

LLM Agents

Super AI Engineer (Season 5)

Ye Kyaw Thu
Lab Leader, LU Lab., Myanmar
Visiting Professor, NECTEC, Thailand

24 June 2025 (Tue)

Table of Content

- Testing LLM Intelligence
- Why Agents are Essential for LLMs
- The Evolution of Software Development
- Diverse Applications of LLM Agents
- LLM Agents Basics and R&Ds
- Demo
- Don't Build Multi-Agents
- Summary

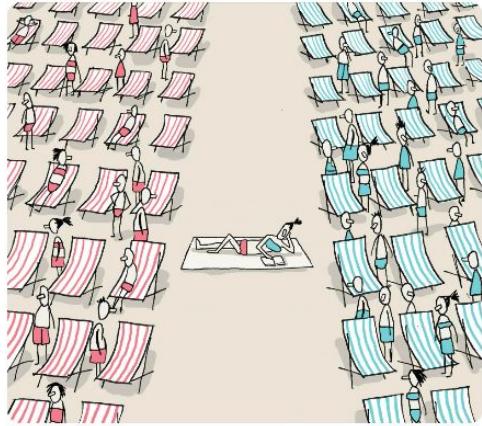
Testing LLM Intelligence

To assess the intelligence of large language models (LLMs), I conducted experiments focusing on two key dimensions:

1. Multimodal reasoning testing: ChatGPT-4o's ability to interpret and explain the meaning behind a cartoon image, and
2. Cross-linguistic comprehension evaluating: Gemini 1.5 Pro's understanding of Myanmar-language proverbs by asking it to explain them in both English and Myanmar.

These scenarios examine how well LLMs handle visual abstraction and cultural-linguistic nuance, revealing strengths and limitations in their intelligence.

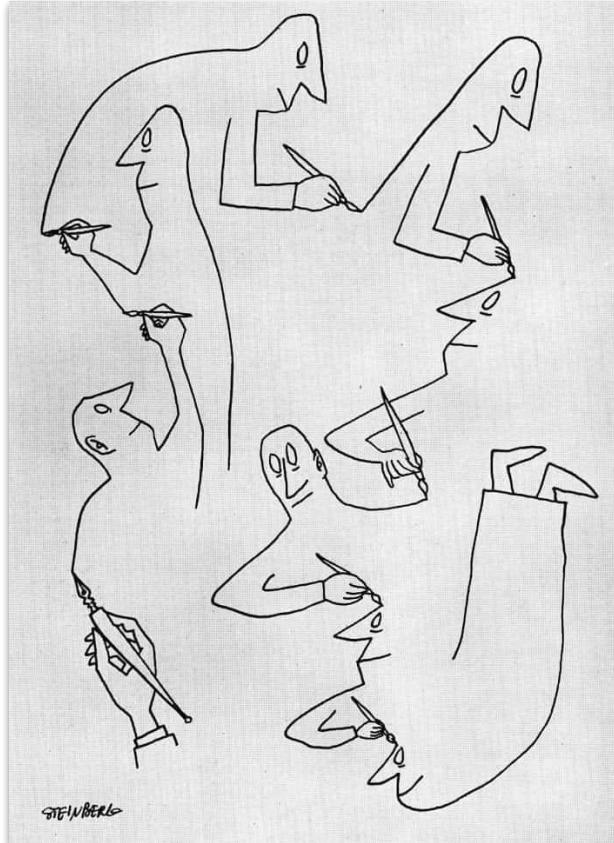
Testing LLM Intelligence(ChatGPT-4o with Cartoon Images)



I am attaching a new cartoon image for you, ChatGPT.
Can you explain to me the meaning or message of this
cartoon image.

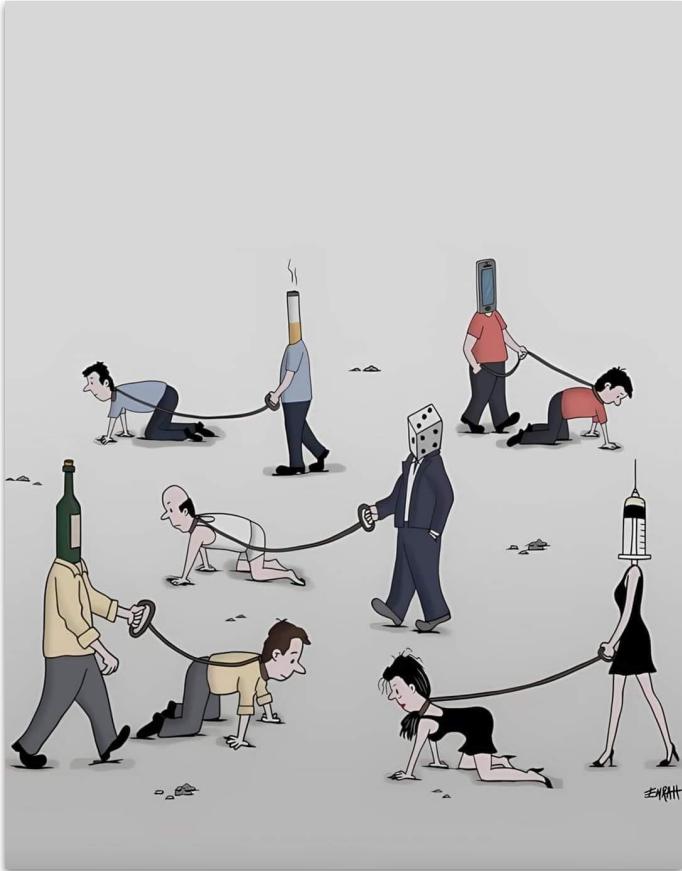
- In summary, the cartoon illustrates the idea of maintaining individuality and neutrality in a world that is often divided into opposing factions. It encourages the viewer to consider the benefits of not strictly aligning with one side and finding peace in a balanced or neutral stance.

Testing LLM Intelligence(ChatGPT-4o with Cartoon Images)



- The main message of the cartoon is about the interconnectedness of human actions and creativity. It suggests that everything we do is part of a larger, continuous cycle of influence and creation.
- It emphasizes the idea that our contributions are part of a collective process, where we both shape and are shaped by the contributions of others.

Testing LLM Intelligence(ChatGPT-4o with Cartoon Images)



- The cartoon portrays how addictions—whether to alcohol, nicotine, drugs, gambling, or technology—can dominate individuals' lives, reducing them to a subservient and degraded state. The people being led by these vices are depicted as powerless, crawling on all fours, illustrating how addictions can strip away personal agency, dignity, and freedom. The overall message highlights the destructive nature of these vices and the way they can enslave individuals, leading them through life rather than allowing them to walk freely on their own terms.

Testing LLM Intelligence(ChatGPT-4o with Cartoon Images)



- In the first panel, a woman walking ahead casually drops her handkerchief behind her, which appears to be an intentional act.
- A man walking behind her notices the dropped handkerchief, quickly picks it up, and chases after her in the second panel.
- In the third panel, the man is eagerly running towards the woman with the handkerchief in hand, ready to return it to her.

Testing LLM Intelligence(ChatGPT-4o with Cartoon Images)



- The cartoon humorously critiques how official or bureaucratic language is often filled with jargon, legal terms, or complex phrases that can be confusing to the general public. It suggests that even people working within the system might not fully understand the language used, leading to miscommunication and misunderstanding. The cartoon may be a satire on how excessive complexity in language can be a barrier to clear communication and transparency.

Testing LLM Intelligence (Gemini 2.5 Flash with Myanmar Proverbs)

Can you explain following Myanmar proverbs?

1. ကိုယ့်ငါးချဉ် ကိုယ်ချဉ်
2. ကောက်ညှင်းက မစေး၊ ဆန်ကြမ်းက စေး
3. ခွေးလျေးခုန်၍ ဖုန်မထ
4. ကြွောက်တစ်လက်၊ ကြွောက်တစ်ခုန်
5. ဆင့်ပါးစပ် နှမ်းပက်

Both English and Myanmar explanation for each proverb
is better.

Testing LLM Intelligence (Gemini 2.5 Flash with Myanmar Proverbs)

1. ကိုယ့်ငါးချုပ် ကိုယ်ချုပ် (Ko Myar Chin Ko Chin)

English Explanation:

This proverb literally translates to "Your own pickled fish, you say it's sour." It means that people tend to praise or defend their own creations, possessions, or even their children, regardless of their actual quality or merit. It highlights a natural bias towards one's own.

Myanmar Explanation:

မိမိပိုင်ဆိုင်သော အရာဝတ္ထု၊ မိမိပြုလုပ်ထားသော အရာဝတ္ထု သို့မဟုတ် မိမိ၏ သားသမီးများကိုပင် မကောင်းသည့်
တိုင်အောင် ကောင်းသည့်ဟု ချီးမွမ်းပြောဆိုခြင်း၊ သေားလူများက မကောင်းဟု ပြောသည့်တိုင်အောင် ကောင်းသည့်
ဟု ထောက်ခံပြောဆိုခြင်းကို ဆိုလိုသည်။

Testing LLM Intelligence (Gemini 2.5 Flash with Myanmar Proverbs)

2. ကောက်ညှင်းက မစေး၊ ဆန့်ကြမ်းက စေး (Kauk Hnyin Ka Ma Say, San Kyam Ka Say)

English Explanation:

Literally, this means "Glutinous rice isn't sticky, coarse rice is sticky." This proverb describes a situation where something that is supposed to be effective or good turns out to be useless, while something unexpected or seemingly inferior proves to be useful or effective. It points to an ironic or counter-intuitive outcome.

Myanmar Explanation:

ဖြစ်သင့်သည်ဟု ထင်ထားသော အရာက မဖြစ်မြောက်ဘဲ၊ ဖြစ်သင့်သည်ဟု မထင်ထားသော အရာက ဖြစ်မြောက် အောင်မြင်သွားခြင်းမျိုးကို ရည်ညွှန်းသည်။ လူမှုဆက်ဆံရေးတွင်လည်း ခင်မင်ရင်းနှီးမှ မရှိသူက ကူညီဖေးမပြီး၊ ရင်းနှီးသူက ကူညီမှ မပေးသည့်အခါမျိုးတွင် သုံးသည်။

Testing LLM Intelligence (Gemini 2.5 Flash with Myanmar Proverbs)

3. ခွေးလျှေးခုန်၍ ဖုန်မထ (Khway Hlay Khun Yway Phon Ma Hta)

English Explanation:

This proverb translates to "A flea jumping doesn't raise dust." It means that an **insignificant action or a trivial person cannot cause any substantial impact or disturbance**. It emphasizes the lack of power or influence of a minor entity.

Myanmar Explanation:

အင်အားနည်းသောသူ သို့မဟုတ် အရေးမပါသောသူတစ်ဦး၏ လုပ်ဆောင်မှုသည် မည်သည့်သို့သောတင်ရားသော အကျိုးသက်ရောက်မှု သို့မဟုတ် ပြဿနာများ ဖြစ်ပေါ်စေခြင်းမရှိကြောင်းကို ဆိုလိုသည်။ အသေးအဖွဲ့ကိစ္စလေးများ က ကြီးမားသည့် အကျိုးသက်ရောက်မှုကို မပေးနိုင်သည့်အခါမျိုးတွင်လည်း သုံးသည်။

Testing LLM Intelligence (Gemini 2.5 Flash with Myanmar Proverbs)

4. ကြွေတစ်လက်၊ ကြက်တစ်ခုနှင့် (Kywe Tat Let, Kyat Ta Khun)

English Explanation:

This proverb translates to "One game of chance, one cockfight." It refers to risky or speculative ventures where one might either gain a lot or lose a lot in a single attempt. It suggests a high-stakes situation with unpredictable outcomes, often associated with gambling or bold decisions.

Myanmar Explanation:

ကံစမ်းမဲများ၊ လောင်းကစားများကဲ့သို့ တစ်ကြိမ်တည်းဖြင့် ကြီးမားသော အကျိုးအမြတ်ရနိုင်သလို၊ အားလုံးကို ဆုံးရှုံးသွားနိုင်သည့် အခြေအနေမျိုးကို ရည်ညွှန်းသည်။ ကံကိုသာ အားကိုပြီး အရှုံးအမြတ် ကြီးကြီးမားမား ဖြစ်နိုင်သည့် အခြေအနေများကို ဖော်ပြသည်။

Testing LLM Intelligence (Gemini 2.5 Flash with Myanmar Proverbs)

5. ဆင့်ပါးစပ် နှမ်းပက် (Sin Pa Zat Hnam Pat)

English Explanation:

Literally, this means "Sprinkling sesame seeds on an ant's mouth." This proverb describes doing something insignificant or offering something trivial to someone who is in a desperate or dire situation, where it will have no real impact or provide no meaningful help. It highlights the futility of a small gesture in the face of a huge problem.

Myanmar Explanation:

အလွန်လိုအပ်နေသောသူတစ်ဦးအား အနည်းငယ်မျှသော အကူအညီပေးခြင်းသည် မည်သည့်အကျိုးအမြတ်မျှ မဖြစ်ထွန်းစေဘဲ အချည်းနှီးဖြစ်ကြောင်းကို ဆိုလိုသည်။ များစွာလိုအပ်နေသူကို အနည်းငယ်မျှဖြင့် မည်သို့မျှ မကူညီနိုင်ခြင်းကို ပုံဆောင်သည်။

Why Agents are Essential for LLMs

- Hallucination & Factual Accuracy: LLMs can generate confident but incorrect information, especially when dealing with uncertain topics.
- Data Transparency & Freshness:
 - Lack of transparency regarding their training data.
 - Trained on static datasets, leading to outdated knowledge.
- Limited Interaction with the Outside World:
 - Cannot inherently interact with real-world systems or dynamic information (even with web search capabilities, deep interaction is limited).

Why Agents are Essential for LLMs

- Need for Context & Specialization:
 - Difficulty incorporating private company data or domain-specific knowledge.
- Agents bridge these gaps by enabling LLMs to:
 - Access external tools and databases.
 - Perform actions in the real world.
 - Incorporate up-to-date and proprietary information.
 - Reduce hallucinations by verifying facts.

