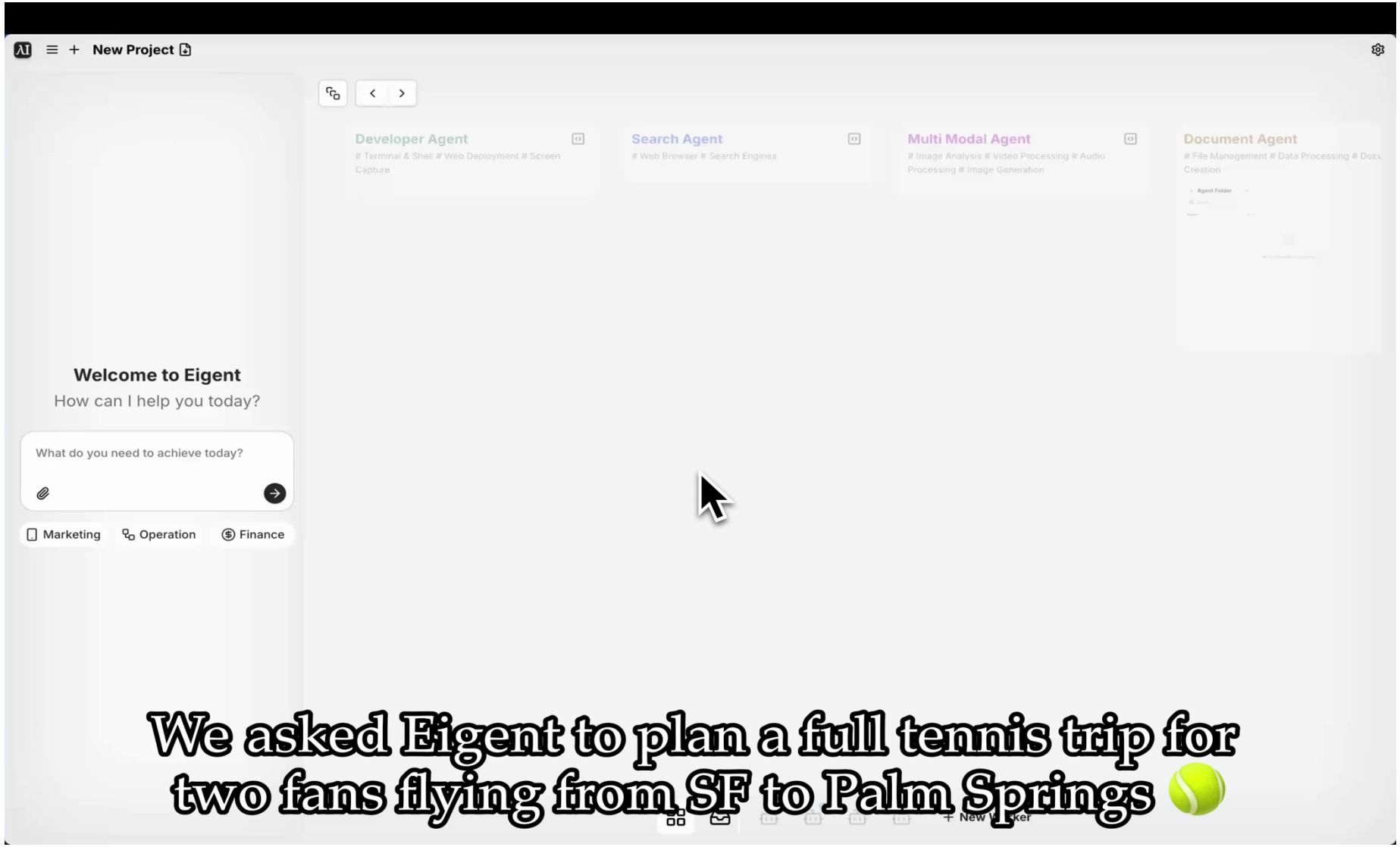
Training Procedure

- Roll out 4 diverse trajectories per task using SFT-initialized Planner
- Offline evaluation:
  - QA/Math/Table ✅ correct → chosen
  - Multimodal ✅ cosine similarity > 0.7 → chosen
- Construct paired preferences: chosen vs rejected
- Use Direct Preference Optimization to update Planner

