Teaching LLMs to Use Tools at Scale: Function Calling to Agents

GORILLA

Shishir G. Patil
18 Nov 2024

BAIR sky
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

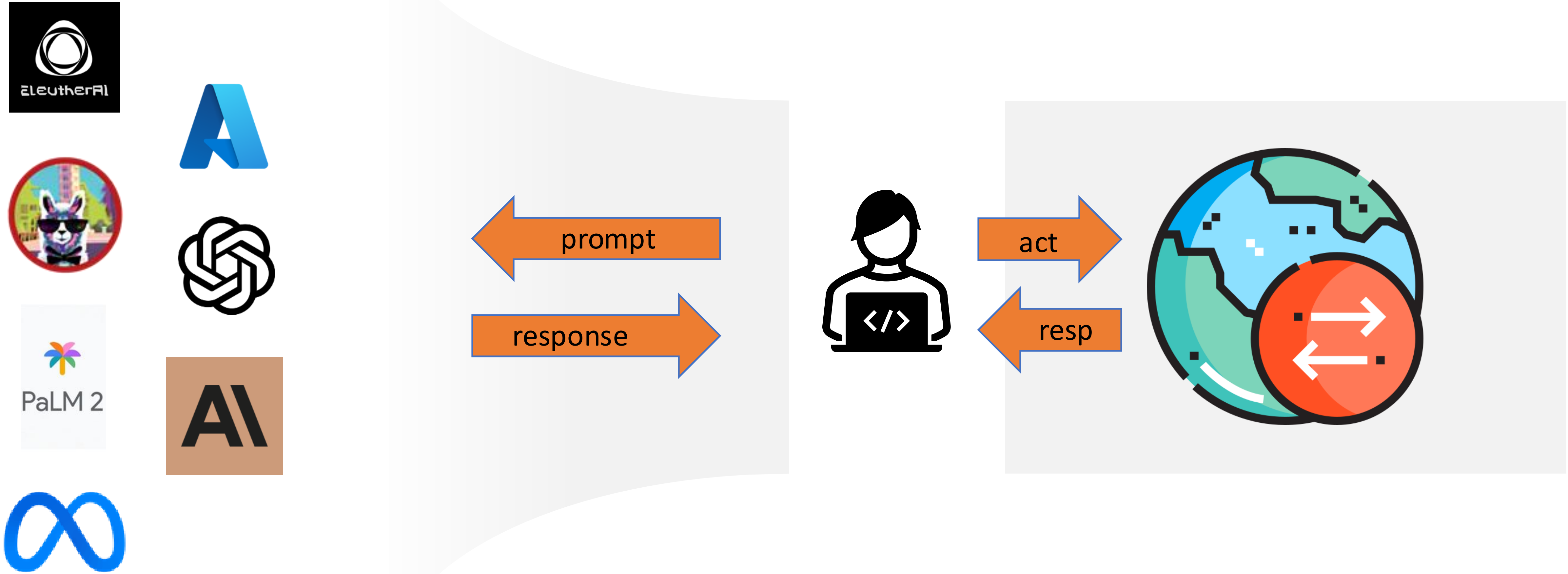
Agenda

Function calls – connecting LLMs to the digital world

- How to train models to do function calls?
- How do we evaluate the models?

From *function calls* to *agentic system*.

- Building a run-time for LLM-agents
- How to evaluate LLM-agents?

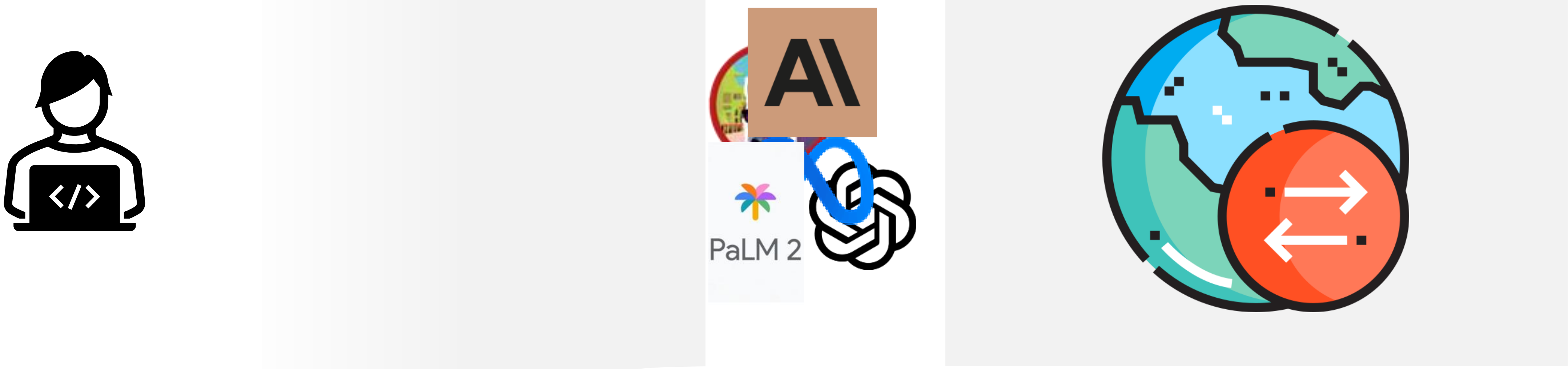


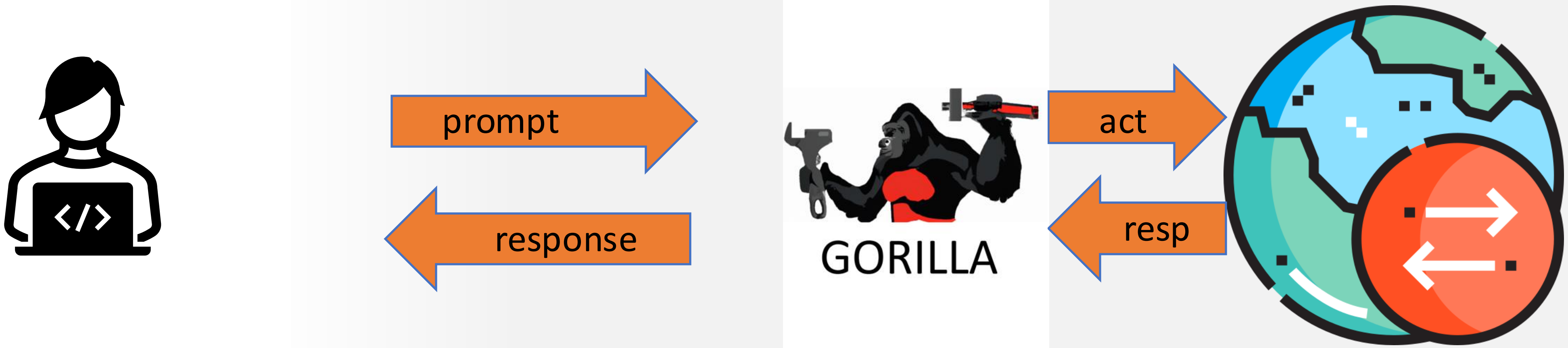
prompt

response

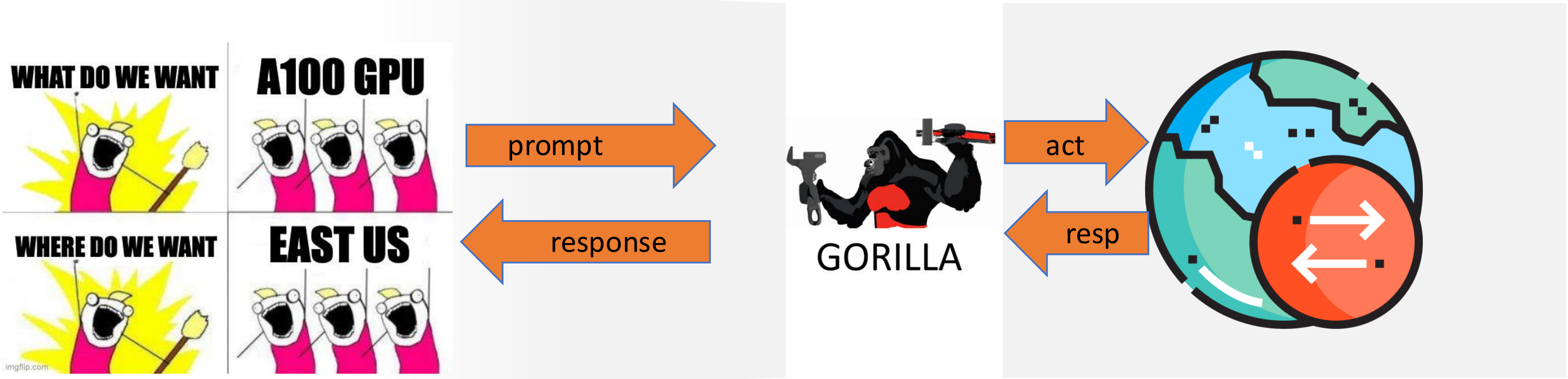
act

resp

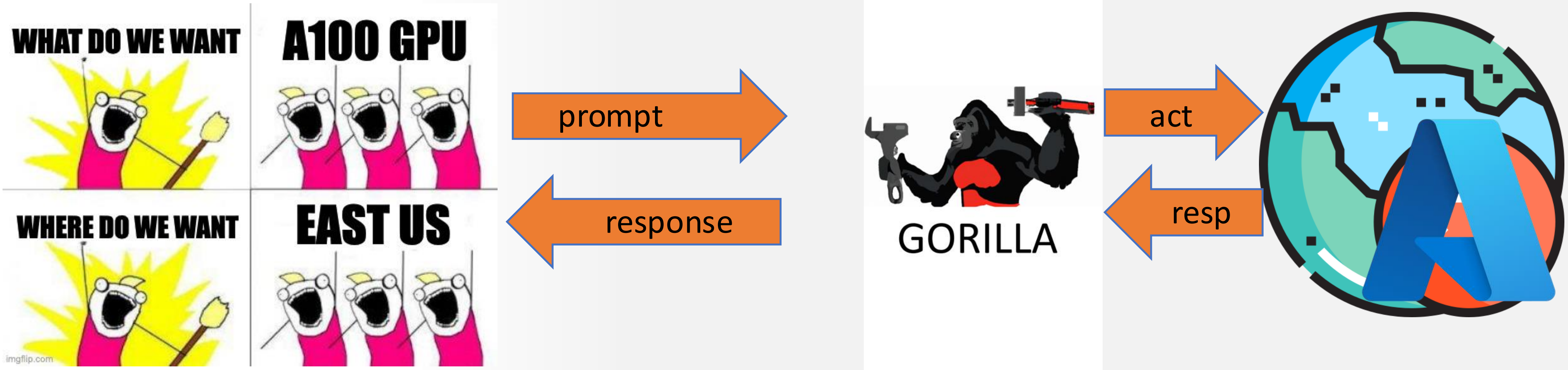

AI
PaLM 2



Humans are good discriminators,
LLMs are good generators



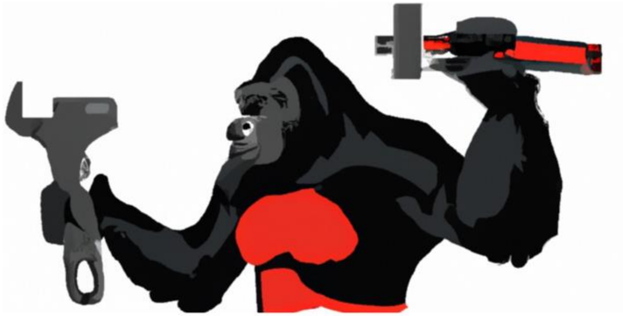
Humans are good discriminators,
LLMs are good generators



Humans are good discriminators,
LLMs are good generators

Let LLMs pick the right APIs for a given task!