The Evolution of Software Development

- The advent of Large Language Models is fundamentally reshaping how software is built, leading to new paradigms:
- **Software 1.0: Coding with Algorithms and Rules**
 - Traditional programming where developers explicitly write logic, algorithms, and rules.
 - Focus on deterministic outcomes based on predefined instructions.

The Evolution of Software Development (Software 1.0)

bash

1. [read-and-move.sh](#)
2. [change-filenames.sh](#)
3. [rm-date-sentences.sh](#)
4. [print-classID-prediction-result.sh](#)
5. [compare-img-or-pdf.sh](#)
6. [chk-sort-by-columns.sh](#)
7. [kill-all-detached.sh](#)
8. [unzip-all-with-one-passwd.sh](#)
9. [cut-filename.sh](#)
10. [calc-avg.sh](#)

perl

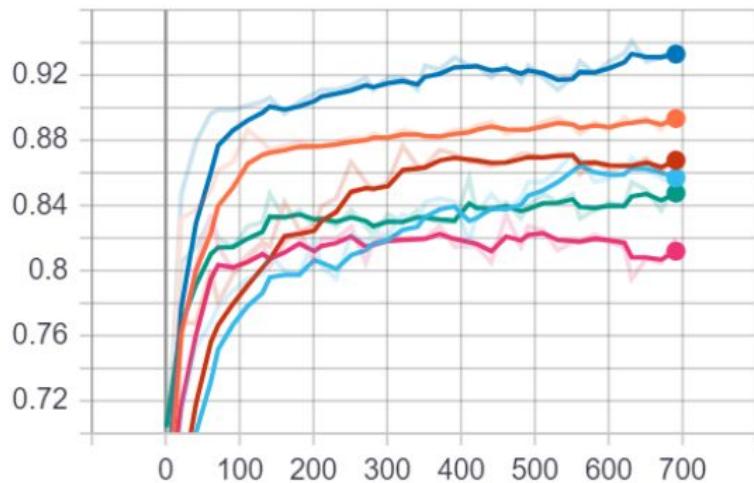
1. [clean-space.pl](#)
2. [rm-EnglishSentences.pl](#)
3. [word-analysis.pl](#)
4. [print-emojiSentences.pl](#)
5. [dq-multilines.pl](#)
6. [mk-abstract-para.pl](#)
7. [print-mySentenceOnly.pl](#)
8. [rm-symbol-and-myVowel-only-sentences.pl](#)
9. [rm-space-btw-numbers.pl](#)
10. [print-ngram.pl](#)

The Evolution of Software Development

- **Software 2.0: Adjusting Neural Network Weights**
 - Emergence of machine learning, where software's behavior is learned from data.
 - Developers primarily focus on data preparation, model architecture, and tuning parameters (e.g., weights in neural networks).
 - Heavily reliant on collaborative platforms like GitHub for code and model sharing.

The Evolution of Software Development (Software 2.0)

accuracy
tag: summary/accuracy



loss
tag: summary/loss

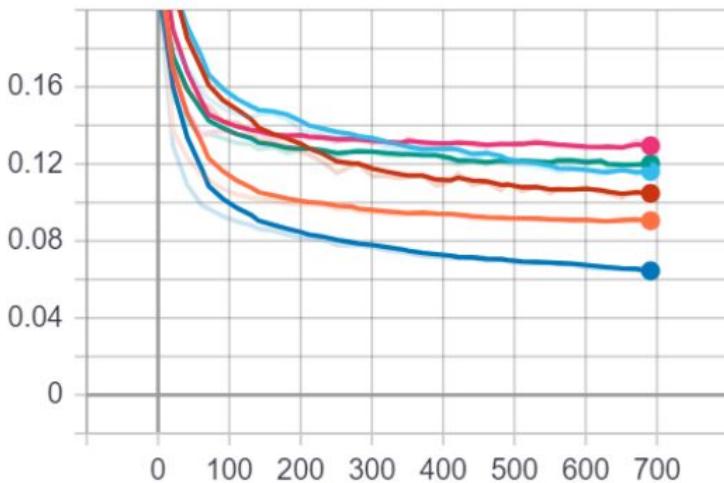


Fig.1 Accuracy and loss comparison graphs for three Siamese models (RNN-Siamese, CNN-Siamese and Transformer-Siamese) with myParaphrase corpus (version 1.0)

The Evolution of Software Development

- **Software 3.0: "Vibe Coding" or Coding with Prompting**
 - The era of LLMs, where high-level instructions (prompts) guide code generation and system behavior.
 - Developers shift towards crafting effective prompts, refining LLM outputs, and orchestrating agents.
 - Platforms like Hugging Face become central for sharing and leveraging pre-trained models and components.

Link of Vibe Coding Wiki: https://en.wikipedia.org/wiki/Vibe_coding

The Evolution of Software Development (Software 3.0)

ChatGPT ▾

Saved memory full ⓘ

Updated Code

python

Copy Edit

```
import argparse
import kenlm

# Function to evaluate the Language model
def evaluate_language_model(model_path, test_file, ngram_order, overall):
    try:
        # Load the KenLM model
        model = kenlm.LanguageModel(model_path)
        print(f"Loaded language model from: {model_path}")
        print(f"Model order: {model.order}")

        # Ensure the model order matches the user-specified n-gram
        if ngram_order and ngram_order != model.order:
            print(f"Warning: Model order {model.order} does not match user-specified n-gram order {ngram_order}.")

    except Exception as e:
        print(f"Error: {e}")

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Evaluate a language model using KenLM")
    parser.add_argument("model_path", type=str, help="Path to the KenLM language model file")
    parser.add_argument("test_file", type=str, help="Path to the test file for evaluation")
    parser.add_argument("--ngram_order", type=int, default=1, help="N-gram order for evaluation (default: 1)")
    parser.add_argument("--overall", action="store_true", help="Compute overall perplexity instead of sentence-level")

    args = parser.parse_args()
    evaluate_language_model(args.model_path, args.test_file, args.ngram_order, args.overall)
```

Ask anything

+ ⚙ Tools

0 ⓘ

- Prompt Engineering

GitHub Link:
<https://github.com/ye-kyaw-thu/tools>

Diverse Applications of LLM Agents

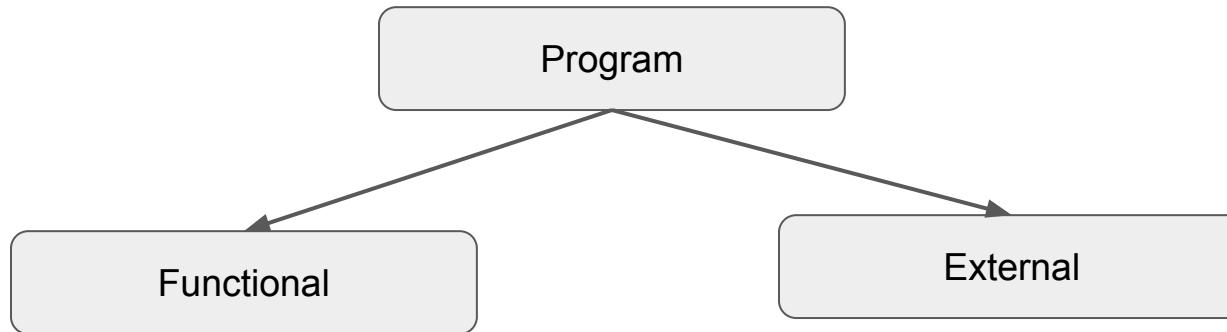
- LLM Agents are rapidly being adopted across various domains, extending the capabilities of large language models. Here are some key examples:
- Software Development Agents: Automating code generation, debugging, testing, and even managing entire development workflows.
- Web Browsing Agents: Navigating the internet, extracting information, summarizing content, and performing tasks requiring web interaction.
- Financial Analysis Agents: Analyzing market data, predicting trends, managing portfolios, and providing personalized financial advice.

Diverse Applications of LLM Agents

- Healthcare Agents: Assisting with diagnostics, personalizing treatment plans, analyzing medical literature, and managing patient care.
- E-commerce Agents: Enhancing customer support, personalizing shopping experiences, optimizing inventory, and managing logistics.

LLM Agents Basics and R&Ds (Tools for LLM Agents)

- An LM-used tool is a function interface to a computer program that runs external to the LM, where the LM generates the function calls and input arguments in order to use the tool.



Shumaker et al. Animal tool behavior: the use and manufacture of tools by animals. JHU Press, 2011

LLM Agents Basics and R&Ds (Tool Execution)

The screenshot shows a red header bar with the arXiv logo and a navigation menu. Below it, a white header bar indicates the category 'Computer Science > Computation and Language'. The main title 'ToolkenGPT: Augmenting Frozen Language Models with Massive Tools via Tool Embeddings' is displayed in large bold black font. Below the title, there are several interactive icons: 'CODE' (with a code icon), 'Notebook' (with a play icon), a copy icon, a bookmark icon, a bell icon, and a share icon. The authors' names 'Shibo Hao, Tianyang Liu, Zhen Wang, Zhiting Hu' are listed below the title. A text input field says 'Ask the author(s) a question! :)' and a blue 'Ask' button is next to it. At the bottom, a note states 'powered by CatalyzeX'. The main content area contains a detailed abstract about the paper's approach to augmenting LLMs with external tools.

[Submitted on 19 May 2023 (v1), last revised 15 Jan 2024 (this version, v4)]

ToolkenGPT: Augmenting Frozen Language Models with Massive Tools via Tool Embeddings

Shibo Hao, Tianyang Liu, Zhen Wang, Zhiting Hu

Ask the author(s) a question! :)

✉ Ask

powered by CatalyzeX

Augmenting large language models (LLMs) with external tools has emerged as a promising approach to solving complex problems. However, traditional methods, which finetune LLMs with tool demonstration data, can be both costly and restricted to a predefined set of tools. Recent in-context learning paradigm alleviates these issues, but the limited context length only allows for a few shots of demonstrations, leading to suboptimal understandings of the tools. Moreover, when there are numerous tools to choose from, in-context learning could completely fail to work. In this paper, we propose an alternative approach, **ToolkenGPT**, which combines the benefits of both sides. Our approach represents each *tool* as a *token* (*toolken*) and learns an embedding for it, enabling tool calls in the same way as generating a regular word token. Once a toolken is triggered, the LLM is prompted to complete arguments for the tool to execute. ToolkenGPT offers the flexibility to plug in an arbitrary number of tools by expanding the set of toolkens on the fly. In addition, it improves tool use by allowing extensive demonstration data for learning the toolken embeddings. In diverse domains, including numerical reasoning, knowledge-based question answering, and embodied plan generation, our approach effectively augments LLMs with tools and substantially outperforms various latest baselines. ToolkenGPT demonstrates the promising ability to use relevant tools from a large tool set in complex scenarios.

LLM Agents Basics and R&Ds

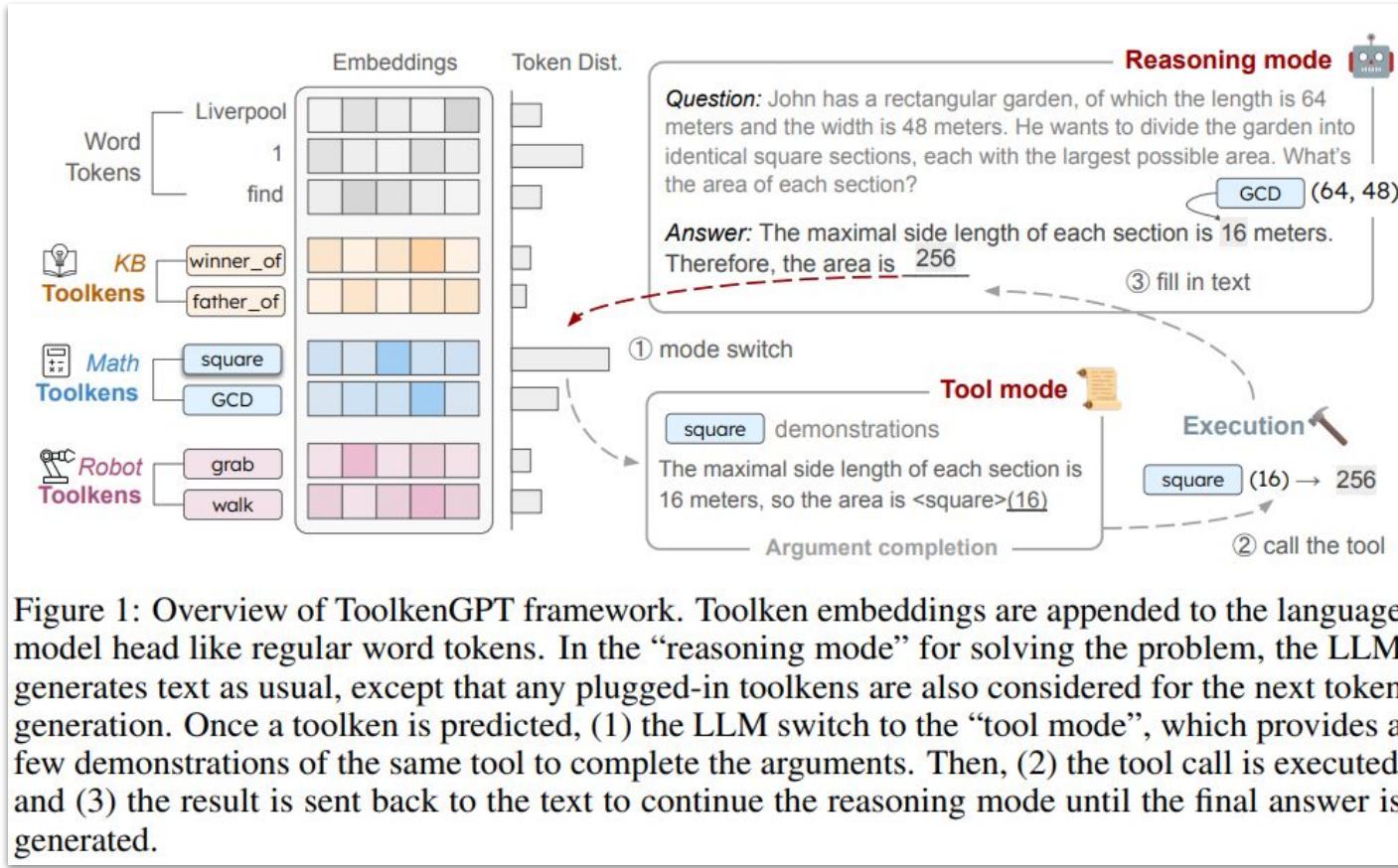


Figure 1: Overview of ToolkenGPT framework. Toolken embeddings are appended to the language model head like regular word tokens. In the “reasoning mode” for solving the problem, the LLM generates text as usual, except that any plugged-in toolkens are also considered for the next token generation. Once a toolken is predicted, (1) the LLM switch to the “tool mode”, which provides a few demonstrations of the same tool to complete the arguments. Then, (2) the tool call is executed, and (3) the result is sent back to the text to continue the reasoning mode until the final answer is generated.