Key Benefit

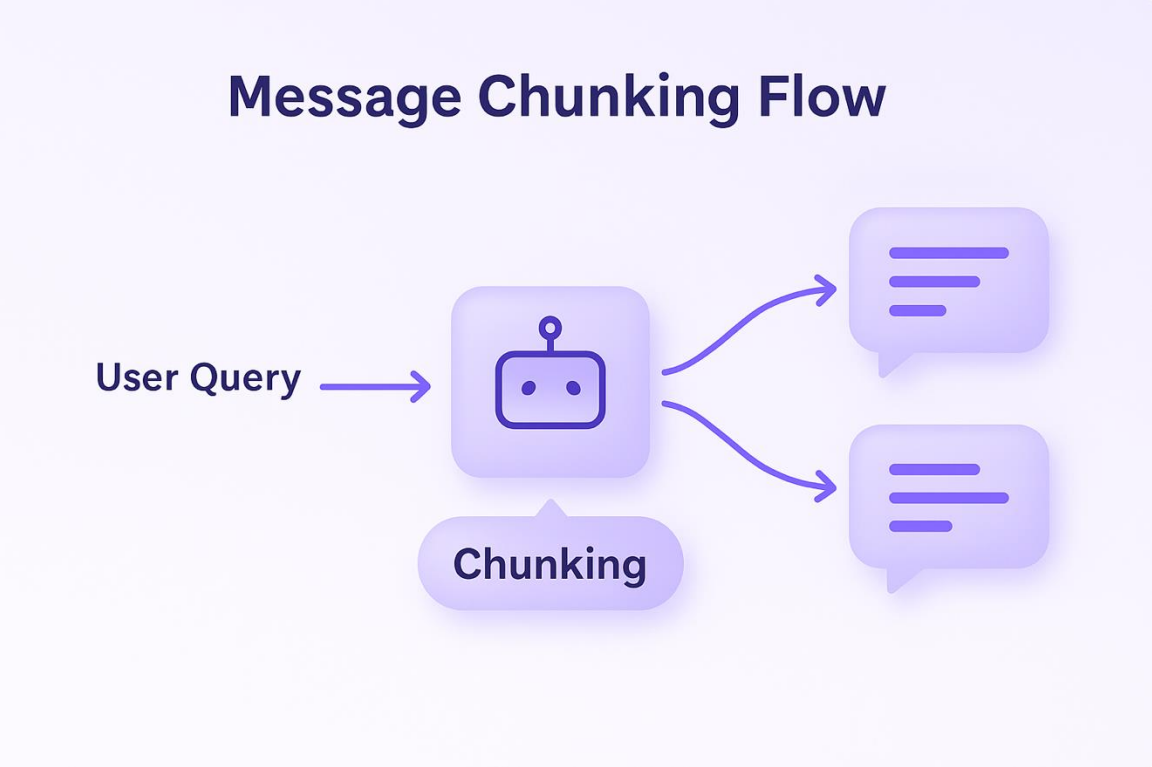
  - Planner prefers strategies with higher success probability

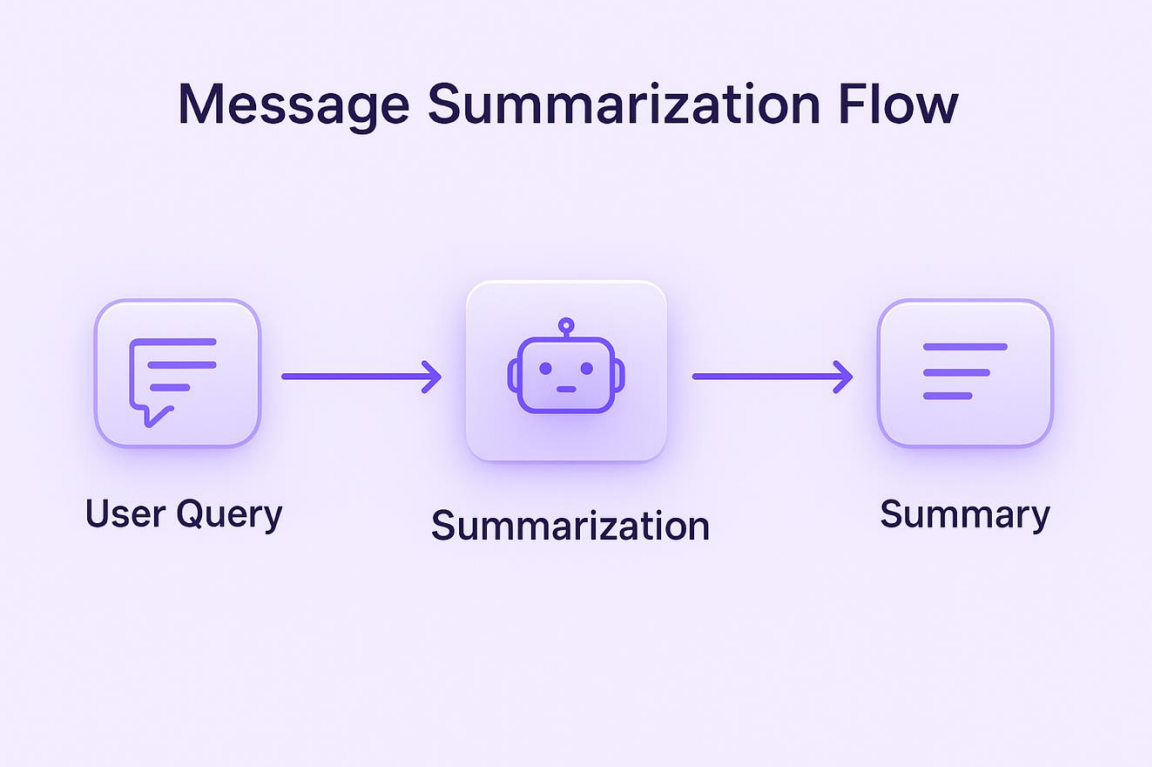
How to build a fowerful Workforce System