WHAT DO WE WANT
A100 GPU
WHERE DO WE WANT
EAST US
imgflip.com



GORILLA

HUGGING FACE

stripe

aws

salesforce

TensorFlow

PyTorch

the Big Question:

Q1. How to mix Fine-Tuning and Retrieval?

Hypothesis (at least what we wish were true):

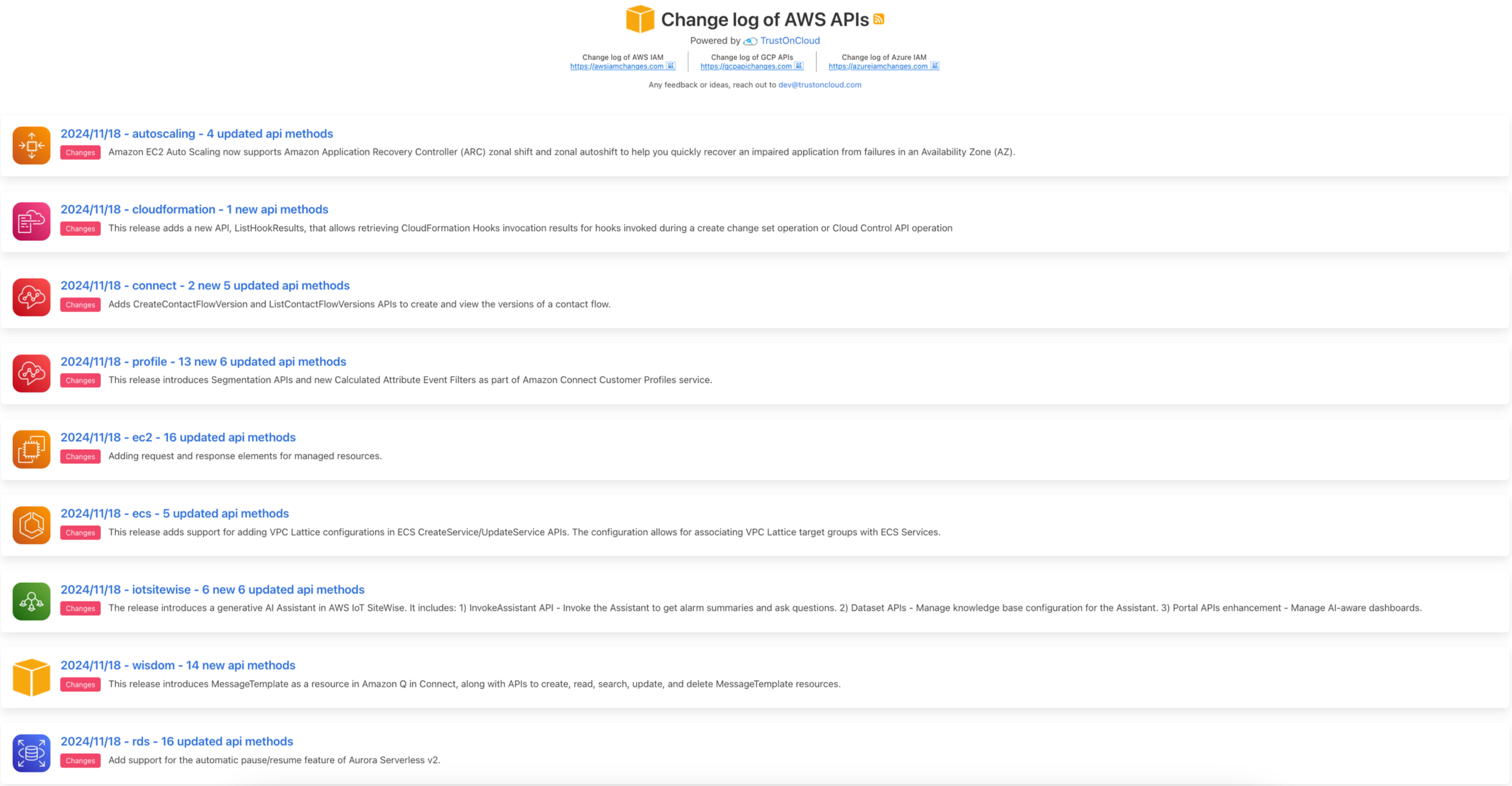
- **Fine-Tuning:** augment the **behavior** of the model
- **Retrieval:** introduce new **knowledge** to the model

Early Evidence (Gorilla): fine-tuning is remarkably effective at behavior and knowledge.

- **…** but you still need retrievers for ***data freshness***
- **…** and **retrievers are inaccurate**

Solution: **RAT**

***Data freshness**: APIs change a lot more frequently than we think*



Change log of AWS APIs

Powered by TrustOnCloud

Change log of AWS IAM https://awsiamchanges.com | Change log of GCP APIs https://gcpapichanges.com | Change log of Azure IAM https://azureiamchanges.com

Any feedback or ideas, reach out to dev@trustoncloud.com

2024/11/18 - autoscaling - 4 updated api methods
Changes Amazon EC2 Auto Scaling now supports Amazon Application Recovery Controller (ARC) zonal shift and zonal autoshift to help you quickly recover an impaired application from failures in an Availability Zone (AZ).

2024/11/18 - cloudformation - 1 new api methods
Changes This release adds a new API, ListHookResults, that allows retrieving CloudFormation Hooks invocation results for hooks invoked during a create change set operation or Cloud Control API operation

2024/11/18 - connect - 2 new 5 updated api methods
Changes Adds CreateContactFlowVersion and ListContactFlowVersions APIs to create and view the versions of a contact flow.

2024/11/18 - profile - 13 new 6 updated api methods
Changes This release introduces Segmentation APIs and new Calculated Attribute Event Filters as part of Amazon Connect Customer Profiles service.

2024/11/18 - ec2 - 16 updated api methods
Changes Adding request and response elements for managed resources.

2024/11/18 - ecs - 5 updated api methods
Changes This release adds support for adding VPC Lattice configurations in ECS CreateService/UpdateService APIs. The configuration allows for associating VPC Lattice target groups with ECS Services.

2024/11/18 - iotsitewise - 6 new 6 updated api methods
Changes The release introduces a generative AI Assistant in AWS IoT SiteWise. It includes: 1) InvokeAssistant API - Invoke the Assistant to get alarm summaries and ask questions. 2) Dataset APIs - Manage knowledge base configuration for the Assistant. 3) Portal APIs enhancement - Manage AI-aware dashboards.

2024/11/18 - wisdom - 14 new api methods
Changes This release introduces MessageTemplate as a resource in Amazon Q in Connect, along with APIs to create, read, search, update, and delete MessageTemplate resources.

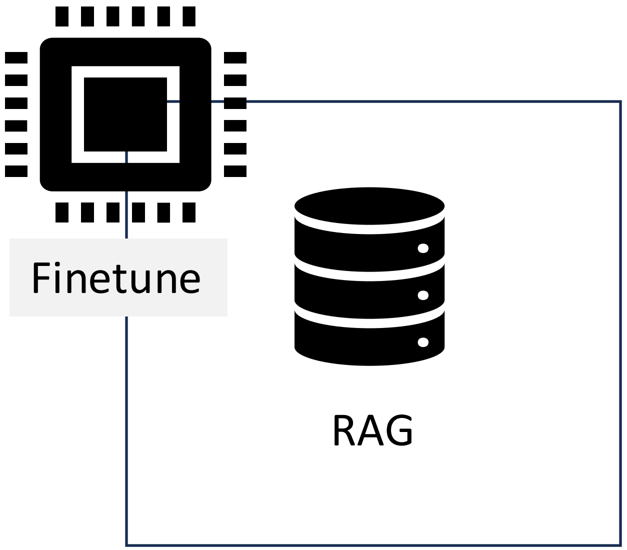
2024/11/18 - rds - 16 updated api methods
Changes Add support for the automatic pause/resume feature of Aurora Serverless v2.