LLM Agents Basics and R&Ds (Environment Rep.)

The screenshot shows a preprint on arXiv with the following details:

- Title:** WebArena: A Realistic Web Environment for Building Autonomous Agents
- Authors:** Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, Graham Neubig
- Submitted:** 25 Jul 2023 (v1), last revised 16 Apr 2024 (this version, v4)
- Actions:** CODE, Notebook, Share, Bookmark, Bell, Print
- Text Input:** Ask the author(s) a question! :)
- Ask Button:** Ask
- Powered By:** CatalyzeX
- Abstract:** A detailed description of the WebArena environment, highlighting its purpose, domains, and results.

With advances in generative AI, there is now potential for autonomous agents to manage daily tasks via natural language commands. However, current agents are primarily created and tested in simplified synthetic environments, leading to a disconnect with real-world scenarios. In this paper, we build an environment for language-guided agents that is highly realistic and reproducible. Specifically, we focus on agents that perform tasks on the web, and create an environment with fully functional websites from four common domains: e-commerce, social forum discussions, collaborative software development, and content management. Our environment is enriched with tools (e.g., a map) and external knowledge bases (e.g., user manuals) to encourage human-like task-solving. Building upon our environment, we release a set of benchmark tasks focusing on evaluating the functional correctness of task completions. The tasks in our benchmark are diverse, long-horizon, and designed to emulate tasks that humans routinely perform on the internet. We experiment with several baseline agents, integrating recent techniques such as reasoning before acting. The results demonstrate that solving complex tasks is challenging: our best GPT-4-based agent only achieves an end-to-end task success rate of 14.41%, significantly lower than the human performance of 78.24%. These results highlight the need for further development of robust agents, that current state-of-the-art large language models are far from perfect performance in these real-life tasks, and that WebArena can be used to measure such progress.

LLM Agents Basics and R&Ds (natural lang 2 web interact.)

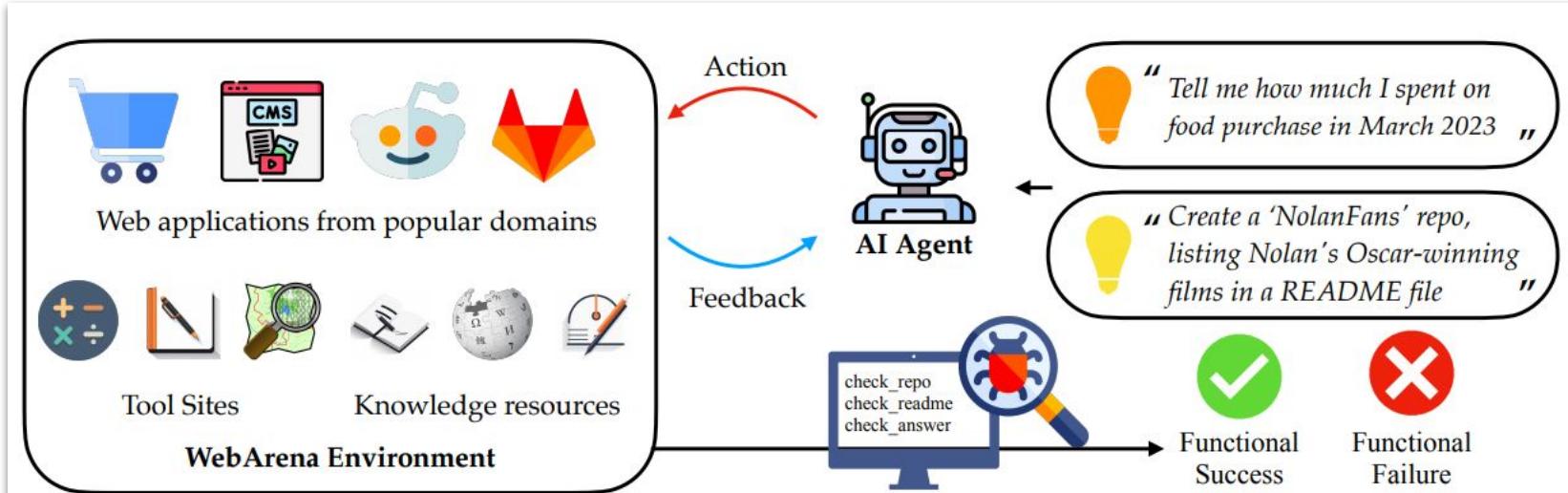


Figure 1: WebArena is a standalone, self-hostable web environment for building autonomous agents. WebArena creates websites from four popular categories with functionality and data mimicking their real-world equivalents. To emulate human problem-solving, WebArena also embeds tools and knowledge resources as independent websites. WebArena introduces a benchmark on interpreting *high-level realistic* natural language command to concrete web-based interactions. We provide validators to programmatically validate the functional correctness of each task.

LLM Agents Basics and R&Ds

“ Create an efficient itinerary to visit all of Pittsburgh's art museums with minimal driving distance starting from Schenley Park. Log the order in my “awesome-northeast-us-travel” repository ”

The diagram illustrates a high-level task being executed in WebArena across three stages:

- Step 1:** A screenshot of a web browser window titled "webarena.wikipedia.com". The page displays a list of museums in Pittsburgh, with a search bar at the top labeled "Pittsburgh museums". Below the list, there is a section about the definition of museums in Pittsburgh, Pennsylvania, and a sidebar with a "Wikimedia Commons has media related to Museums in Pittsburgh" link.
- Step 2:** A screenshot of a web browser window titled "webarena.openstreetmap.com". It shows an OpenStreetMap interface with a search bar. The user has entered "Schenley Park, Pittsburgh, Allegheny County" and "The Andy Warhol Museum, 117, Sandusky St". A "Car (OSRM)" dropdown is selected, and a "Go" button is visible. Below the map, a "Directions" panel shows two steps: "1. Start on Panther Hollow Road" and "2. Slight right onto unnamed road".
- Step 3:** A screenshot of a web browser window titled "webarena.gitlab.com". The page shows a "README.md" file with 158 B of content. The file contains a list of art museums in Pittsburgh: "Miller Gallery at Carnegie Mellon University", "American Jewish Museum", and "Carnegie Museum of Art".

Below each step, there is a blue icon of a computer keyboard and mouse, followed by a brief description:

- Search for museums in Pittsburgh
- Search for each art museum on the Map
- Record the optimized results to the repo

Figure 2: A high-level task that can be fully executed in WebArena. Success requires sophisticated, long-term planning and reasoning. To accomplish the goal (top), an agent needs to (1) find Pittsburgh art museums on Wikipedia, (2) identify their locations on a map (while optimizing the itinerary), and (3) update the README file in the appropriate repository with the planned route.

LLM Agents Basics and R&Ds

Patio, Lawn & Garden

Shop By

Shopping Options

Category

Gardening & Lawn Care [108]

Patio Furniture & Accessories [92]

Price

\$0.00 - \$999.99 [311]

\$1,000.00 - \$1,999.99 [1]

\$3,000.00 and above [1]

Compare Products

You have no items to compare.

My Wish List

You have no items in your wish list.

Items 1-12 of 320

Sort By Position 1

```
<li>
<div>
<a href="..."></a>
<div class>
<a href="...">Outdoor Patio ...
</a>
<div>
<span>Rating:</span>
<div>
<span>82%</span>
</div>
<a href="#reviews">12
<span>Reviews</span></a>
```

RootWebArea 'Patio, Lawn ..'
link 'Image'
img 'Image'
link 'Outdoor Patio..'
LayoutTable
 StaticText 'Rating:'
 generic '82%'
 link '12 Reviews'
 StaticText '\$49.99'
 button 'Add to Cart' focusable: True
 button 'Wish List' focusable: ...
 button 'Compare' focusable: ...

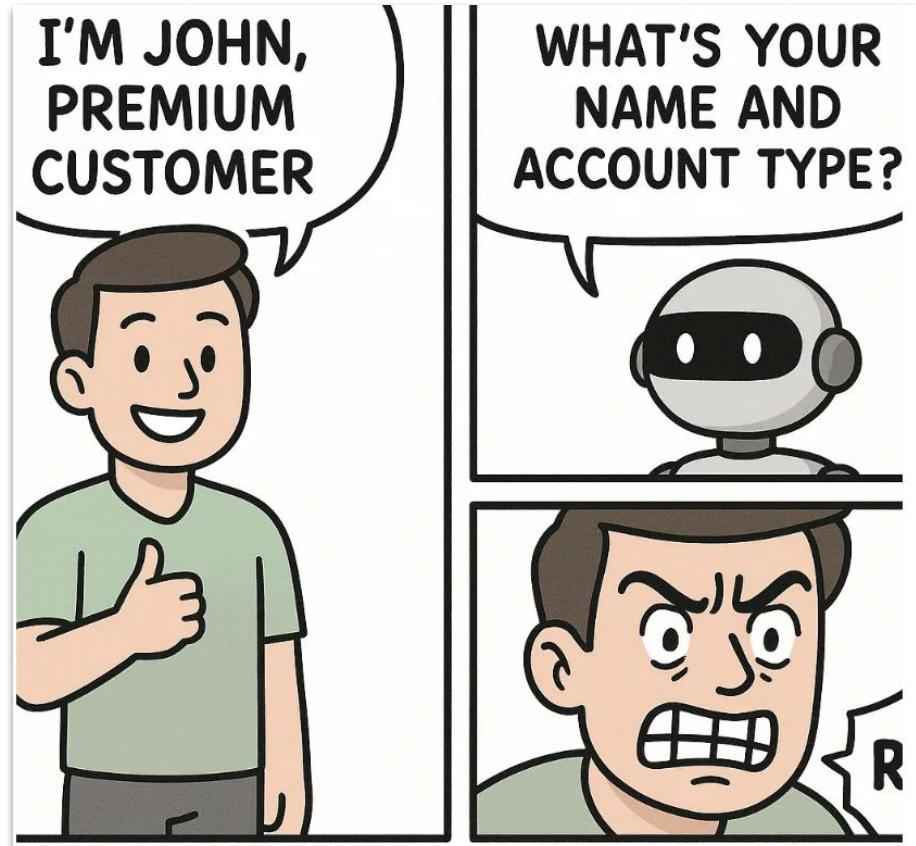
Figure 3: We design the observation to be the URL and the content of a web page, with options to represent the content as a screenshot (left), HTML DOM tree (middle), and accessibility tree (right). The content of the middle and right figures are trimmed to save space.

LLM Agents Basics and R&Ds (Memory Management)

- User: “Hi, I’m John and I need help with my premium account.”
- Bot: “Hello! I’d be happy to help. What’s your name and account type?”

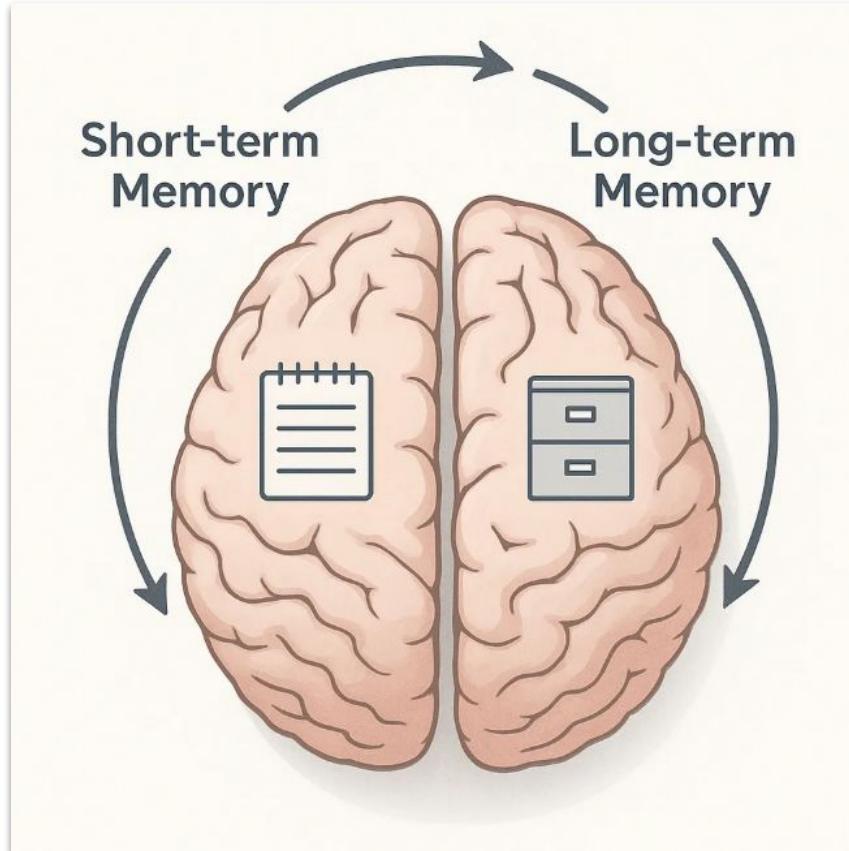
Link:

<https://medium.com/@nomannayem/building-ai-agents-that-actually-remember-a-developers-guide-to-memory-management-in-2025-062fd0be80a1>



LLM Agents Basics and R&Ds

- Just like humans, AI agents need two different types of memory



LLM Agents Basics and R&Ds

```
from langchain.memory import ConversationBufferMemory
from langchain.chains import ConversationChain
from langchain.llms import OpenAI

# Create a memory system that remembers everything
memory = ConversationBufferMemory()

# Connect it to a conversation chain
conversation = ConversationChain(
    llm=OpenAI(temperature=0.7),
    memory=memory,
    verbose=True # So you can see what's happening
)

# Now your AI remembers context!
response1 = conversation.predict(input="Hi, I'm Sarah and I love hiking")
print(response1) # AI introduces itself and acknowledges Sarah likes hiking

response2 = conversation.predict(input="What outdoor activities would you recommend")
print(response2) # AI remembers Sarah likes hiking and suggests related activities
```

LLM Agents Basics and R&Ds

```
from langchain.memory import ConversationSummaryMemory

# Create memory that summarizes old conversations
summary_memory = ConversationSummaryMemory(
    llm=OpenAI(temperature=0),
    max_token_limit=1000 # Start summarizing when we hit 1000 tokens
)

conversation_with_summary = ConversationChain(
    llm=OpenAI(temperature=0.7),
    memory=summary_memory,
    verbose=True
)

# The AI will automatically summarize older parts of the conversation
# while keeping recent messages in full detail
```

LLM Agents Basics and R&Ds

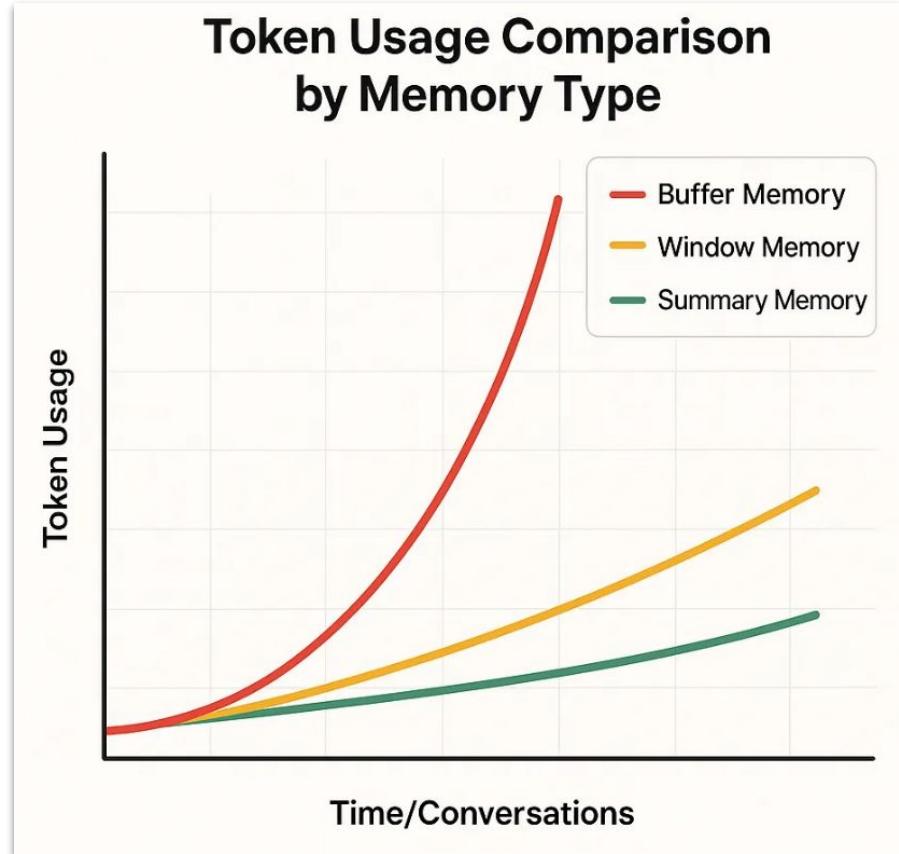
Table. Comparison of memory types

Memory Type	Best For	Pros	Cons	Code Complexity
Buffer	Short conversations	Low resource usage	Unable to handle long conversations	Simple implementation
Summary	Long conversations	Can handle large amounts of information	Summary losing useful information	Moderate
Window	General-purpose	Able to access recent information	Higher resource usage	Complex

LLM Agents Basics and R&Ds

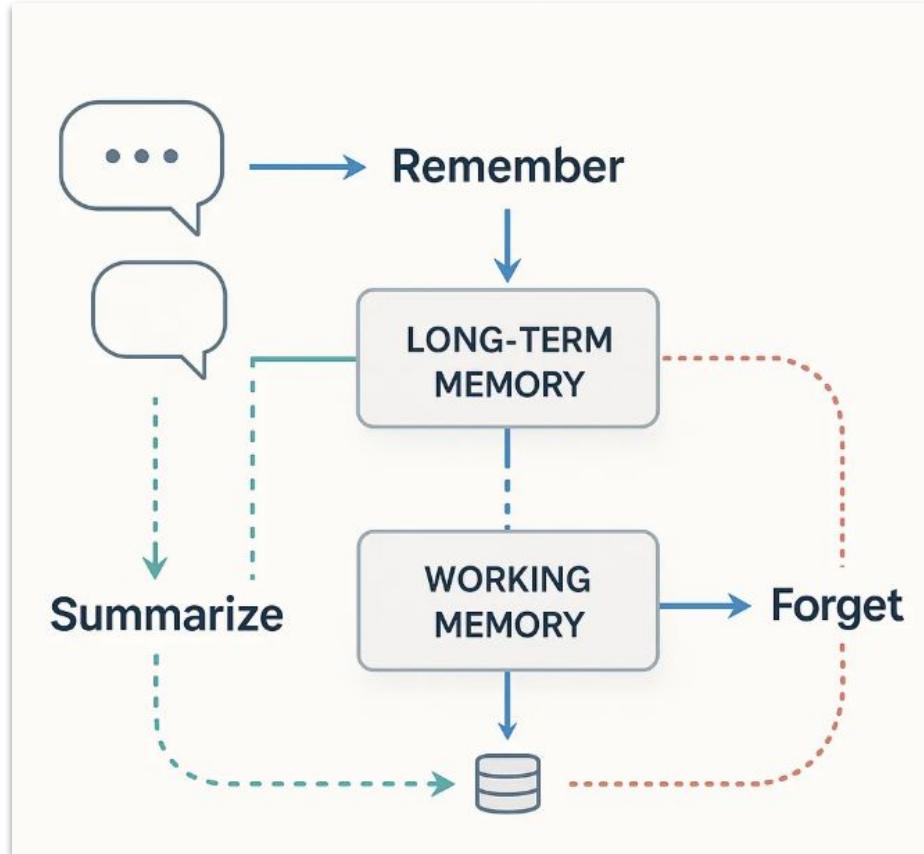
Here's what happens behind the scenes:

1. Early in the conversation: AI remembers everything word-for-word
2. As the conversation grows: AI starts summarizing older messages
3. Recent messages: Still kept in full detail
4. Very old messages: Condensed into brief summaries



LLM Agents Basics and R&Ds

- Short-term memory is like a sticky note, but long-term memory is like a filing cabinet. While short-term memory helps your AI stay coherent during a single conversation, long-term memory is what transforms your bot from a helpful tool into something that feels almost... human.



Demo: OpenHands Cloud

The screenshot shows a detailed view of an arXiv preprint page for the paper "OpenHands: An Open Platform for AI Software Developers as Generalist Agents".

Header: arXIV > cs > arXiv:2407.16741 | Help | A

Category: Computer Science > Software Engineering

[Submitted on 23 Jul 2024 (v1), last revised 18 Apr 2025 (this version, v3)]

Title: OpenHands: An Open Platform for AI Software Developers as Generalist Agents

Actions: CODE, Notebook, Share, Bookmark, Bell, Print

Authors: Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, Graham Neubig

Text Input: Ask the author(s) a question! :)

Ask Button: Ask

Powered By: CatalyzeX

Abstract: Software is one of the most powerful tools that we humans have at our disposal; it allows a skilled programmer to interact with the world in complex and profound ways. At the same time, thanks to improvements in large language models (LLMs), there has also been a rapid development in AI agents that interact with and affect change in their surrounding environments. In this paper, we introduce OpenHands (f.k.a. OpenDevin), a platform for the development of powerful and flexible AI agents that interact with the world in similar ways to those of a human developer: by writing code, interacting with a command line, and browsing the web. We describe how the platform allows for the implementation of new agents, safe interaction with sandboxed environments for code execution, coordination between multiple agents, and incorporation of evaluation benchmarks. Based on our currently incorporated benchmarks, we perform an evaluation of agents over 15 challenging tasks, including software engineering (e.g., SWE-BENCH) and web browsing (e.g., WEBARENA), among others. Released under the permissive MIT license, OpenHands is a community project spanning academia and industry with more than 2.1K contributions from over 188 contributors.

Demo: OpenHands Cloud

The screenshot shows the main interface of the OpenHands Cloud application. At the top, there is a purple header bar with a back arrow, forward arrow, refresh icon, and a URL field containing "app.all-hands.dev". Below the header is a dark-themed sidebar on the left side of the screen, featuring icons for hands, a plus sign, and a three-line menu.

The main content area has a dark background with yellow hand icons. It features a large "Let's Start Building!" heading and a subtext: "OpenHands makes it easy to build and maintain software using AI-driven development."

On the left, under "Connect to a Repository", there is a dropdown menu showing "ye-kyaw-thu/myParaphrase" and another dropdown menu below it showing "main". A "Launch" button is located at the bottom of this section. An "Add GitHub repos" link is also present.

On the right, under "Suggested Tasks", the message "No tasks available" is displayed.

Demo: OpenHands Cloud

The screenshot shows the OpenHands Cloud interface with a terminal session and a browser view tool.

Terminal Session:

```
--- 2023 ---  
Bash: 1  
Perl: 0  
Python: 91  
  
--- 2024 ---  
Bash: 7  
Perl: 0  
Python: 46  
$ echo "Generated visualization files:" && ls -la /tmp/*.png  
Generated visualization files:  
< -rw-r--r-- 1 root root 660955 Jun 23 13:11 /tmp/detailed_repository_analysis.png  
> -rw-r--r-- 1 root root 501595 Jun 23 13:11 /tmp/repository_history.png  
-rw-r--r-- 1 root root 422860 Jun 23 13:12 /tmp/technology_evolution.png  
$ ls -la /tmp/*.png  
-rw-r--r-- 1 root root 660955 Jun 23 13:11 /tmp/detailed_repository_analysis.png  
-rw-r--r-- 1 root root 501595 Jun 23 13:11 /tmp/repository_history.png  
-rw-r--r-- 1 root root 422860 Jun 23 13:12 /tmp/technology_evolution.png  
$ cat /tmp/oh-server-url  
http://localhost:60731  
$ cd /tmp && python3 -m http.server 12000 --bind 0.0.0.0 > /dev/null 2>&1 &  
[1] 4945  
$ 
```

Browser View Tool:

Now you can download the files by visiting these URLs in your browser:

1. **Repository History Graph:** https://work-1-mahuahjnzaovpfw.prod-runtime.all-hands.dev/repository_history.png
2. **Detailed Analysis Graph:** https://work-1-mahuahjnzaovpfw.prod-runtime.all-hands.dev/detailed_repository_analysis.png
3. **Technology Evolution Graph:** https://work-1-mahuahjnzaovpfw.prod-runtime.all-hands.dev/technology_evolution.png

Instructions:

- Click on each URL above
- Right-click on the image and select "Save image as..."
- Choose your desired location on your computer

Method 2: Using Browser View Tool

Let me also show you the files using the browser view tool:

Browsing completed

What do you want to build?