How to build a fowerful Workforce System

Context Engineering

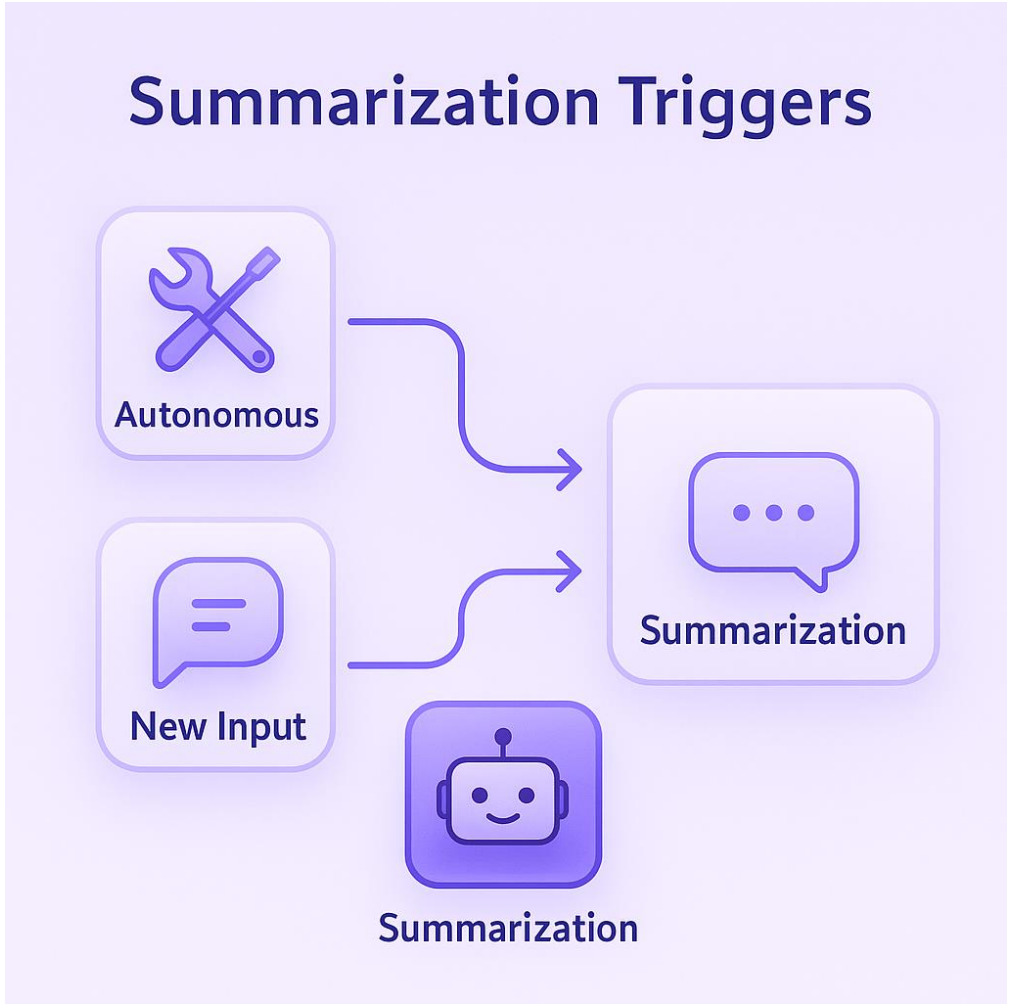




How to build a fowerful Workforce System

Context Engineering

```
template = textwrap.dedent(
    """\
    Summarize the conversation below.
    Produce markdown that strictly follows this outline and numbering:

    Summary:
    1. **Primary Request and Intent**:
    2. **Key Concepts**:
    3. **Errors and Fixes**:
    4. **Problem Solving**:
    5. **Pending Tasks**:
    6. **Current Work**:
    7. **Optional Next Step**:

    Requirements:
    - Use bullet lists under each section (`- item`). If a section has no
      information, output `- None noted`.
    - Keep the ordering, headings, and formatting as written above.
    - Focus on concrete actions, findings, and decisions.
    - Do not invent details that are not supported by the conversation.

    Conversation:
    {conversation_text}
    """
)
```



How to build a fowerful Workforce System

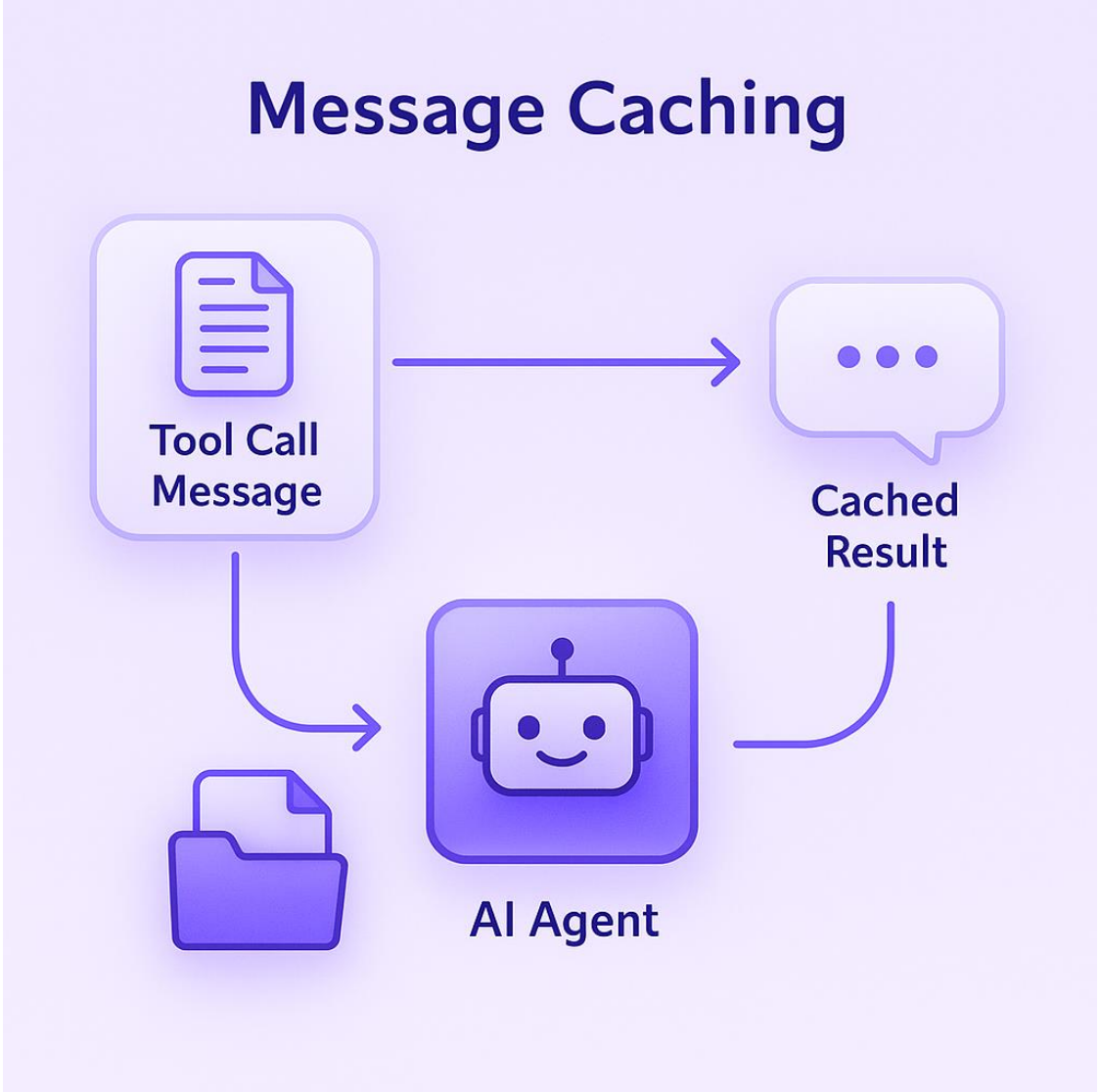
Simply using a prompt is not enough.