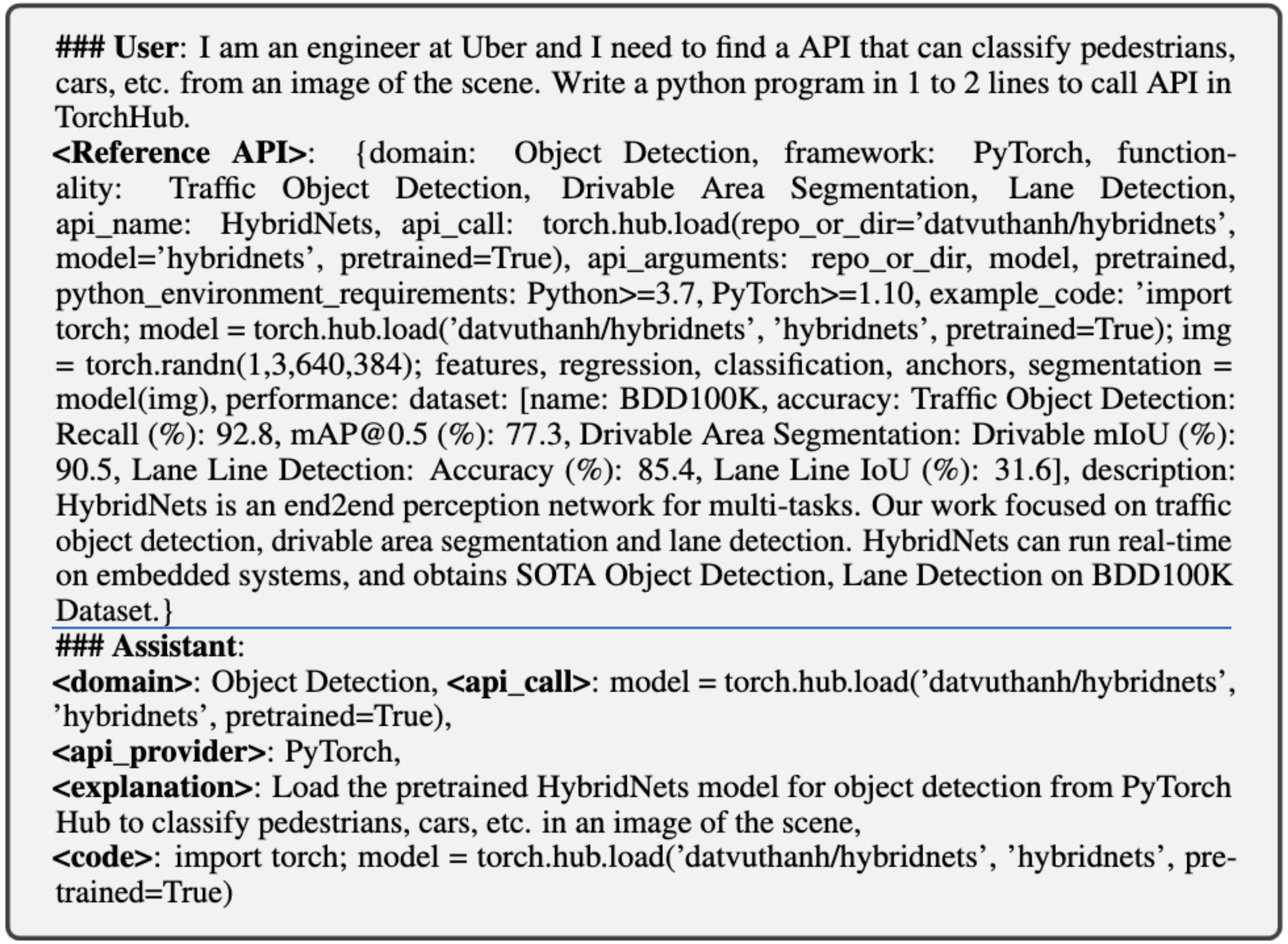
Big Idea: Retrieval Aware Training (RAT)

Fine-tune the model to **use** or ***ignore*** **retrieved context**.

- Introduce *correct* and *incorrect* **retrieval** results during **instruction** fine-tuning
- Ensures model is **robust to low-quality retrieval**

Retrieval Aware Training





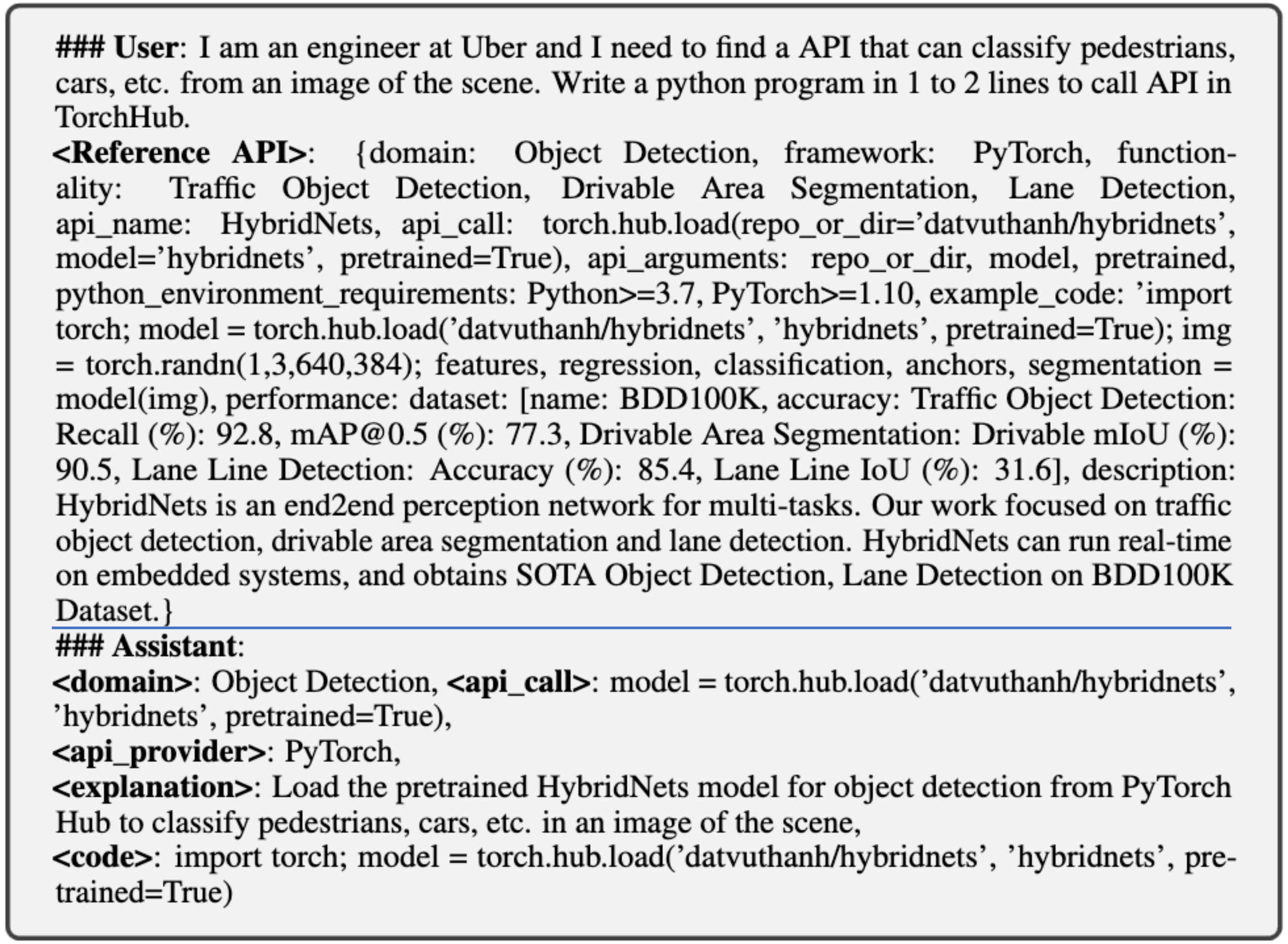
### User: I am an engineer at Uber and I need to find a API that can classify pedestrians, cars, etc. from an image of the scene. Write a python program in 1 to 2 lines to call API in TorchHub.
<Reference API>: {domain: Object Detection, framework: PyTorch, functionality: Traffic Object Detection, Drivable Area Segmentation, Lane Detection, api_name: HybridNets, api_call: torch.hub.load(repo_or_dir=’datvuthanh/hybridnets’, model=’hybridnets’, pretrained=True), api_arguments: repo_or_dir, model, pretrained, python_environment_requirements: Python>=3.7, PyTorch>=1.10, example_code: ’import torch; model = torch.hub.load(’datvuthanh/hybridnets’, ’hybridnets’, pretrained=True); img = torch.randn(1,3,640,384); features, regression, classification, anchors, segmentation = model(img), performance: dataset: [name: BDD100K, accuracy: Traffic Object Detection: Recall (%): 92.8, mAP@0.5 (%): 77.3, Drivable Area Segmentation: Drivable mIoU (%): 90.5, Lane Line Detection: Accuracy (%): 85.4, Lane Line IoU (%): 31.6], description: HybridNets is an end2end perception network for multi-tasks. Our work focused on traffic object detection, drivable area segmentation and lane detection. HybridNets can run real-time on embedded systems, and obtains SOTA Object Detection, Lane Detection on BDD100K Dataset.}

### Assistant:
<domain>: Object Detection, **<api_call>**: model = torch.hub.load(’datvuthanh/hybridnets’, ’hybridnets’, pretrained=True),
<api_provider>: PyTorch,
<explanation>: Load the pretrained HybridNets model for object detection from PyTorch Hub to classify pedestrians, cars, etc. in an image of the scene,
<code>: import torch; model = torch.hub.load(’datvuthanh/hybridnets’, ’hybridnets’, pretrained=True)

Retrieval Aware Training

Is this relevant?

Step 1: Determine to use or discard the retrieved API!