Agent is running task

Current Repository Information

ye-kyaw-thu/tools

Created 11m ago

Demo: OpenHands Cloud

runtime.all-hands.dev/detailed_repository_analysis.png

3. **Technology Evolution Graph:** https://work-1-mahuahjnzaopfw.prod-runtime.all-hands.dev/technology_evolution.png

Instructions:

- Click on each URL above
- Right-click on the image and select "Save image as..."
- Choose your desired location on your computer

Method 2: Using Browser View Tool

Let me also show you the files using the browser view tool:

Browsing completed ✓

Method 3: Using Command Line (if you have terminal access)

If you have command line access to download files, you can use:

Ran echo "You can also use these commands if you have a terminal access" ✓

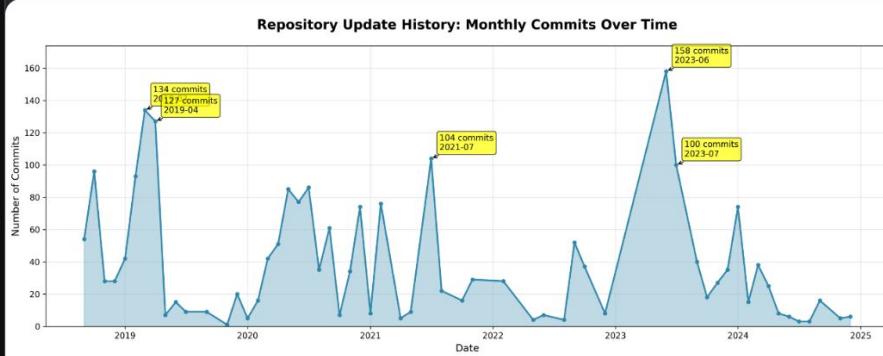
What do you want to build? ⬆

Agent has finished the task. ⬆

Changes VS Code Terminal Jupyter App BETA Browser

http://localhost:60731/view?path=/tmp/repository_history.png

Repository Update History: Monthly Commits Over Time



Date	Commits
April 2019	134
July 2019	26127
July 2021	104
March 2023	158
July 2023	100

Annual Commit Summary (2018-2024)

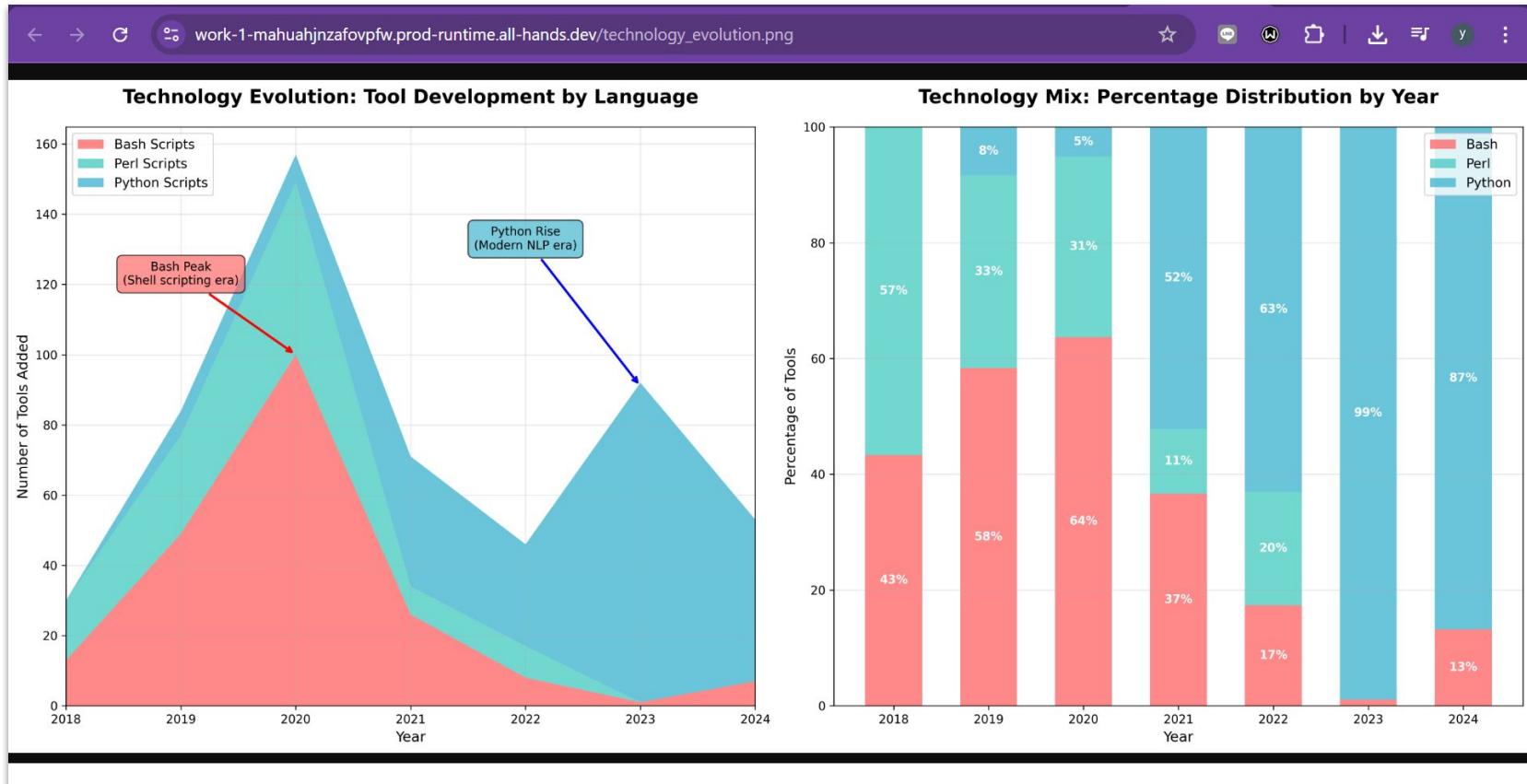


Year	Commits
2018	457
2019	573
2020	573

Current Repository Information

ye-kyaw-thu/tools Created 15m ago

Demo: OpenHands Cloud



Demo: OpenHands Cloud

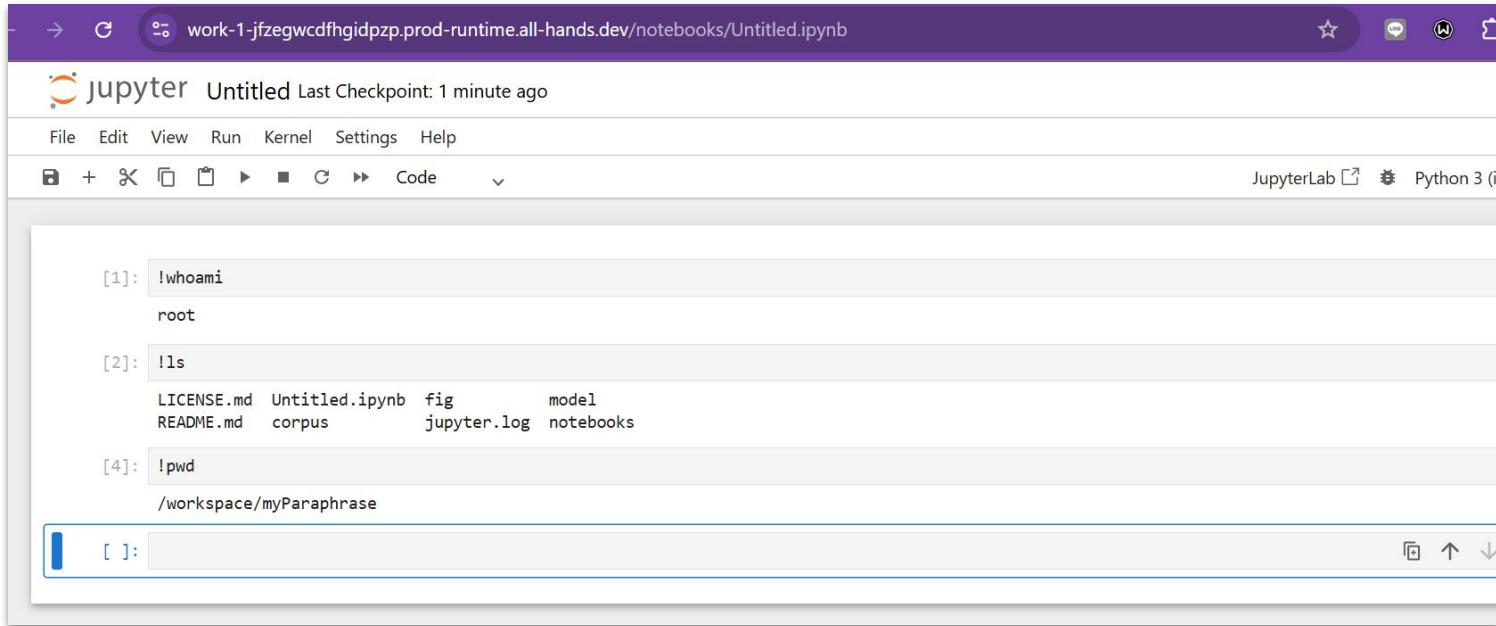
The screenshot shows a Jupyter Notebook interface with the title "CNN-Siamese-Training-with-myParaphrase-ver1.0". The notebook has a single section titled "CNN-Siamese Training for Myanmar Paraphrase" which contains a subsection "Data Information". The code cells show the user navigating to a directory, copying files to a corpora folder, and performing a word count operation. A note at the bottom indicates differences in training and test data formats.

```
[1]: !pwd  
/home/yekyaw.thu/exp/siamese/multihead-siamese-nets  
  
Copied to corpora/QQP/ folder as follows:  
  
(siamese) yekyaw.thu@gpu:~/exp/siamese/multihead-siamese-nets/corpora/QQP$ cp ../../4release/4multihead-siamese/* .  
  
[12]: !wc ./corpora/QQP/*.csv  
1001 14907 208163 ./corpora/QQP/closed-test-qqp.csv  
1001 10699 144369 ./corpora/QQP/test.csv  
40462 591453 9529408 ./corpora/QQP/train.csv  
42464 617059 9881940 total  
  
Note, training and test data format are difference as follows:  
  
[13]: !head -3 ./corpora/QQP/*.csv
```

- Running Jupyter Notebook

GitHub Link:
<https://github.com/ye-kyaw-thu/myParaphrase>

Demo: OpenHands Cloud



The screenshot shows a Jupyter Notebook interface running on the OpenHands Cloud. The title bar indicates the URL: `work-1-jfzegwcdfhgidpzp.prod-runtime.all-hands.dev/notebooks/Untitled.ipynb`. The menu bar includes File, Edit, View, Run, Kernel, Settings, Help, and a toolbar with various icons. On the right, there are links to "JupyterLab" and "Python 3 (i)". The main area displays a terminal session with the following commands:

```
[1]: !whoami  
root  
[2]: !ls  
LICENSE.md  Untitled.ipynb  fig      model  
README.md   corpus        jupyter.log  notebooks  
[4]: !pwd  
/workspace/myParaphrase
```

The input cell for the next command is currently active, indicated by a blue border.

- Very convenient
- Automation of coding, github repository updating

Demo: Mind2Web

The screenshot shows a research paper on the arXiv platform. The title is "Mind2Web: Towards a Generalist Agent for the Web". It was submitted on June 9, 2023, and last revised on December 9, 2023. The authors are Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. There is a form to ask the authors a question, and a link to CatalyzeX.

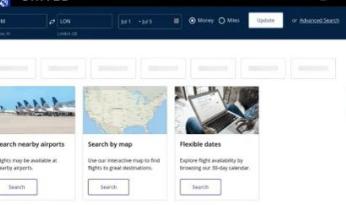
We introduce Mind2Web, the first dataset for developing and evaluating generalist agents for the web that can follow language instructions to complete complex tasks on any website. Existing datasets for web agents either use simulated websites or only cover a limited set of websites and tasks, thus not suitable for generalist web agents. With over 2,000 open-ended tasks collected from 137 websites spanning 31 domains and crowdsourced action sequences for the tasks, Mind2Web provides three necessary ingredients for building generalist web agents: 1) diverse domains, websites, and tasks, 2) use of real-world websites instead of simulated and simplified ones, and 3) a broad spectrum of user interaction patterns. Based on Mind2Web, we conduct an initial exploration of using large language models (LLMs) for building generalist web agents. While the raw HTML of real-world websites are often too large to be fed to LLMs, we show that first filtering it with a small LM significantly improves the effectiveness and efficiency of LLMs. Our solution demonstrates a decent level of performance, even on websites or entire domains the model has never seen before, but there is still a substantial room to improve towards truly generalizable agents. We open-source our dataset, model implementation, and trained models ([this https URL](#)) to facilitate further research on building a generalist agent for the web.

Demo: Mind2Web

(a) Find one-way flights from New York to Toronto.



(b) Book a roundtrip on July 1 from Mumbai to London and vice versa on July 5 for two adults..



(c) Search receipt with the eTicket 12345678 for the trip reserved by Jason Two



(d) Find a flight from Chicago to London on 20 April and return on 23 April.



(e) Search for the interactions between ibuprofen and aspirin.



(f) As a Verizon user, finance a blue iPhone 13 with 256gb along with monthly apple care.



(g) Find Elon Musk's profile and start following, start notifications and like the latest tweet.



(h) Browse comedy films streaming on Netflix that was released from 1992 to 2007.



(i) Open page to schedule an appointment for car knowledge test.



Link: <https://osu-nlp-group.github.io/Mind2Web/>

Demo: Design2Code

Design2Code: How Far Are We From Automating Front-End Engineering

Chenglei Si¹, Yanzhe Zhang², Zhengyuan Yang³, Ruibo Liu⁴, Diyi Yang¹

¹Stanford University, ²Georgia Tech, ³Microsoft, ⁴Google DeepMind



Stanford
University



Georgia
Tech.



Microsoft



DeepMind

Paper

Code

Data

Link: <https://salt-nlp.github.io/Design2Code/>

Demo: Design2Code

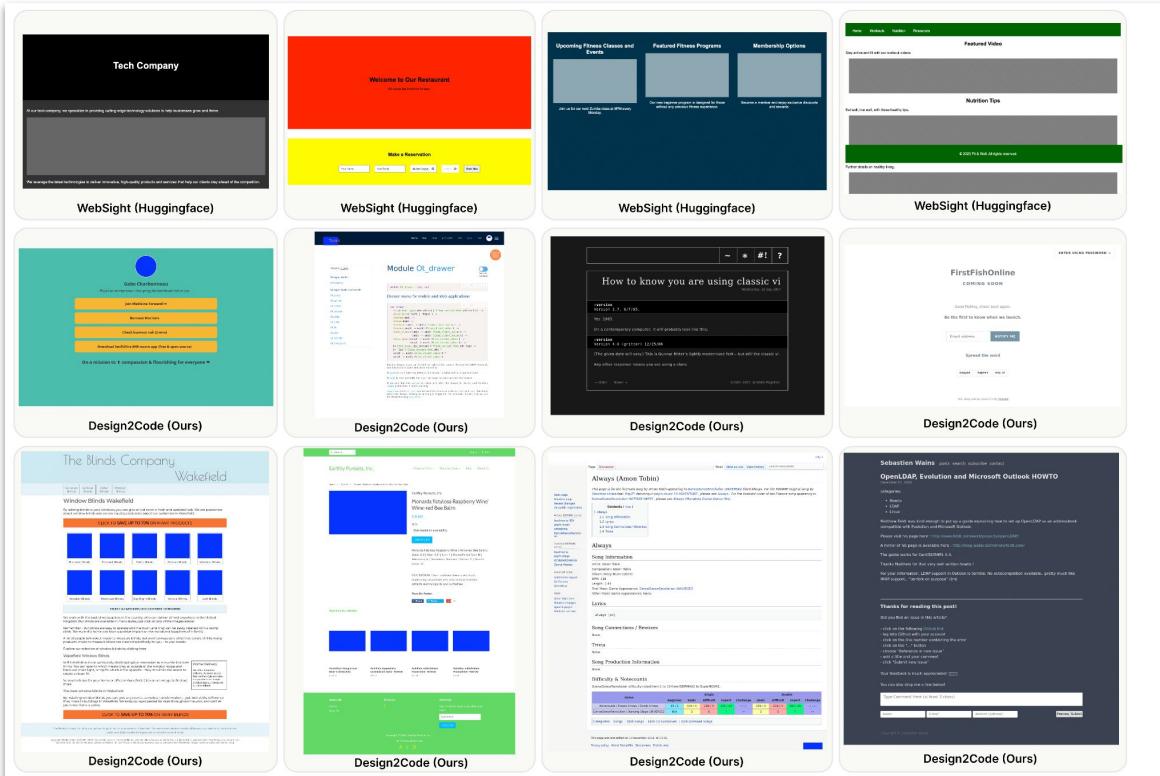


Fig. Test set examples

Demo: Design2Code

Atlantic Flyway Canada Goose Recruitment Studies

The summary for the Atlantic Flyway Canada Goose Recruitment Studies grant is detailed below. This summary states who is eligible for the grant, how much grant money will be awarded, current and past deadlines, Catalog of Federal Domestic Assistance (CFDA) numbers, and sampling of similar programs grants. Verify the accuracy of the data FederalGrants.com provides by visiting the webpage noted in the Link to Full Announcement/Program by clicking the appropriate link provided at the Grant Announcement Contact. If any sections are incomplete, please visit the website for the Fish and Wildlife Service, which is the U.S. government agency offering this grant.