- The information compression resulting from summarizing is unavoidable.
- We need to use code to help the agent obtain as much information as possible.
- Guide the agent to continue working

How to build a fowerful Workforce System

Context engineering at Toolkit Level

- Only retain one overview message
- The original tool call information is stored in the file system.
- The agent can obtain the raw information using grep.

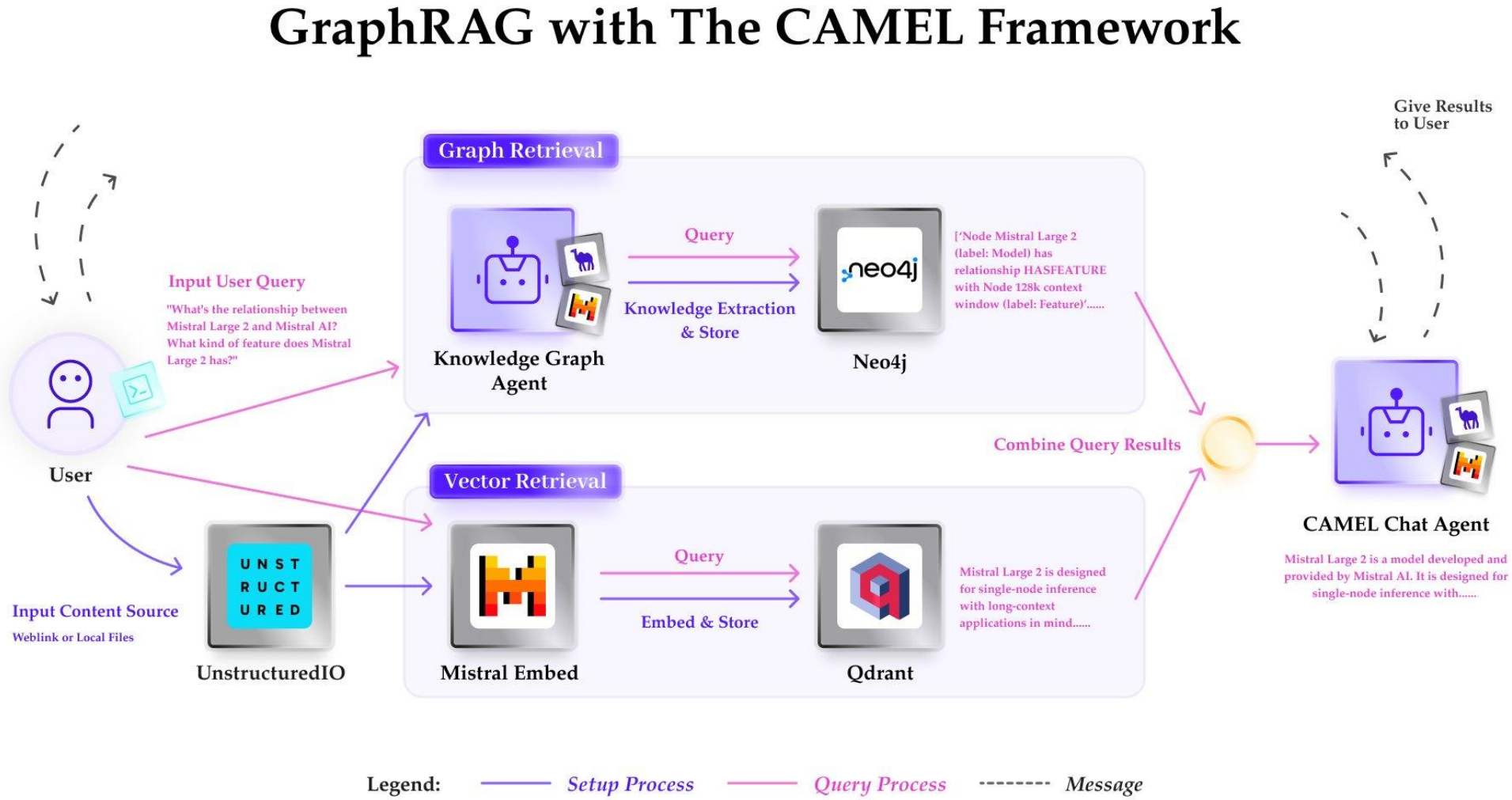


Building the Future of Multi-agent Workforce

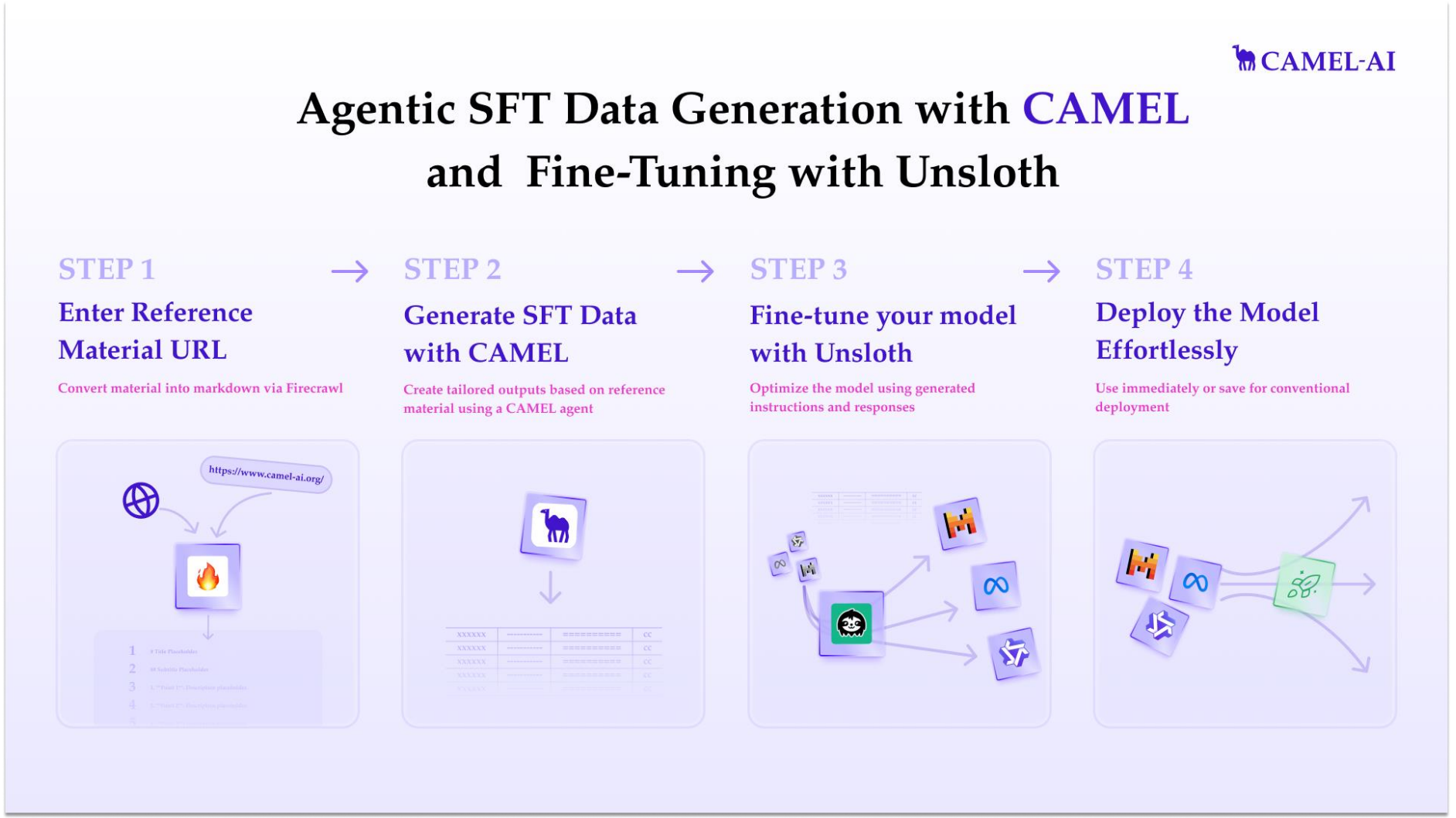
Training ->
Ability of Evolution?

Scaling Laws of Multi-agent Systems?

Graph + Vector RAG



Agentic Data Generation



https://docs.camel-ai.org/cookbooks/sft_data_generation_and_unsloth_finetuning_tinyllama.html