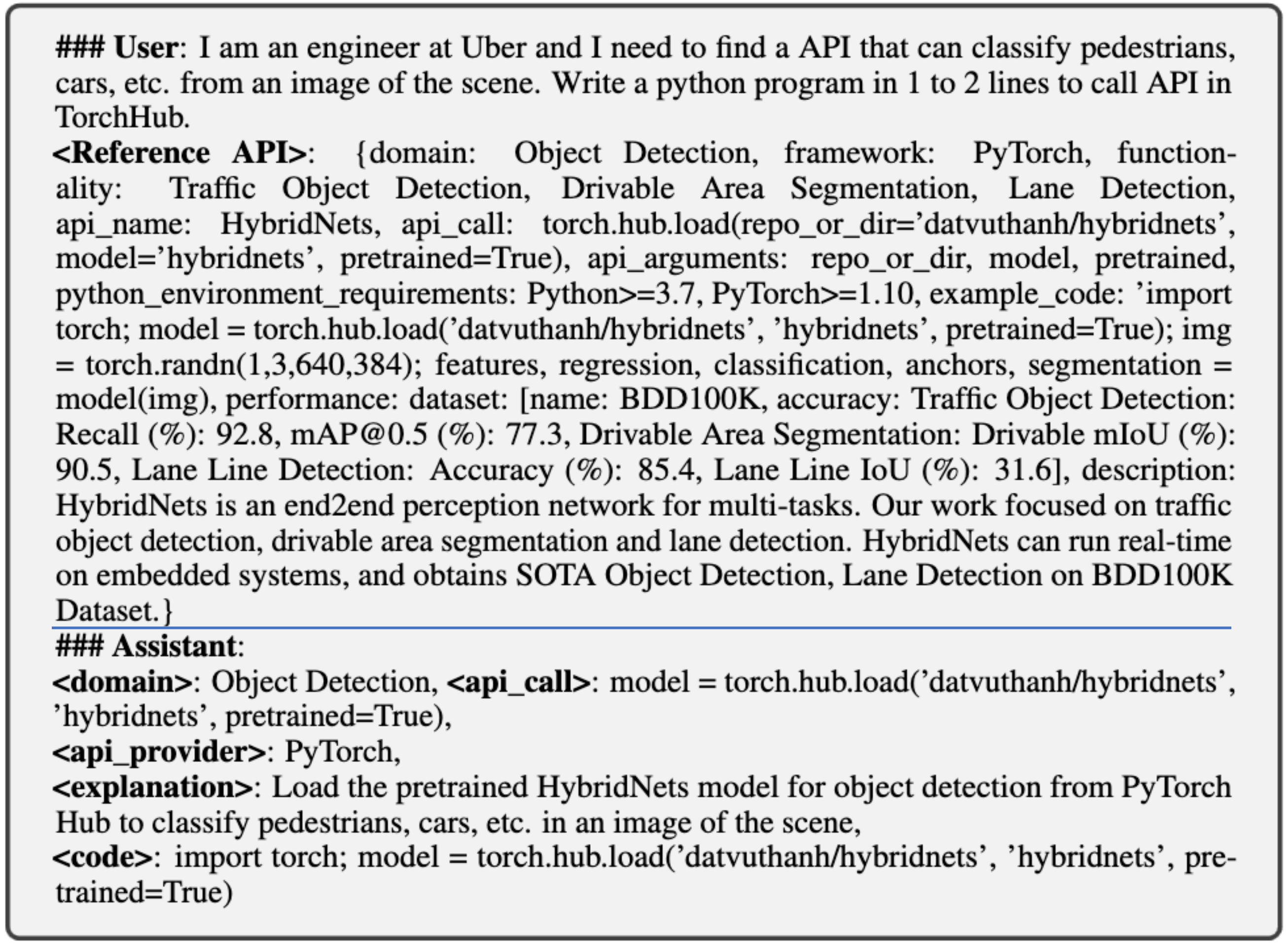
### User: I am an engineer at Uber and I need to find a API that can classify pedestrians, cars, etc. from an image of the scene. Write a python program in 1 to 2 lines to call API in TorchHub.
<Reference API>: {domain: Object Detection, framework: PyTorch, functionality: Traffic Object Detection, Drivable Area Segmentation, Lane Detection, api_name: HybridNets, api_call: torch.hub.load(repo_or_dir=’datvuthanh/hybridnets’, model=’hybridnets’, pretrained=True), api_arguments: repo_or_dir, model, pretrained, python_environment_requirements: Python>=3.7, PyTorch>=1.10, example_code: ’import torch; model = torch.hub.load(’datvuthanh/hybridnets’, ’hybridnets’, pretrained=True); img = torch.randn(1,3,640,384); features, regression, classification, anchors, segmentation = model(img), performance: dataset: [name: BDD100K, accuracy: Traffic Object Detection: Recall (%): 92.8, mAP@0.5 (%): 77.3, Drivable Area Segmentation: Drivable mIoU (%): 90.5, Lane Line Detection: Accuracy (%): 85.4, Lane Line IoU (%): 31.6], description: HybridNets is an end2end perception network for multi-tasks. Our work focused on traffic object detection, drivable area segmentation and lane detection. HybridNets can run real-time on embedded systems, and obtains SOTA Object Detection, Lane Detection on BDD100K Dataset.}

### Assistant:
<domain>: Object Detection, **<api_call>**: model = torch.hub.load(’datvuthanh/hybridnets’, ’hybridnets’, pretrained=True),
<api_provider>: PyTorch,
<explanation>: Load the pretrained HybridNets model for object detection from PyTorch Hub to classify pedestrians, cars, etc. in an image of the scene,
<code>: import torch; model = torch.hub.load(’datvuthanh/hybridnets’, ’hybridnets’, pretrained=True)

Retrieval Aware Training

Step 1: Determine to use or discard the retrieved API!

Step 2: Respond to the prompt!



### User: I am an engineer at Uber and I need to find a API that can classify pedestrians, cars, etc. from an image of the scene. Write a python program in 1 to 2 lines to call API in TorchHub.
<Reference API>: {domain: Object Detection, framework: PyTorch, functionality: Traffic Object Detection, Drivable Area Segmentation, Lane Detection, api_name: HybridNets, api_call: torch.hub.load(repo_or_dir='datvuthanh/hybridnets', model='hybridnets', pretrained=True), api_arguments: repo_or_dir, model, pretrained, python_environment_requirements: Python>=3.7, PyTorch>=1.10, example_code: 'import torch; model = torch.hub.load('datvuthanh/hybridnets', 'hybridnets', pretrained=True); img = torch.randn(1,3,640,384); features, regression, classification, anchors, segmentation = model(img), performance: dataset: [name: BDD100K, accuracy: Traffic Object Detection: Recall (%): 92.8, mAP@0.5 (%): 77.3, Drivable Area Segmentation: Drivable mIoU (%): 90.5, Lane Line Detection: Accuracy (%): 85.4, Lane Line IoU (%): 31.6], description: HybridNets is an end2end perception network for multi-tasks. Our work focused on traffic object detection, drivable area segmentation and lane detection. HybridNets can run real-time on embedded systems, and obtains SOTA Object Detection, Lane Detection on BDD100K Dataset.}

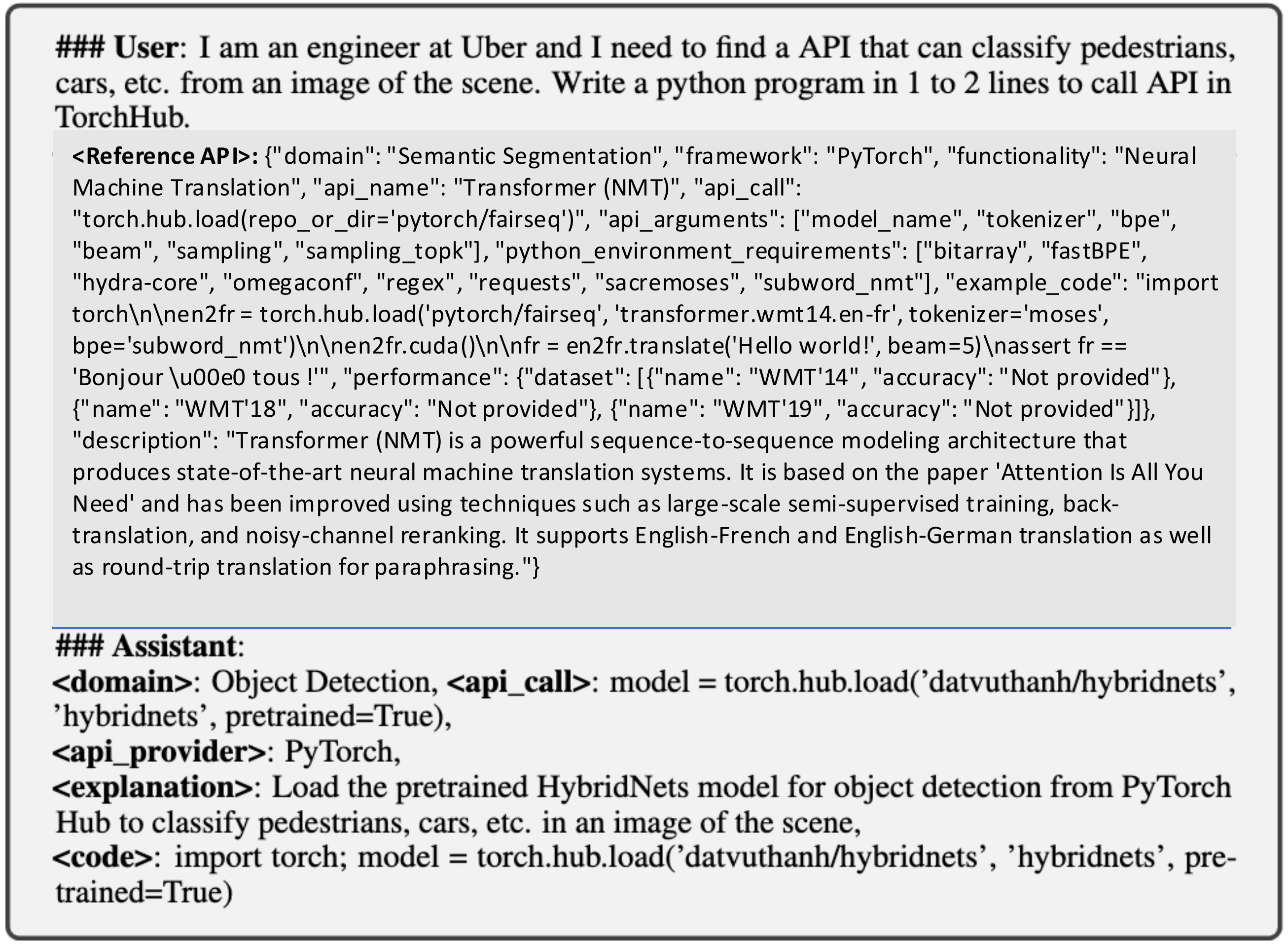
### Assistant:
<domain>: Object Detection, **<api_call>**: model = torch.hub.load('datvuthanh/hybridnets', 'hybridnets', pretrained=True),
<api_provider>: PyTorch,
<explanation>: Load the pretrained HybridNets model for object detection from PyTorch Hub to classify pedestrians, cars, etc. in an image of the scene,
<code>: import torch; model = torch.hub.load('datvuthanh/hybridnets', 'hybridnets', pretrained=True)

Retrieval Aware Training

Is this relevant?

Step 1: Determine to use or discard the retrieved API!

Step 2: Ignore the retrieved information!



### User: I am an engineer at Uber and I need to find a API that can classify pedestrians, cars, etc. from an image of the scene. Write a python program in 1 to 2 lines to call API in TorchHub.