Atlanta Flyway Canada Goose Recruitment Studies: THIS IS A NOTICE OF INTENT TO AWARD TO A SINGLE SOURCE, AND THERE IS NO FULL ANNOUNCEMENT ASSOCIATED WITH THIS NOTICE.

FederalGrants Search

Federal Grants Search

Federal Grants by Category

Federal Grants by Agency

ARTICLES

What is a Grant?

RESOURCES

Applicants Eligible for this Grant

Other (see text field entitled "Additional Information or Eligibility" for classification)

Additional Information on Eligibility

Foreign Government

Grant Announcement Contact

Patricia Stever, Administrative Officer Phone (PHONE REDACTED) | US

Help [EMAIL REDACTED]

Similar Government Grants

- F23AS00083 Waterfowl Management Program - 2023
- Northern Migratory Gander Program Notice of Funding Opportunity - Fiscal Year 2023
- Pacific Brant mid-winter ground survey in Mexico
- Migratory Bird Monitoring, Assessment and Conservation
- Western Canada Cooperative Banding Program
- Migratory Bird Project Support: Monitoring, Assessment, Conservation
- Migratory Bird Project Support: Monitoring, Assessment, Conservation of Red Knots

More Grants from the Fish and Wildlife Service

- F23AS00139 F23020 Implementation of the Quagga and Zebra Mussels Action Plan (QZAP) in the ...
- F23AS00140 F23020 Implementation of the Quagga and Zebra Mussels Action Plan (QZAP) in the ...
- F23AS00149 Fisheries Restoration and Irrigation Mitigation Act Fiscal Year 2023
- F23AS00180 Science in the Service Internship Funding Opportunity
- F23AS00187 Aquatic Invasive Species Research

Reference Webpage

Atlantic Flyway Canada Goose Recruitment Studies

THIS IS A NOTICE OF INTENT TO AWARD TO A SINGLE SOURCE, AND THERE IS NO FULL ANNOUNCEMENT ASSOCIATED WITH THIS NOTICE.

RESEARCH

Federal Grants Search

Federal Grants by Category

Federal Grants by Agency

ARTICLES

What is a Grant?

RESOURCES

Home

Federal Grants Search

Grants for Women

Federal Pell Grant

Contact

Privacy Policy

Terms

Atlantic Flyway Canada Goose Recruitment Studies

More details about the grant...

GPT-4V Direct Prompting

THIS IS A NOTICE OF INTENT TO AWARD TO A SINGLE SOURCE, AND THERE IS NO FULL ANNOUNCEMENT ASSOCIATED WITH THIS NOTICE.

Federal Grant Title:

Atlantic Flyway Canada Goose Recruitment Studies

Federal Agency Name:

Fish and Wildlife Service

Grant Categories:

Natural Resources

Type of Opportunity:

Discretionary

Applicants Eligible for this Grant

Others (see text field entitled "Additional Information on Eligibility" for classification)

Additional Information on Eligibility

Foreign Government

Grant Announcement Contact

Patricia Stever, Administrative Officer Phone (PHONE REDACTED)
Help [EMAIL REDACTED]

Similar Government Grants

- F23AS00117 Webless Migratory Game Bird Program 2023
- F23AS00120 Migratory Gamebird Program Notice of Funding Opportunity - Fiscal Year 2023
- Pacific Brant mid-winter ground survey in Mexico
- Migratory Bird Monitoring, Assessment and Conservation
- Western Canada Cooperative Banding Program
- Migratory Bird Project Support: Monitoring, Assessment, Conservation of Red Knots

More Grants from the Fish and Wildlife Service

- F23AS00193 FY2023 Implementation of the Quagga and Zebra Mussel Action Plan (QZAP) in the ...
- F23AS00194 FY2023 Implementation of the Quagga and Zebra Mussel Action Plan (QZAP) in the ...
- F23AS00199 Fisheries Restoration and Irrigation Mitigation Act Fiscal Year 2023
- F23AS00239 Science in the Service Internship Funding Opportunity
- F23AS00237 Aquatic Invasive Species Research

FederalGrants.com is not endorsed by, or affiliated with, any government agency. Copyright ©2007-2023
FederalGrants.com

GPT-4V Text-Augmented Prompting

Fig. some case study examples to compare between different prompting methods and different models

Demo: Design2Code

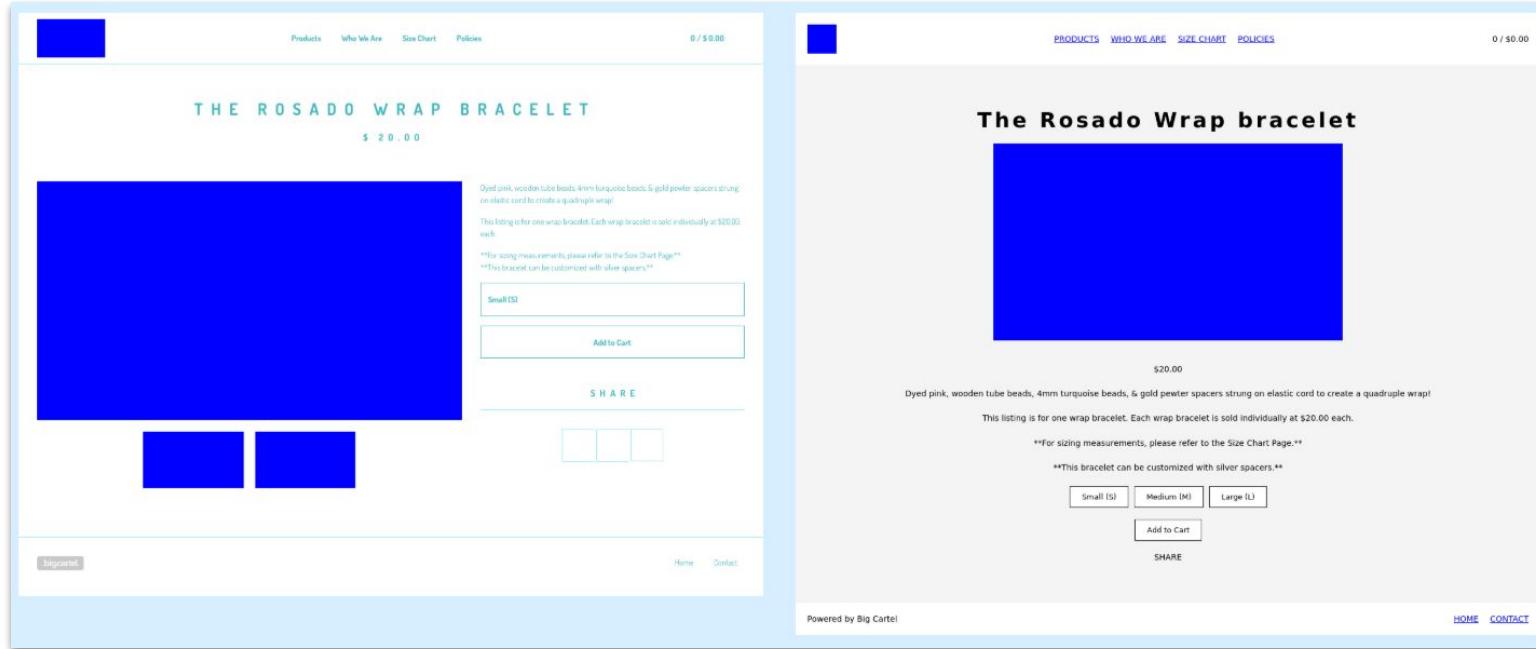


Fig. Examples of GPT-4V generated webpages in comparison with the original reference webpages. (Left: test dat)

Demo: Agentic Object Detection

va.landing.ai/demo/agentic-od

Agentic Object Detection

An industry-leading object detection app that identifies highly specific items, like "gaming chair" or "green shoe" with a text prompt. It returns a list of bounding boxes with normalized coordinates and label names.

Input

Try one of these images ▾

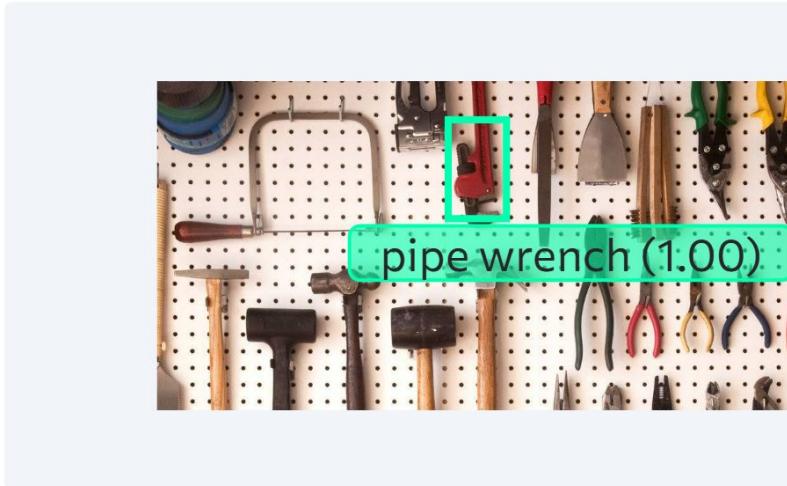
Tool Prompt Hide Tips

pipe wrench

Use singular terms for better detection (e.g., use 'chair' to find chairs).
Only one object type can be detected at a time. Use separate searches for different objects.

Run Now

Output



pipe wrench (1.00)

Link: <https://va.landing.ai/demo/agentic-od>

Demo: Agentic Object Detection

Output

JSON output

⊕ ⊖ Reset



Match (1.00)

output

Download Copy

```
[  
  [  
    {  
      "label": "pipe wrench",  
      "score": 1,  
      "bounding_box": [  
        456,  
        61,  
        548,  
        217  
      ]  
    }  
  ]  
]
```

Fig. JSON output

Demo: Agentic Object Detection



The screenshot shows a browser window with the title "API". Below the title, there are three tabs: "Python", "JavaScript", and "cURL". The "Python" tab is selected, displaying the following code:

```
python

import requests

url = "https://api.va.landing.ai/v1/tools/agentic-object-detection"
files = {
    "image": open="{{path_to_image}}", "rb"
}
data = {
    "prompts": "{{prompt}}",
    "model": "agentic"
}
```

Below the code, there is a link labeled "API key" with a copy icon. At the bottom of the window, there is a section titled "Pay as You Go for API Usage" with the text: "Get \$10 in free credits to start, with a charge of \$0.03 per image processed. For higher usage and advanced".

Fig. Python API

Demo: Agentic Object Detection



The image shows a screenshot of a web-based API documentation interface. At the top left, it says "API". Below that are three tabs: "Python", "JavaScript", and "cURL". The "cURL" tab is selected. A large text area contains a cURL command for "shell". The command is as follows:

```
curl --request POST \
--url https://api.va.landing.ai/v1/tools/agentic-object-detection \
--header 'Authorization: Basic {{your_api_key}}' \
--form "image=@{{path_to_image}}" \
--form "prompts={{prompt}}" \
--form "model=agentic"
```

At the bottom left of the code block, there is a link labeled "API key" with a small icon next to it.

Fig. cURL

Demo: Visual Q&A

Prompt: Can you explain me about the attached photo?