Workforce workflow memory

```python
async def demonstrate_second_session():
    workforce = Workforce("Simple Demo Team - Session 2")
    # Add workers with same descriptive names as before
    math_agent = create_math_agent()
    workforce.add_single_agent_worker(
        description="math_expert",  # Same description = loads matching
        # workflow
        worker=math_agent,
        enable_workflow_memory=False,  # Not saving in this session
    )

    writer_agent = create_writer_agent()
    workforce.add_single_agent_worker(
        description="content_writer",  # Same description = loads
        # matching workflow
        worker=writer_agent,
        enable_workflow_memory=False,  # Not saving in this session
    )

    # Load previous workflows
    loaded_workflows = workforce.load_workflow_memories()

    # Process new tasks with loaded workflow context
    new_tasks = [
        Task(
            content="Calculate the area of a circle with radius 7.5 meters",
            id="new_math_task",
        ),
        Task(
            content="Write a brief technical explanation of machine "
            "learning for beginners",
            id="new_writing_task",
        ),
    ]

    for task in new_tasks:
        try:
            await workforce.process_task_async(task)
        except Exception as e:
            logger.warning(f"Failed to process task {task.id}: {e}")

    return loaded_workflows
```

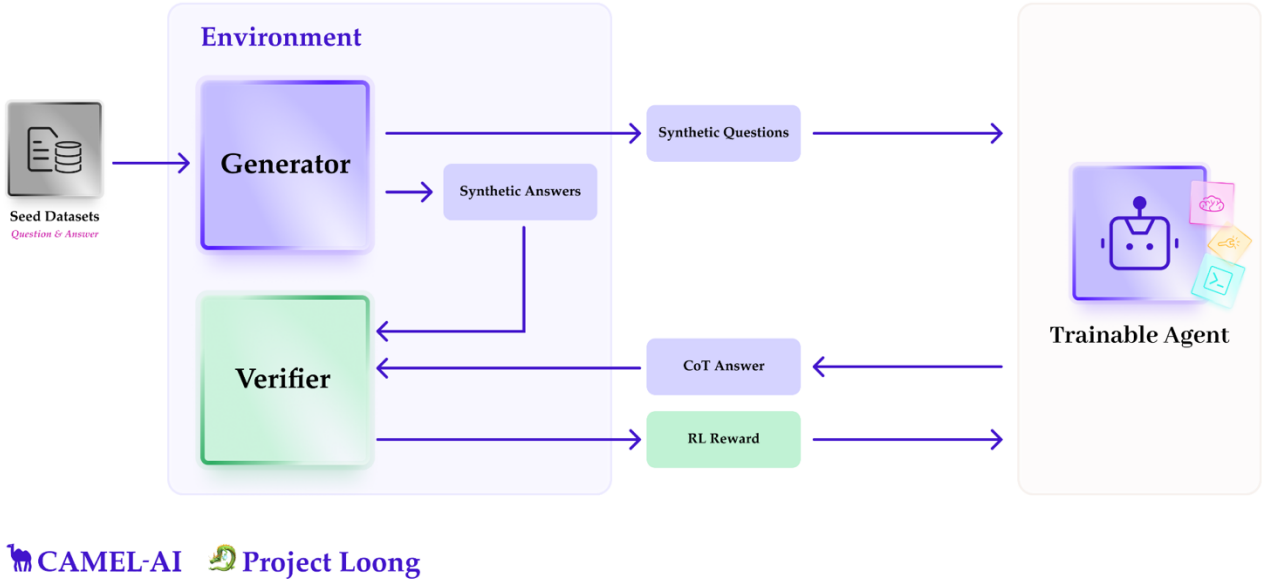
```
### Tools
[Bullet point list of tools used]

### Steps
1. Identify the required formula: A = P*(1 + r)^n.
2. Substitute given values into the formula: P = 1000, r = 0.05, n = 3.
3. Compute the sum inside parentheses: calculate 1 + r = 1.05.
4. Compute the exponentiation: calculate 1.05^3 = 1.157625.
5. Multiply by principal: compute A = 1000 * 1.157625 = 1157.625 (unrounded).
6. Compute total interest unrounded: Interest = A - P =
   1157.625 - 1000 = 157.625.
7. Round results to 2 decimals and format as USD: A → $1157.63;
   Interest → $157.63.
8. Return deliverables: formula, each arithmetic step with numbers substituted,
   unrounded A, rounded A in USD, and rounded interest in USD.

### Failure And Recovery Strategies
(No failure and recovery strategies recorded)

### Notes And Observations
No errors encountered.
</MARKDOWN CONTENT>

System message of math agent after loading workflow:
You are a math expert specialized in solving mathematical problems.
You can perform calculations, solve equations, and work with various
mathematical concepts.
Use the math tools available to you.

--- Workflow Memory ---
The following is the context from a previous session or workflow which might be
useful for to the current task. This information might help you understand the
background, choose which tools to use, and plan your next steps.

## WorkflowSummary
[The Workflow Summary from the above]
```

Project Loong:

Synthetic Data at Scale with Verifiers

- Advanced Math: 1,611 questions
- Advanced Physics: 429 questions
- Computational Biology: 51 questions
- Finance: 235 questions
- Game: 926 questions
- Graph & Discrete Math: 178 questions
- Logic: 130 questions
- Mathematical Programming: 76 questions
- Medicine: 916 questions
- Security & Safety: 516 questions
- Programming: 585 questions

Agent-Environment Loop



Thank You

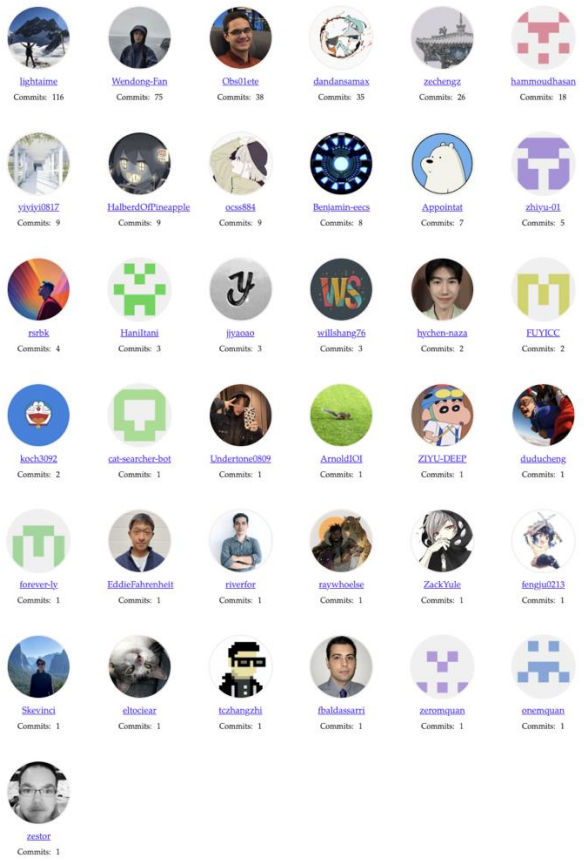
CAMEL-AI.org

An open-source research organization

CAMEL-AI

Eigent AI

All contributors



Join us in finding the scaling law of *Agents*!

科技生态圈峰会 + 深度研习

——1000 + 技术团队的共同选择

KEYLINK ing

AiDD 峰会

AiDD峰会 上海站
AI+研发数字峰会
时间：2026.05.22-23

AiDD峰会 北京站
AI+研发数字峰会
时间：2026.08.21-22

AiDD峰会 深圳站
AI+研发数字峰会
时间：2026.11.20-21

AiDD峰会详情

AiDD × AI+产品峰会

AiDD × 上海站
AI+产品创新峰会
时间：2026.01.16-17

AiDD × 北京站
AI+产品创新峰会
时间：2026.08.23-24

产品峰会详情

AiDD × K

2026 AI+ PRODUCT INNOVATION SUMMIT

01.16-17 · ShangHai

AI+产品创新峰会

Track 1: AI 产品战略与创新设计

从0到1的AI原生产品构建

论坛1: AI时代的用户洞家与需求发现

论坛2: AI原生产品战路与商业模式重构

论坛3: AgenticAI产品创新与交互设计

Track 2: AI 产品开发与工程实践

从1到10的工程化落地实践

论坛1: 面向Agent智能体的产品开发

论坛2: 具身智能与AI硬件产品

论坛3: AI产品出海与本地化开发

Track 3: AI 产品运营与智能演化

从10到100的AI产品运营

论坛1: AI赋能产品运营与增长黑客

论坛2: AI产品的数据飞轮与智能演化

论坛3: 行业爆款AI产品案例拆解

2-hour Speech: 回归本质

用户洞察的第一性

——2小时思维与方法论工作坊

在数字爆炸、AI迅速发展的时代，仍然考验“看见”的“同理心”

Panel 1: 出海前瞻

“出海避坑地图”圆桌对话

——不止于翻译:

AI时代的出海新范式

Panel 2: 失败复盘

为什么很多AI产品“叫好不叫座”?

——从伪需求到真价值:

AI产品商业化落地的关键挑战

智能重构产品 数据驱动增长

Reinventing Products with Intelligence, Driven by Data

80+ 业界大咖 汇聚产品大咖的真知灼见

40+ 创新案例 开拓全新AI+产品视角

10+ 分论坛 涵盖产品到商业增长的全链路

800+ 听众规模 共同探索产品的智能重构

扫码查看会议详情

第8届AI+研发数字峰会
AI+ Development Digital Summit

感谢聆听！

扫码领取会议PPT资料