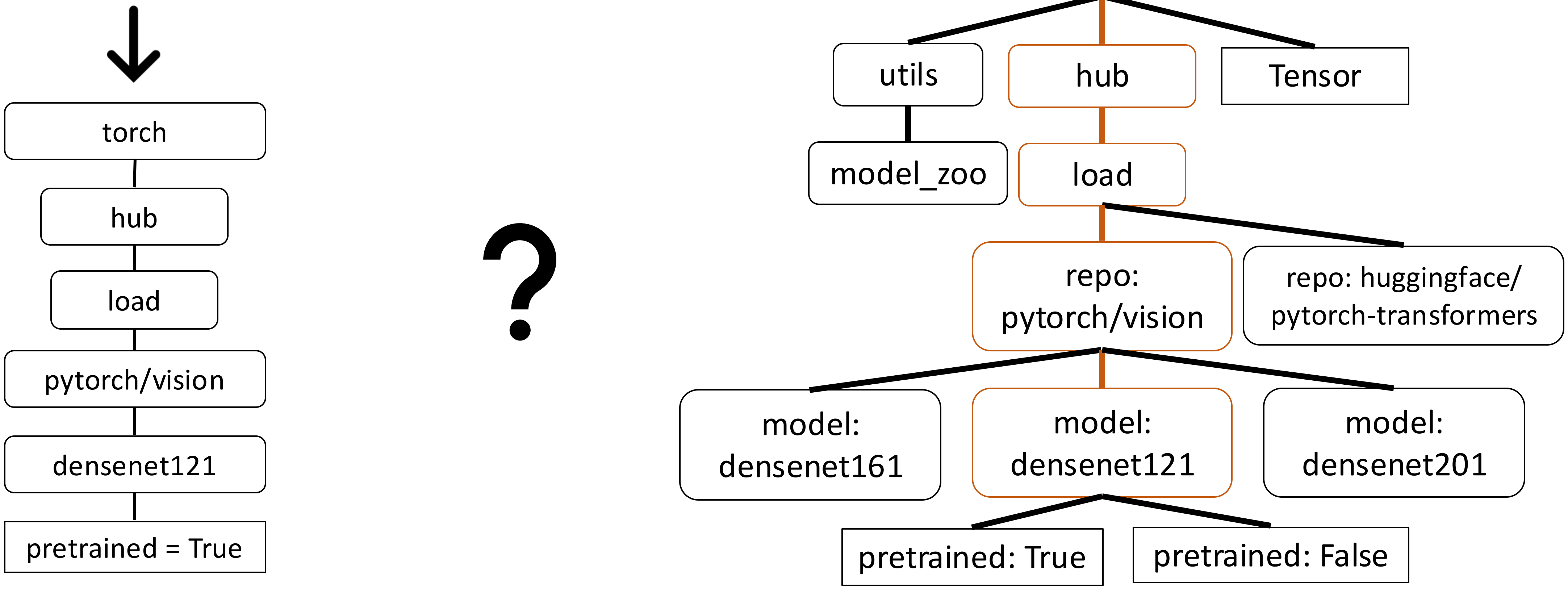
<Reference API>: {"domain": "Semantic Segmentation", "framework": "PyTorch", "functionality": "Neural Machine Translation", "api_name": "Transformer (NMT)", "api_call": "torch.hub.load(repo_or_dir='pytorch/fairseq')", "api_arguments": ["model_name", "tokenizer", "bpe", "beam", "sampling", "sampling_topk"], "python_environment_requirements": ["bitarray", "fastBPE", "hydra-core", "omegaconf", "regex", "requests", "sacremoses", "subword_nmt"], "example_code": "import torch\n\nen2fr = torch.hub.load('pytorch/fairseq', 'transformer.wmt14.en-fr', tokenizer='moses', bpe='subword_nmt')\n\nen2fr.cuda()\n\nfr = en2fr.translate('Hello world!', beam=5)\nassert fr == 'Bonjour \u00e0 tous !'", "performance": {"dataset": [{"name": "WMT'14", "accuracy": "Not provided"}, {"name": "WMT'18", "accuracy": "Not provided"}, {"name": "WMT'19", "accuracy": "Not provided"}]}, "description": "Transformer (NMT) is a powerful sequence-to-sequence modeling architecture that produces state-of-the-art neural machine translation systems. It is based on the paper 'Attention Is All You Need' and has been improved using techniques such as large-scale semi-supervised training, back-translation, and noisy-channel reranking. It supports English-French and English-German translation as well as round-trip translation for paraphrasing."}

### Assistant:
<domain>: Object Detection, **<api_call>**: model = torch.hub.load('datvuthanh/hybridnets', 'hybridnets', pretrained=True),
<api_provider>: PyTorch,
<explanation>: Load the pretrained HybridNets model for object detection from PyTorch Hub to classify pedestrians, cars, etc. in an image of the scene,
<code>: import torch; model = torch.hub.load('datvuthanh/hybridnets', 'hybridnets', pretrained=True)

Question: How much does GPT-4 hallucinate?

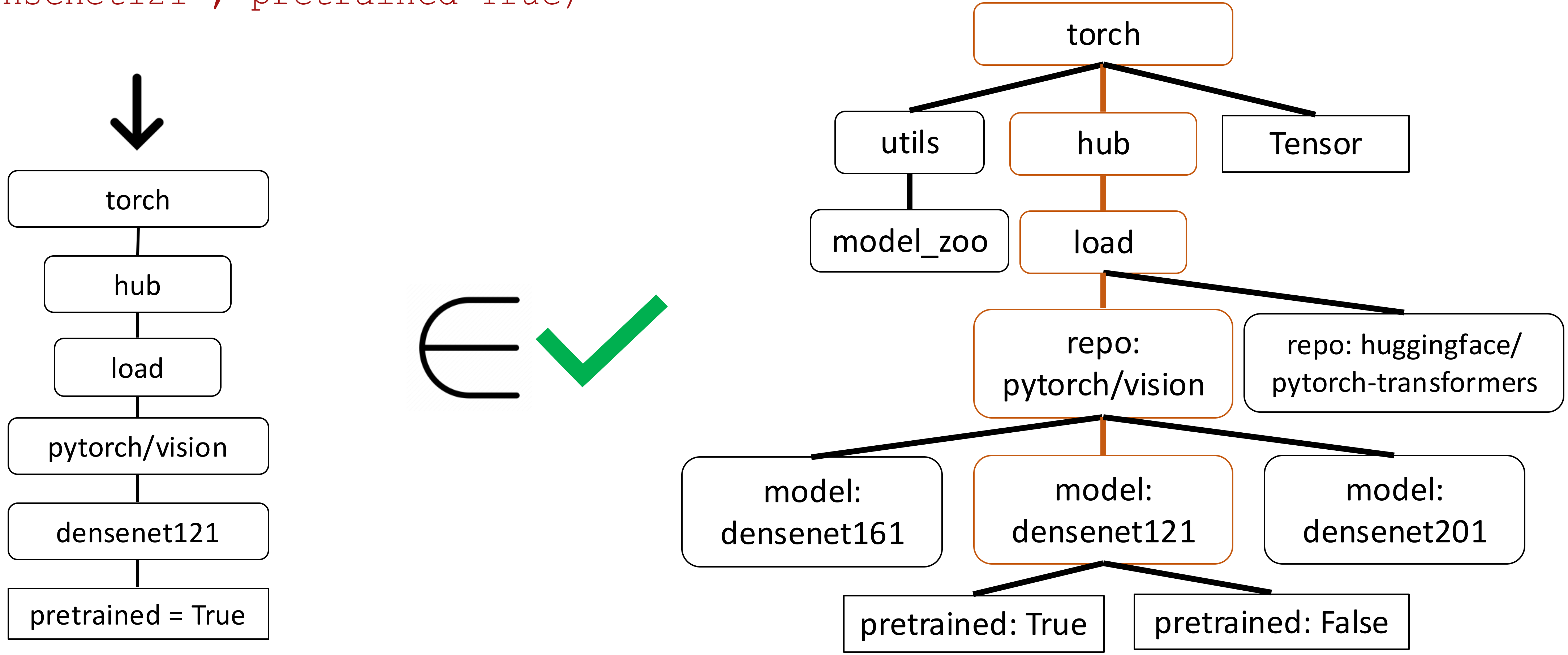
AST sub-tree matching - measure Hallucination

```
torch.hub.load('pytorch/vision:v0.10.0',
      'densenet121', pretrained=True)
```



AST sub-tree matching - measure Hallucination

```
torch.hub.load('pytorch/vision:v0.10.0',
        'densenet121', pretrained=True)
```

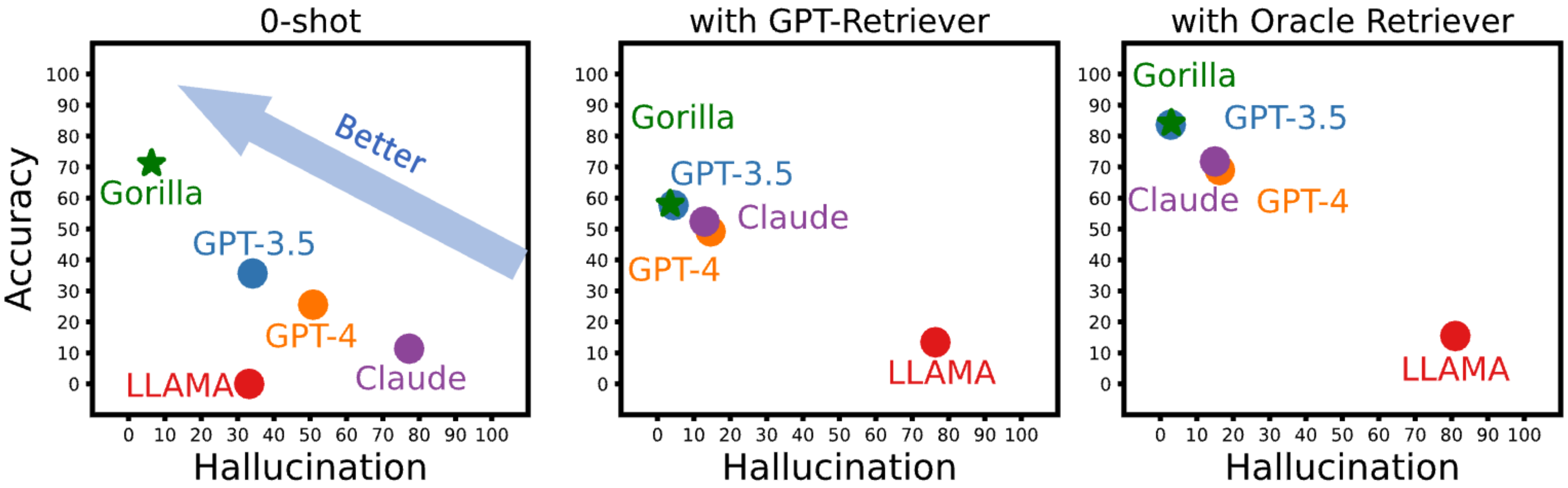


torch
hub
load
pytorch/vision
densenet121
pretrained = True

∈ ✓

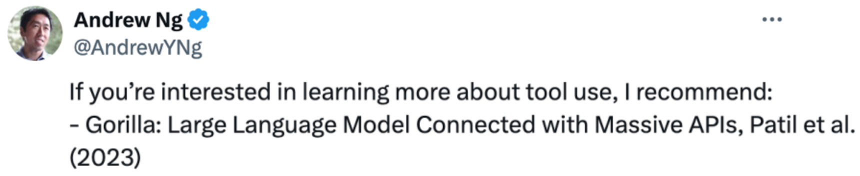
torch
utils
hub
Tensor
model_zoo
load
repo: pytorch/vision
repo: huggingface/ pytorch-transformers
model: densenet161
model: densenet121
model: densenet201
pretrained: True
pretrained: False