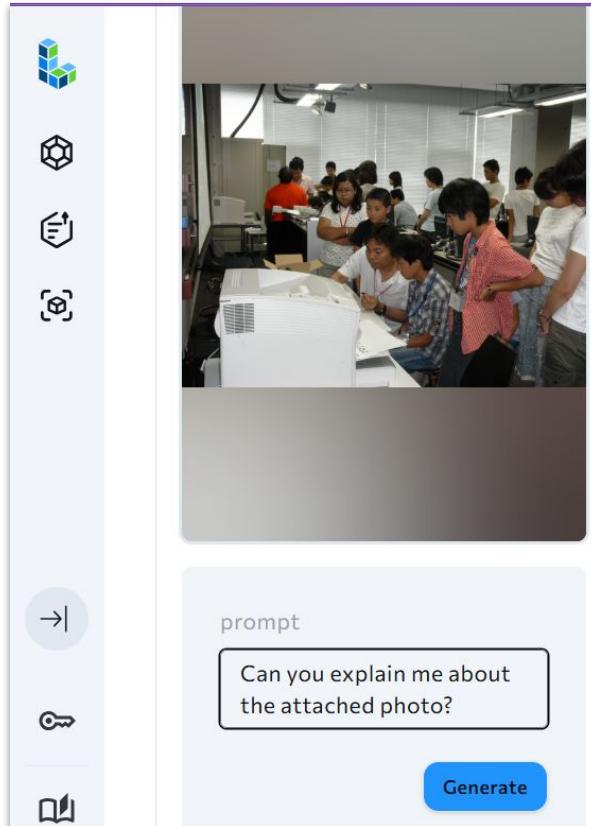
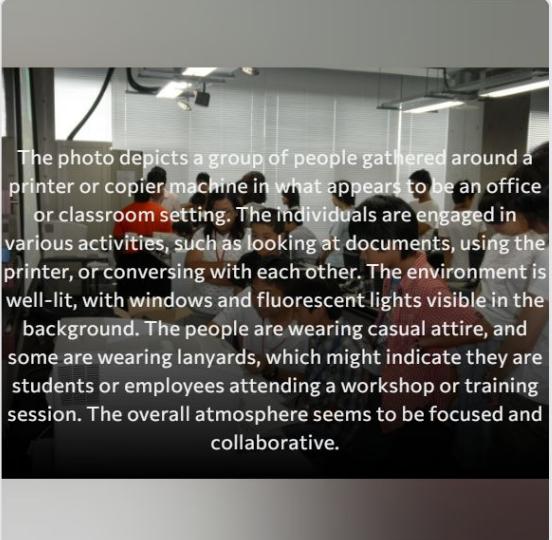


Fig. Qwen2 VL Images VQA

Demo: Visual Q&A

Output



The photo depicts a group of people gathered around a printer or copier machine in what appears to be an office or classroom setting. The individuals are engaged in various activities, such as looking at documents, using the printer, or conversing with each other. The environment is well-lit, with windows and fluorescent lights visible in the background. The people are wearing casual attire, and some are wearing lanyards, which might indicate they are students or employees attending a workshop or training session. The overall atmosphere seems to be focused and collaborative.

json

```
{  
  "data": "The photo depicts a group of people gathered around a printer or copier machine in what appears to be an office or classroom setting. The individuals are engaged in various activities, such as looking at documents, using the printer, or conversing with each other. The environment is well-lit, with windows and fluorescent lights visible in the background. The people are wearing casual attire, and some are wearing lanyards, which might indicate they are students or employees attending a workshop or training session. The overall atmosphere seems to be focused and collaborative."  
}
```

Fig. Output

The photo depicts a group of people gathered around a printer or copier machine in what appears to be an office or classroom setting. The individuals are engaged in various activities, such as looking at documents, using the printer, or conversing with each other. The environment is well-lit, with windows and fluorescent lights visible in the background. The people are wearing casual attire, and some are wearing lanyards, which might indicate they are students or employees attending a workshop or training session. The overall atmosphere seems to be focused and collaborative.

Demo: Video Tracking

Prompt: soccer, human

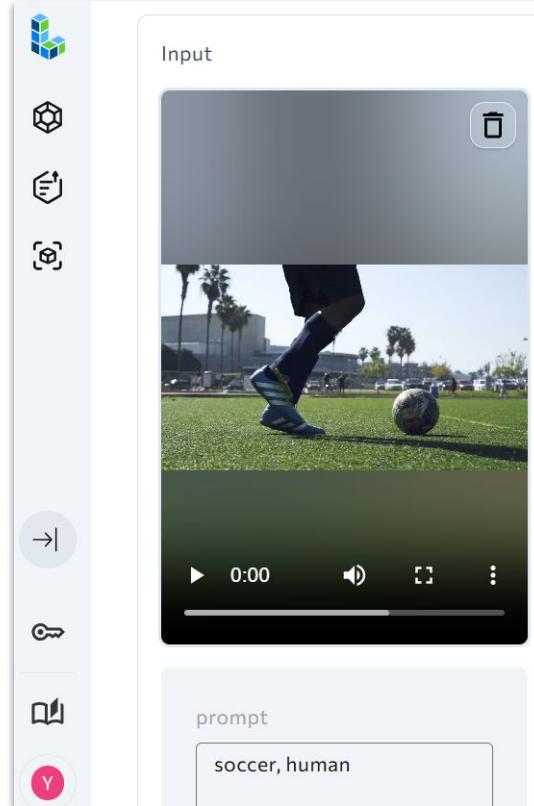


Fig. Florence-2 Sam2 Video Tracking

Demo: Video Tracking

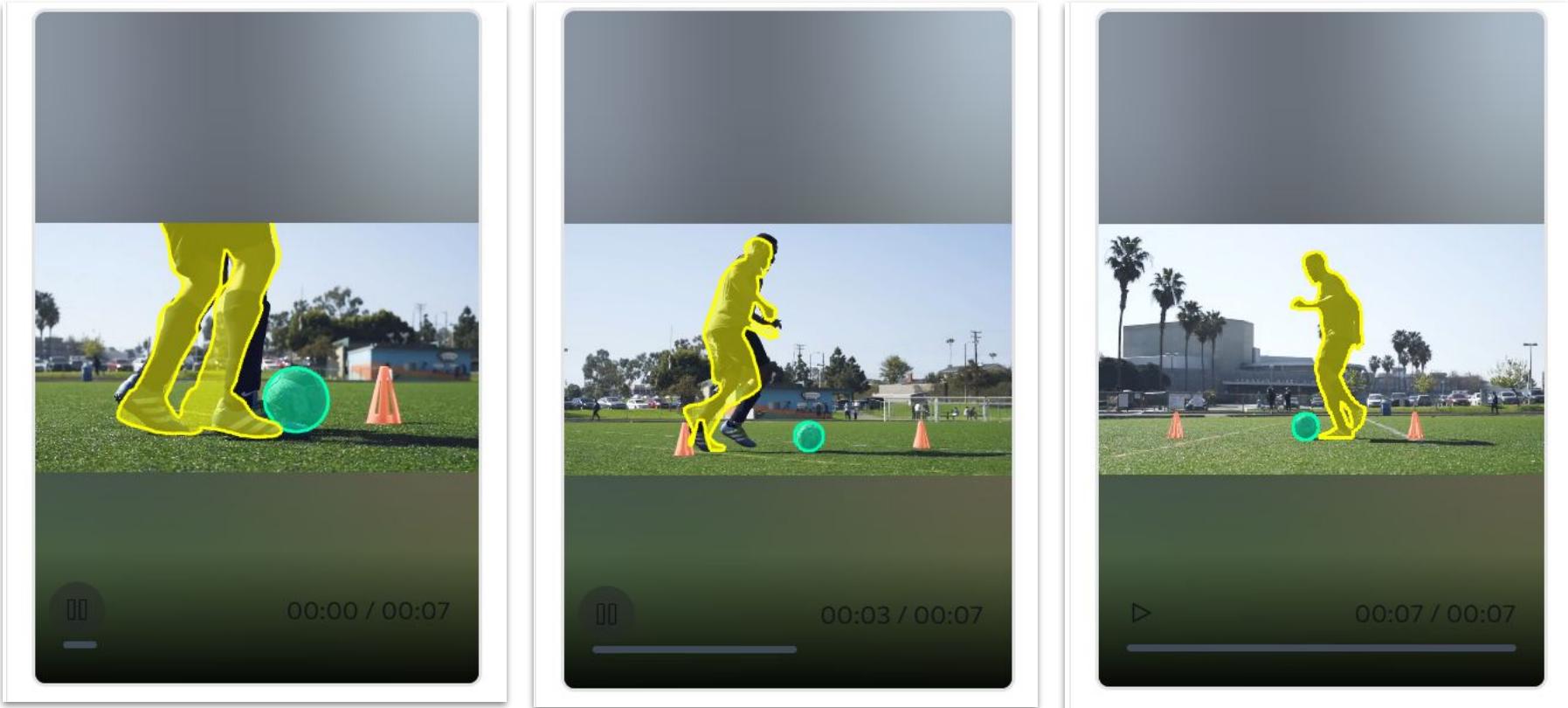


Fig. Video tracking result for soccer and human

Don't Build Multi-Agents



DEVIN +

BLOG

CONTACT

Why think about principles?

HTML was introduced in 1993. In 2013, Facebook released React to the world. It is now 2025 and React (and its descendants) dominates the way developers build sites and apps. Why? Because React is not just a scaffold for writing code. It is a philosophy. By using React, you embrace building applications with a pattern of reactivity and modularity, which people now accept to be a standard requirement, but this was not always obvious to early web developers.

In the age of LLMs and building AI Agents, it feels like we're still playing with raw HTML & CSS and figuring out how to fit these together to make a good experience. No single approach to building agents has become the standard yet, besides some of the absolute basics.

Link: <https://cognition.ai/blog/dont-build-multi-agents#principles-of-context-engineering>

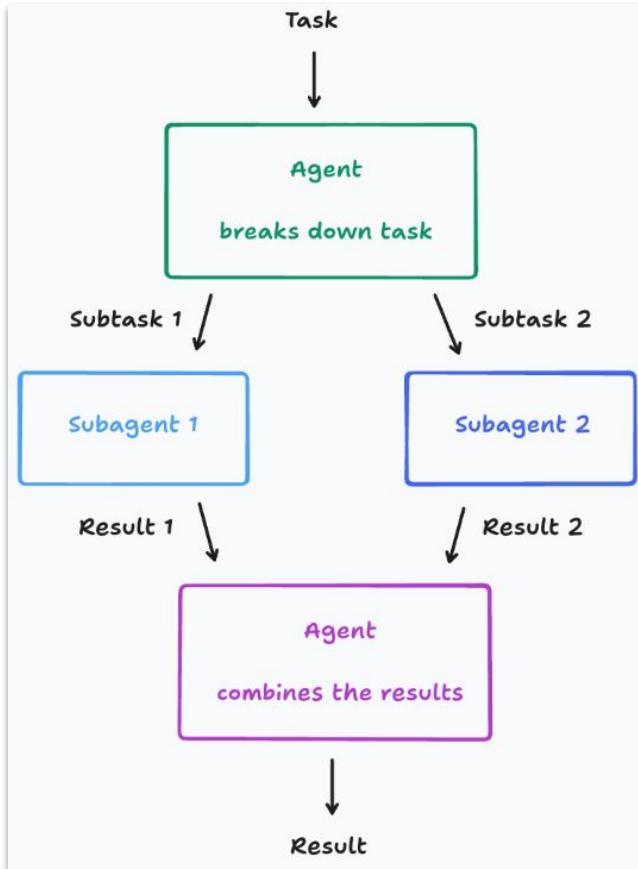
Don't Build Multi-Agents

In some cases, libraries such as <https://github.com/openai/swarm> by OpenAI and <https://github.com/microsoft/autogen> by Microsoft actively push concepts which I believe to be the wrong way of building agents. Namely, using multi-agent architectures, and I'll explain why.

That said, if you're new to agent-building, there are lots of resources on how to set up the basic scaffolding [1] [2]. But when it comes to building serious production applications, it's a different story.

Fig. Walden Yan (Cognition Labs, US) made the above claim

Don't Build Multi-Agents



“Context engineering” is the next level of this
Take an example of a common type of agent. This agent

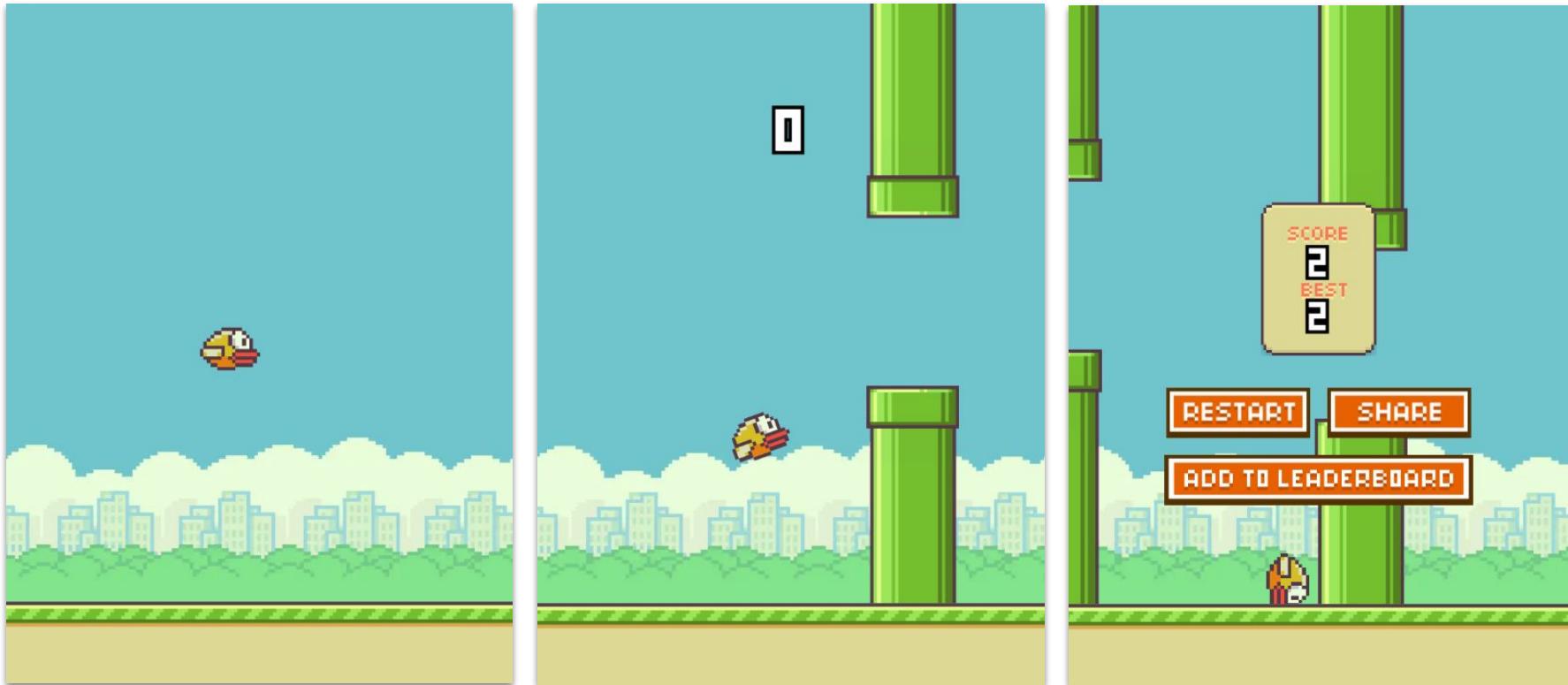
1. Breaks its work down into multiple parts
2. Starts subagents to work on those parts
3. Combines those results in the end

Don't Build Multi-Agents

Suppose your **Task** is “build a Flappy Bird clone”. This gets divided into **Subtask 1** “build a moving game background with green pipes and hit boxes” and **Subtask 2** “build a bird that you can move up and down”.