Overview of the Performance: accuracy vs. hallucination for all models

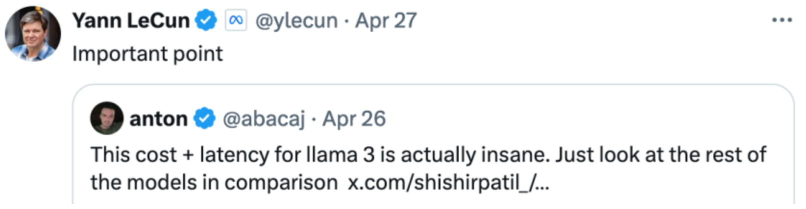


Used at:

Cisco NVIDIA Microsoft LinkedIn NETFLIX Adobe TESLA


Andrew Ng @AndrewYNg

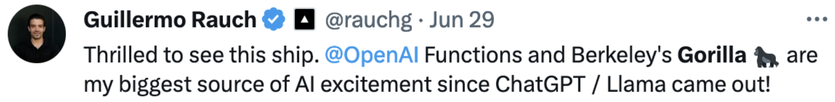
If you're interested in learning more about tool use, I recommend:
- Gorilla: Large Language Model Connected with Massive APIs, Patil et al. (2023)


Yann LeCun @ylecun · Apr 27

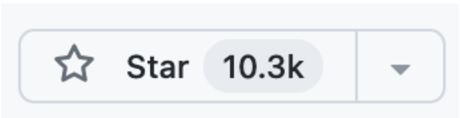
Important point

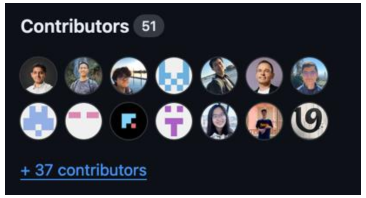
anton @abacaj · Apr 26

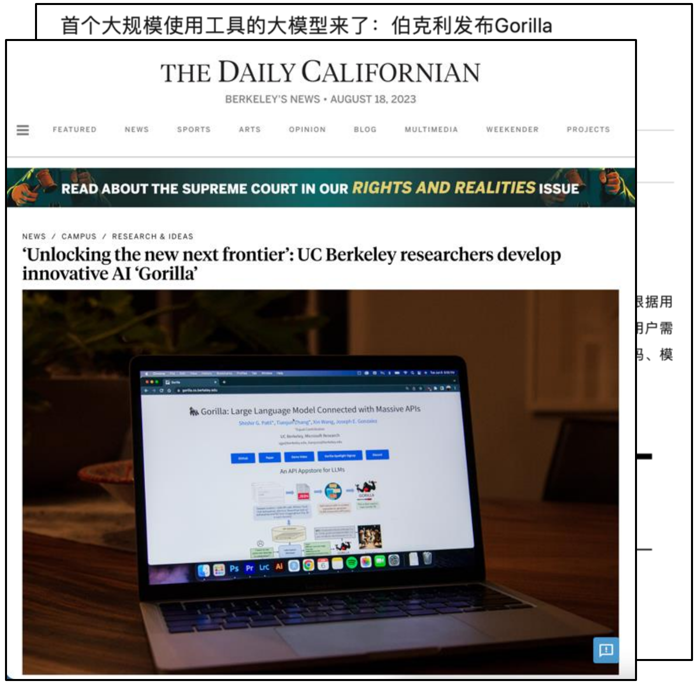
This cost + latency for llama 3 is actually insane. Just look at the rest of the models in comparison x.com/shishirpatil_/...


Guillermo Rauch @rauchg · Jun 29

Thrilled to see this ship. @OpenAI Functions and Berkeley's **Gorilla** 🦍 are my biggest source of AI excitement since ChatGPT / Llama came out!






首个大规模使用工具的大模型来了：伯克利发布Gorilla

THE DAILY CALIFORNIAN

BERKELEY'S NEWS • AUGUST 18, 2023

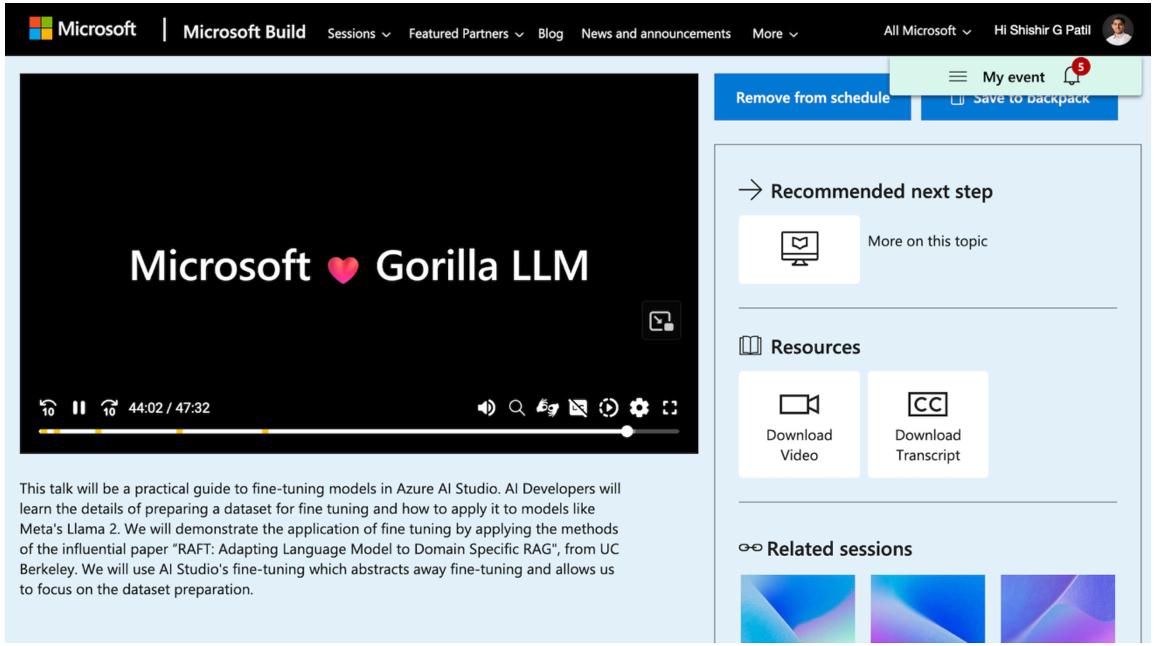
FEATURED
NEWS
SPORTS
ARTS
OPINION
BLOG
MULTIMEDIA
WEEKENDER
PROJECTS
READ ABOUT THE SUPREME COURT IN OUR RIGHTS AND REALITIES ISSUE
NEWS / CAMPUS / RESEARCH & IDEAS

'Unlocking the new next frontier': UC Berkeley researchers develop innovative AI 'Gorilla'

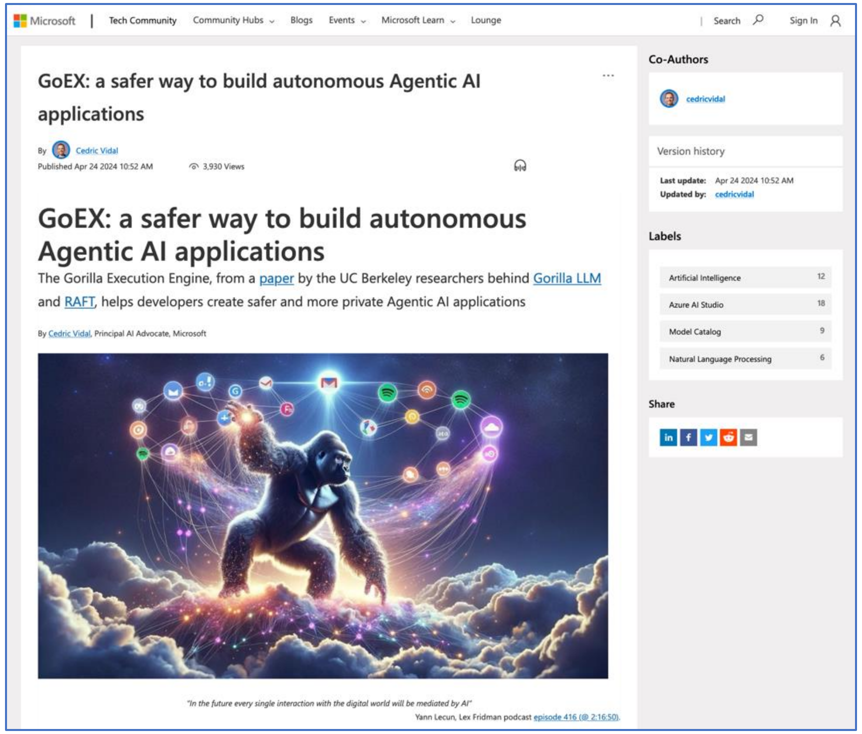
Gorilla: Large Language Model Connected with Massive APIs

Technical Collaboration and Pull Requests into Gorilla from:

Mistral AI Google DeepMind cohere ANTHROPIC



This talk will be a practical guide to fine-tuning models in Azure AI Studio. AI Developers will learn the details of preparing a dataset for fine tuning and how to apply it to models like Meta's Llama 2. We will demonstrate the application of fine tuning by applying the methods of the influential paper "RAFT: Adapting Language Model to Domain Specific RAG", from UC Berkeley. We will use AI Studio's fine-tuning which abstracts away fine-tuning and allows us to focus on the dataset preparation.



GoEX: a safer way to build autonomous Agentic AI applications

By Cedric Vidal
Published Apr 24 2024 10:52 AM
3,930 Views

Last update: Apr 24 2024 10:52 AM
Updated by: cedricvidal

Artificial Intelligence	12
Azure AI Studio	18
Model Catalog	9
Natural Language Processing	6

GoEX: a safer way to build autonomous Agentic AI applications

The Gorilla Execution Engine, from a paper by the UC Berkeley researchers behind Gorilla LLM and RAFT, helps developers create safer and more private Agentic AI applications

By Cedric Vidal, Principal AI Advocate, Microsoft

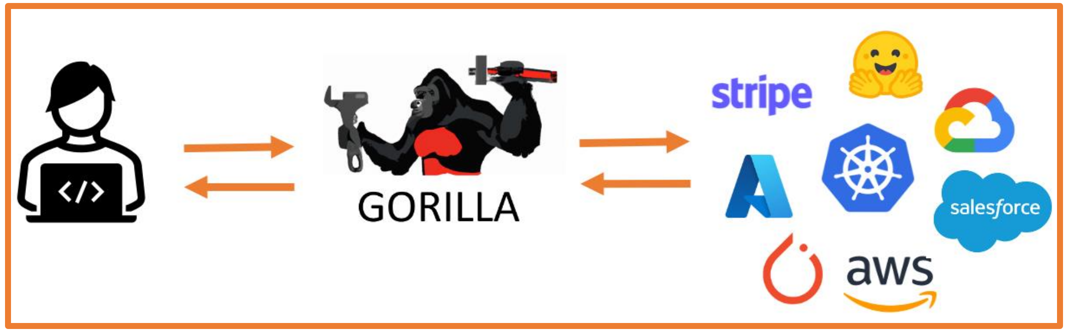
"In the future every single interaction with the digital world will be mediated by AI"

Yann Lecun, Lex Fridman podcast episode 416 (@ 2:16:50).

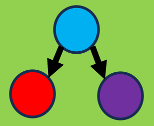
GORILLA

gorilla.cs.berkeley.edu

Retrieval Aware Training (RAT)



Measure Hallucination!



Agent-Arena

{ } Open Functions

Berkeley Function-Calling Leaderboard

Berkeley Function-Calling Leaderboard

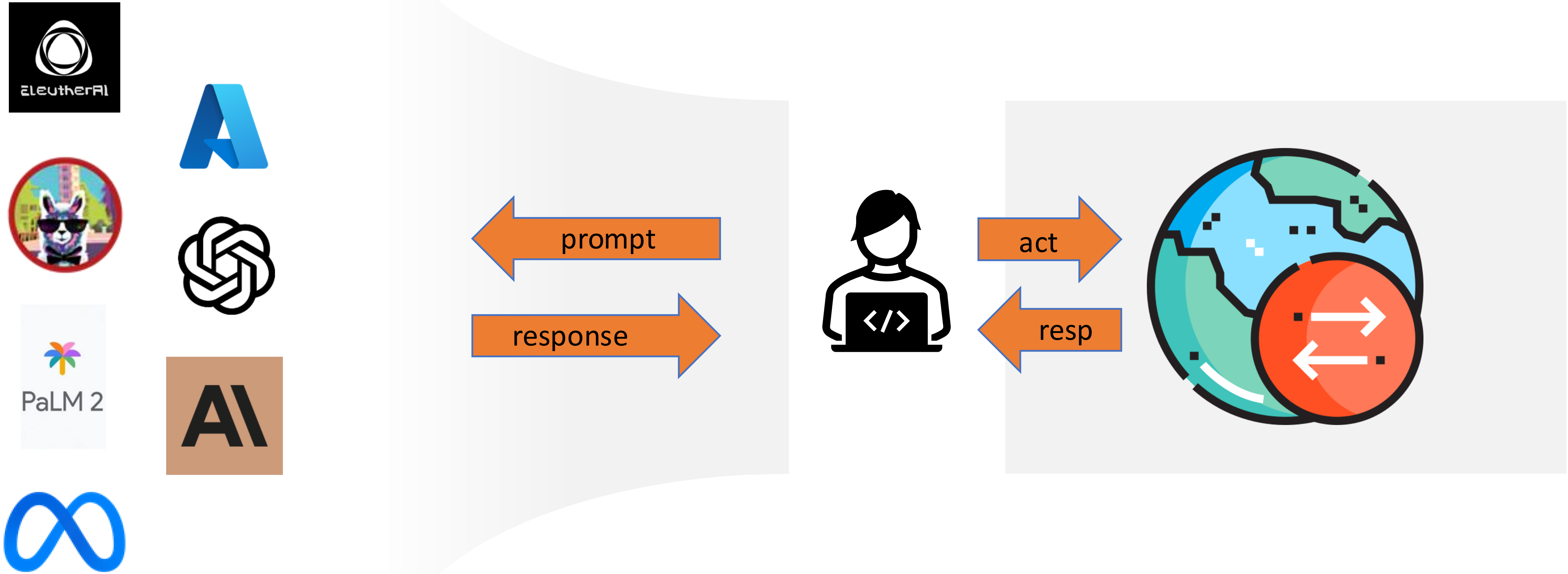
Leaderboard

GoEx: Execution Engine

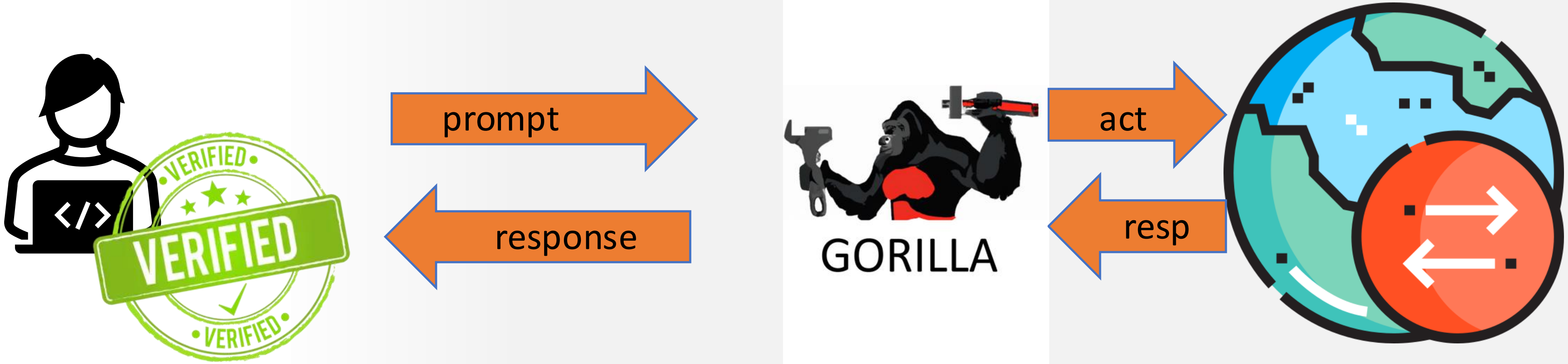
GoEx: Gorilla Execution Engine

A runtime to Execute actions that LLM generates

Users verify the *process*



Users verify the the *output*



prompt

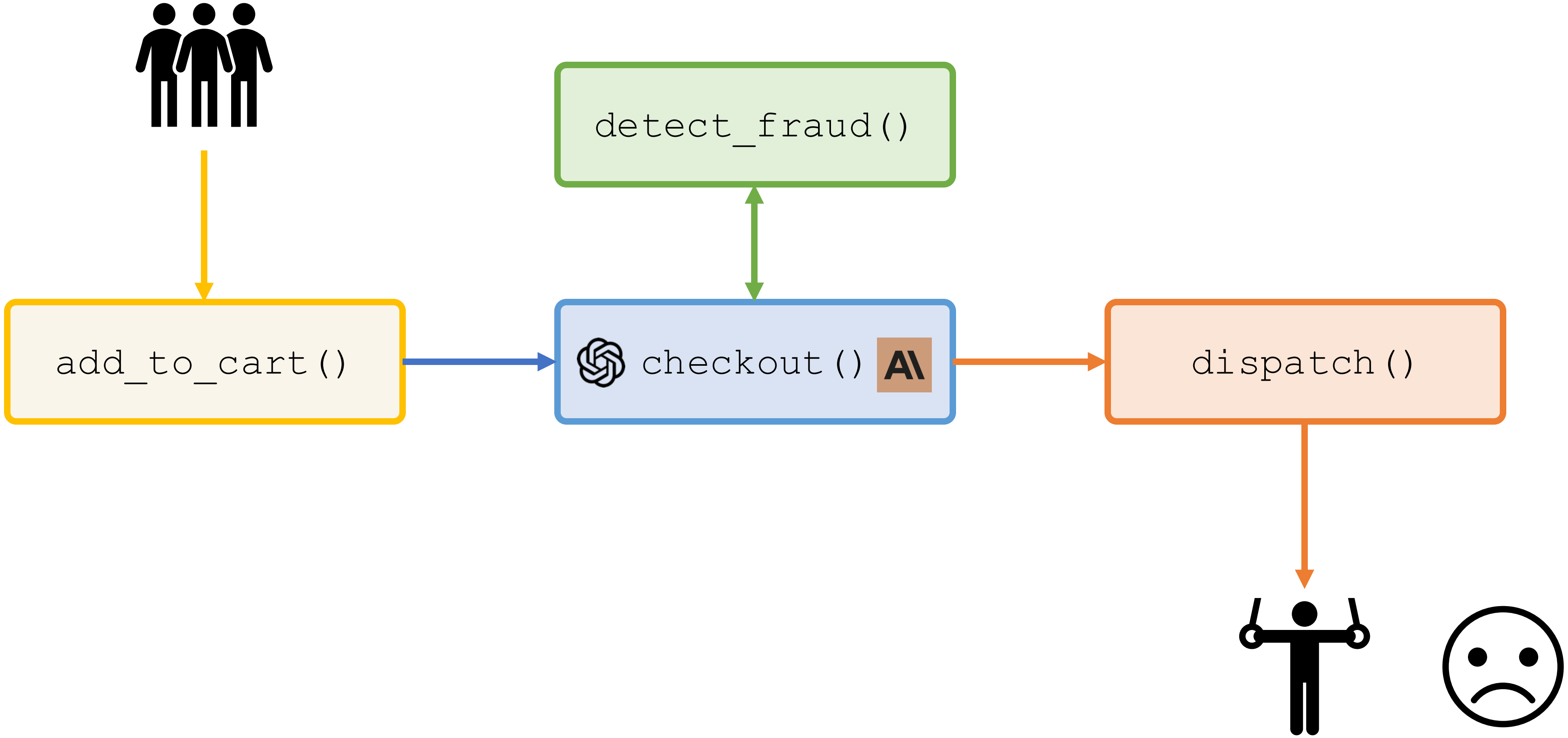
response

GORILLA

act

resp

Example: Amazon Shopping Cart



detect_fraud()

add_to_cart()

checkout()

dispatch()

Problems:

1. `Delayed` verification.

2. Only `downstream` outcome visible

Solution:

1. Guarantee `reversibility` for LLM actions

2. Bound the `blast-radius`

GoEx: A Runtime to Execute actions LLM generates!