It turns out subagent 1 actually mistook your subtask and started building a background that looks like Super Mario Bros. Subagent 2 built you a bird, but it doesn’t look like a game asset and it moves nothing like the one in Flappy Bird. Now the final agent is left with the undesirable task of combining these two miscommunications.

Don't Build Multi-Agents



Flappy Game Link: <https://flappybird.io/>

Don't Build Multi-Agents

- Principle 1
- Share context, and share full agent traces, not just individual messages
- However, still unreliable

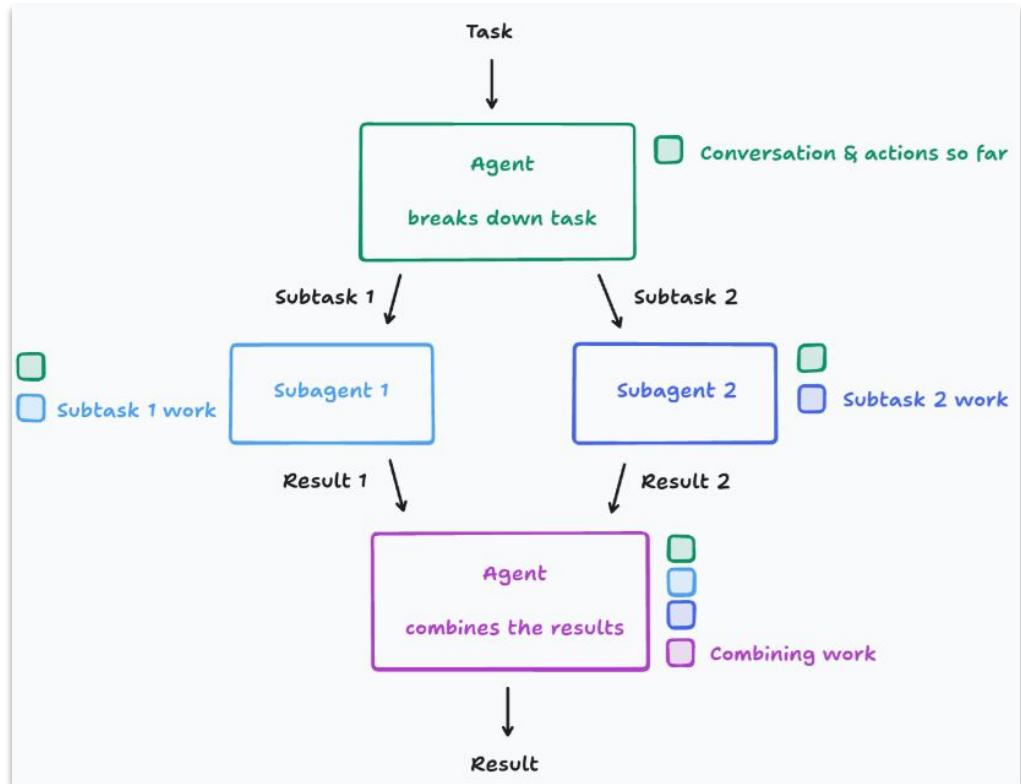


Fig. Context sharing among subagents

Don't Build Multi-Agents

Unfortunately, we aren't quite out of the woods. When you give your agent the same Flappy Bird cloning task, this time, you might end up with a bird and background with completely different visual styles. Subagent 1 and subagent 2 cannot see what the other was doing and so their work ends up being inconsistent with each other.

The actions subagent 1 took and the actions subagent 2 took were based on conflicting assumptions not prescribed upfront.

Fig. Walden Yan (Cognition Labs, US) made the above claim

Don't Build Multi-Agents

- Principle 2
- Actions carry implicit decisions, and conflicting decisions carry bad results
- I would argue that Principles 1 & 2 are so critical, and so rarely worth violating, that you should by default rule out any agent architectures that don't abide by them.
- You might think this is constraining, but there is actually a wide space of different architectures you could still explore for your agent.

Don't Build Multi-Agents

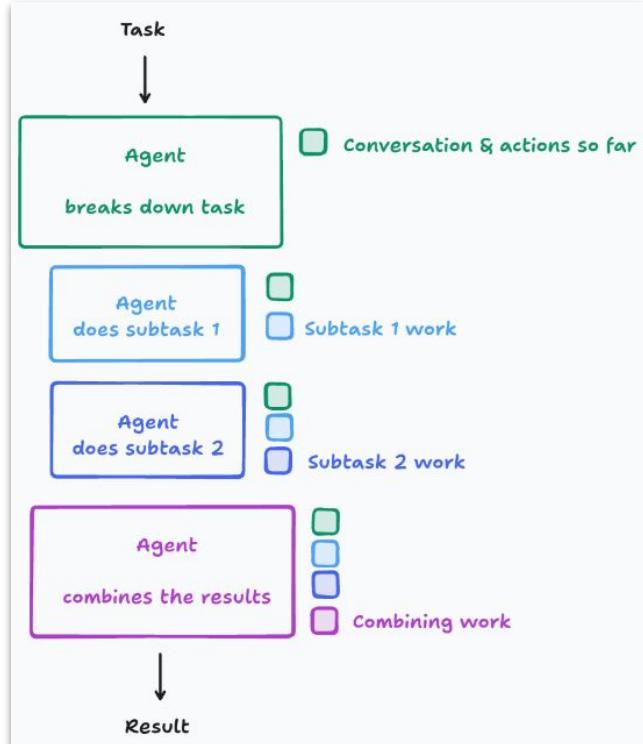


Fig. Simple & reliable, the context is continuous

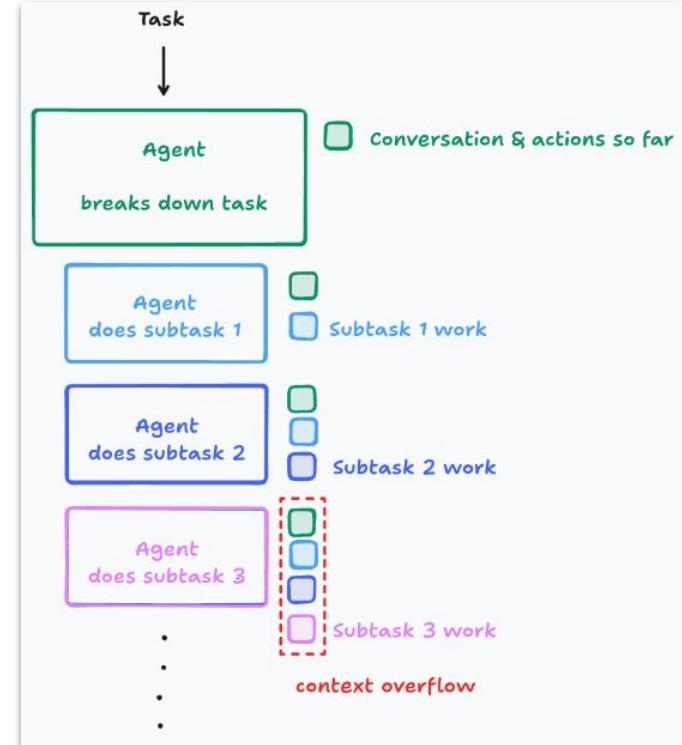
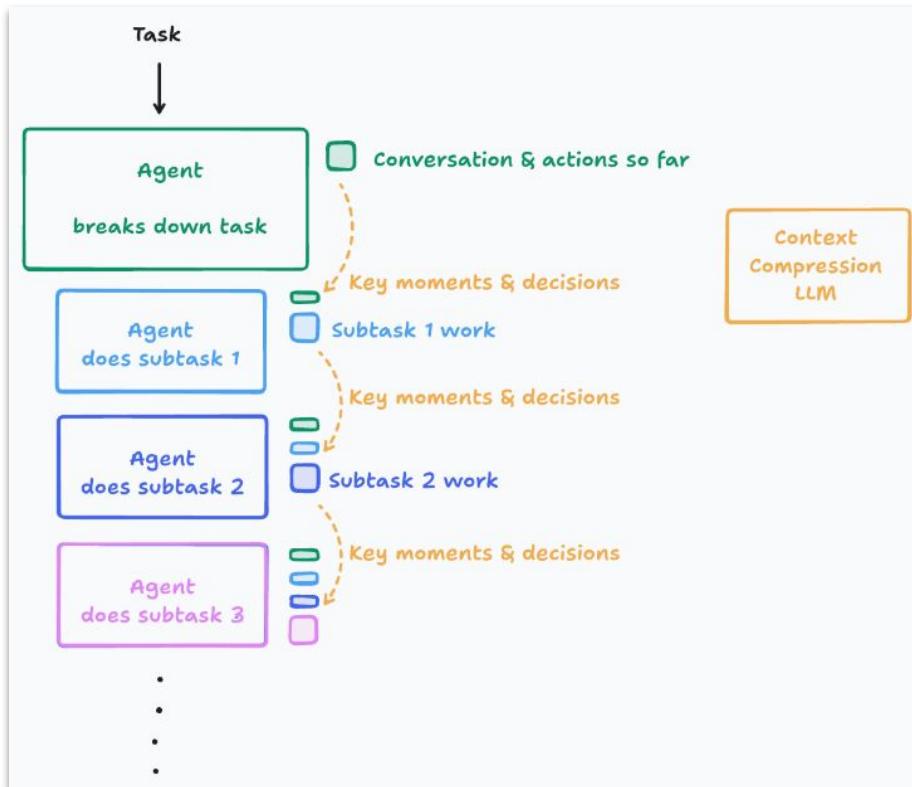


Fig. However, struggle with longer tasks

Don't Build Multi-Agents



1. introduce a new LLM model whose key purpose is to compress a history of actions & conversation into key details, events, and decisions
2. This is hard to get right.

Fig. Reliable on longer tasks

Summary

- Safety & Control
 - Sandboxing, credential management, and auditing will remain critical for secure agent deployment.
- LLM Advancements
 - Continued investment in training will enhance capabilities while reducing costs.
- Benchmarking Challenges
 - Evaluation methods will improve, but measuring true agent intelligence remains difficult.

Summary

- Invisible Integration
 - Agents will increasingly operate through native interfaces, making software interactions less visible.
- Accelerating Progress
 - Development cycles will shorten, and agents will soon autonomously build other agents.

References

1. Agent AI: Graham Neubig, Professor, SCS, Carnegie Mellon University.
Co-Founder, All Hands AI: <https://www.youtube.com/watch?v=DNKb1loJoaU>
2. Design2Code: How Far Are We From Automating Front-End Engineering?
<https://github.com/NoviScl/Design2Code?tab=readme-ov-file>
3. CMU Advanced NLP Fall 2024 (15): Tool Use and LLM Agent Basics:
<https://www.youtube.com/watch?v=a3SjRsqV9ZA&t=1s>
4. CMU Advanced NLP Fall 2024 (16): Agents for Software Development and Web Browsing: <https://www.youtube.com/watch?v=lcNVAYNlp20&t=1s>
5. Stanford Webinar - Agentic AI: A Progression of Language Model Usage:
<https://www.youtube.com/watch?v=kJLiOGle3Lw&t=2s>

References

6. LU Lab GitHub: <https://github.com/ye-kyaw-thu>
7. Shumaker et al. Animal tool behavior: the use and manufacture of tools by animals. JHU Press, 2011
8. ToolkenGPT: Augmenting Frozen Language Models with Massive Tools via Tool Embeddings, Shibo Hao, Tianyang Liu, Zhen Wang, Zhiting Hu, <https://arxiv.org/abs/2305.11554>
9. WebArena: A Realistic Web Environment for Building Autonomous Agents, Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, Graham Neubig, <https://arxiv.org/abs/2307.13854>
10. Building AI Agents That Actually Remember: A Developer's Guide to Memory Management in 2025, Nayeem Islam: <https://medium.com/@nomannayeem/building-ai-agents-that-actually-remember-a-developers-guide-to-memory-management-in-2025-062fd0be80a1>

References

11. OpenHands: An Open Platform for AI Software Developers as Generalist Agents, Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, Graham Neubig, <https://arxiv.org/abs/2407.16741>
12. Mind2Web: Towards a Generalist Agent for the Web, Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, Yu Su, <https://arxiv.org/abs/2306.06070>
13. Mind2Web Project Link: <https://osu-nlp-group.github.io/Mind2Web/>
14. Flappy Bird Game: <https://flappybird.io/>
15. Design2Code Project Page: <https://salt-nlp.github.io/Design2Code/>

References

16. Andrew Ng Explores The Rise Of AI Agents And Agentic Reasoning | BUILD 2024 Keynote:
https://www.youtube.com/watch?v=KrRD7r7y7NY&list=PLW0CAVDRXv0Y7LxtdHAqx_QedUt8HIsKc
17. Agentic Object Detection: <https://va.landing.ai/demo/agentic-od>
18. Qwen2 VL Images VQA:
<https://va.landing.ai/demo/api/Qwen2%20VL%20Images%20VQA>
19. Andrej Karpathy: Software Is Changing (Again):
<https://www.youtube.com/watch?v=LCEmiRjPEtQ>
20. Awesome LLM Apps: <https://github.com/Shubhamsaboo/awesome-llm-apps>

Thank you!

Homepage: <https://sites.google.com/site/yekyawthunlp/>

GitHub: <https://github.com/ye-kyaw-thu>

Email: ykt.nlp.ai@gmail.com