“undo”

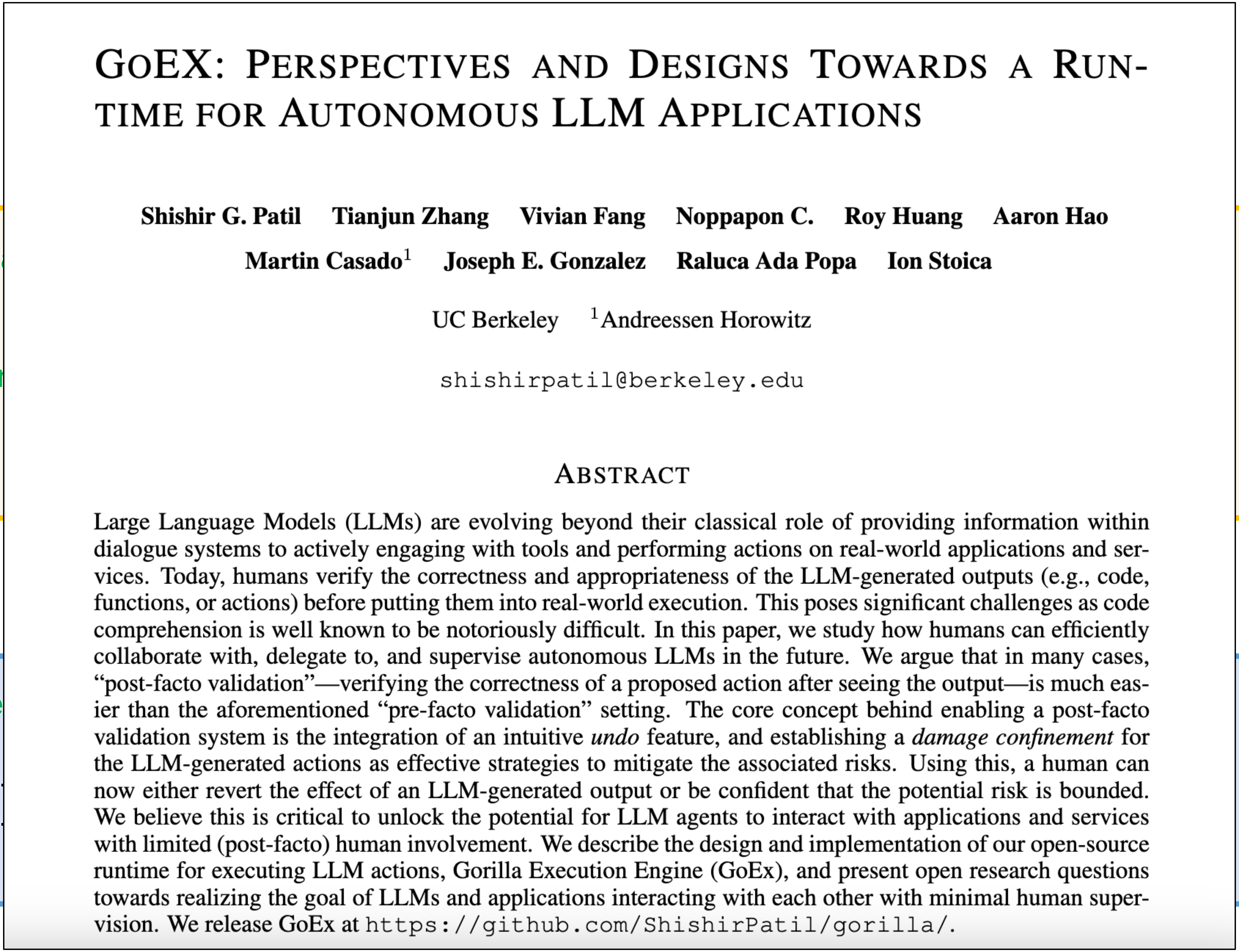
”blast-radius confinement”

`Abstractions` and `Policies` to enable LLMs to execute actions in the presence of delayed and downstream signals!

M

Sr

Whe



GoEX: Perspectives and Designs Towards a Runtime for Autonomous LLM Applications

Shishir G. Patil **Tianjun Zhang** **Vivian Fang** **Noppapon C.** **Roy Huang** **Aaron Hao**
Martin Casado[1] **Joseph E. Gonzalez** **Raluca Ada Popa** **Ion Stoica**

UC Berkeley [1]Andreessen Horowitz

shishirpatil@berkeley.edu

Abstract

Large Language Models (LLMs) are evolving beyond their classical role of providing information within dialogue systems to actively engaging with tools and performing actions on real-world applications and services. Today, humans verify the correctness and appropriateness of the LLM-generated outputs (e.g., code, functions, or actions) before putting them into real-world execution. This poses significant challenges as code comprehension is well known to be notoriously difficult. In this paper, we study how humans can efficiently collaborate with, delegate to, and supervise autonomous LLMs in the future. We argue that in many cases, "post-facto validation"—verifying the correctness of a proposed action after seeing the output—is much easier than the aforementioned "pre-facto validation" setting. The core concept behind enabling a post-facto validation system is the integration of an intuitive *undo* feature, and establishing a *damage confinement* for the LLM-generated actions as effective strategies to mitigate the associated risks. Using this, a human can now either revert the effect of an LLM-generated output or be confident that the potential risk is bounded. We believe this is critical to unlock the potential for LLM agents to interact with applications and services with limited (post-facto) human involvement. We describe the design and implementation of our open-source runtime for executing LLM actions, Gorilla Execution Engine (GoEx), and present open research questions towards realizing the goal of LLMs and applications interacting with each other with minimal human supervision. We release GoEx at `https://github.com/ShishirPatil/gorilla/`.

Can we build GoEx using tools available today?

Gorilla Execution Engine (GoEX)

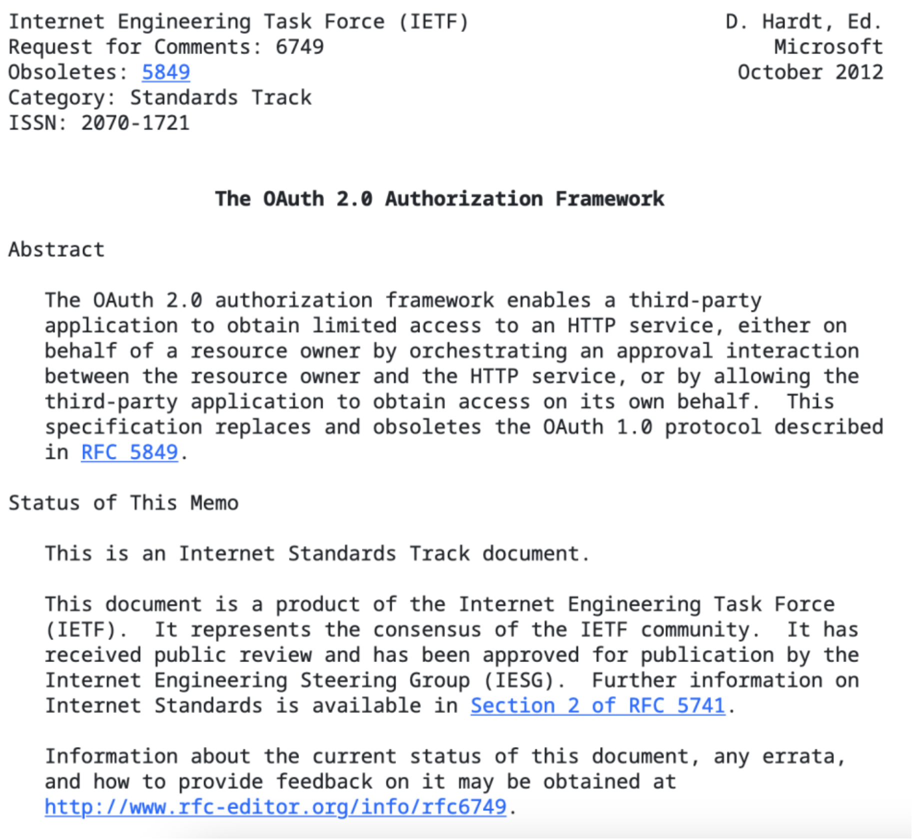
A *runtime* for LLMs

Two Components

- Identifying Minimal Permissions
  - Humas as ultimate-judge!!
  - Can we lower human dependence?
    - Self-consistency
- Execution Runtime
  - RESTful
  - Database
  - Filesystems

Identifying Minimal Permissions

Authentication and Scopes



Internet Engineering Task Force (IETF) D. Hardt, Ed.
Request for Comments: 6749 Microsoft
Obsoletes: 5849 October 2012
Category: Standards Track
ISSN: 2070-1721

The OAuth 2.0 Authorization Framework

Abstract

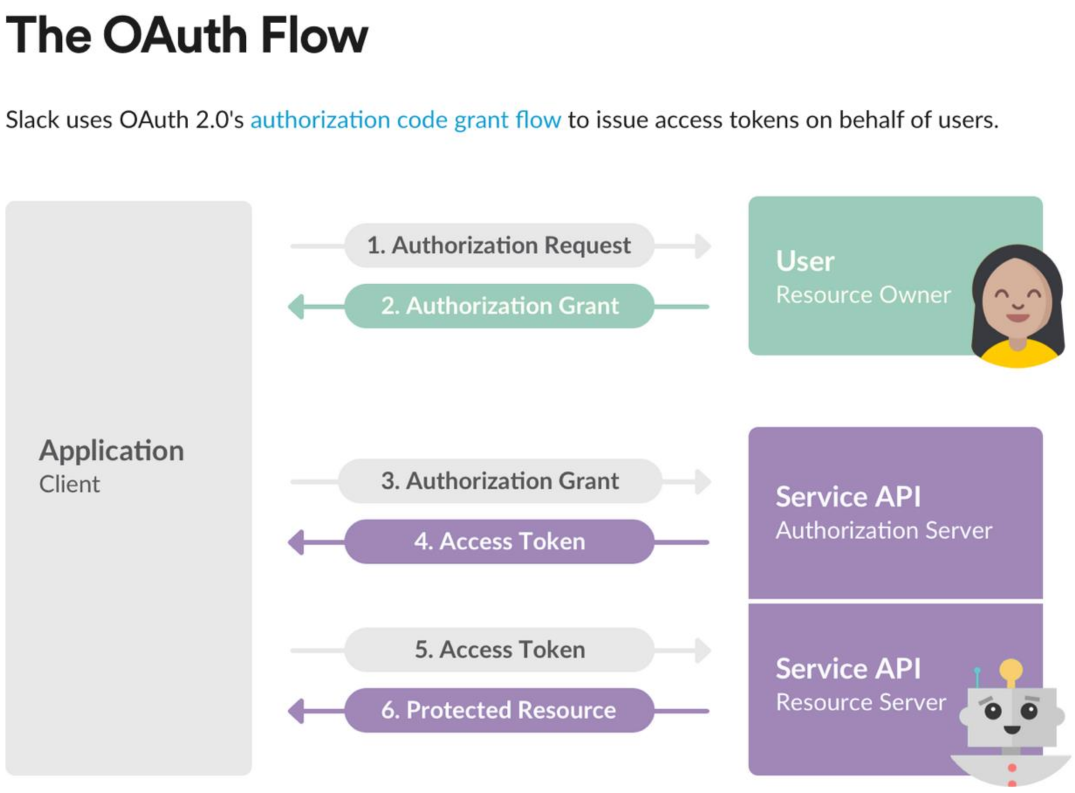
The OAuth 2.0 authorization framework enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its own behalf. This specification replaces and obsoletes the OAuth 1.0 protocol described in RFC 5849.

Status of This Memo

This is an Internet Standards Track document.

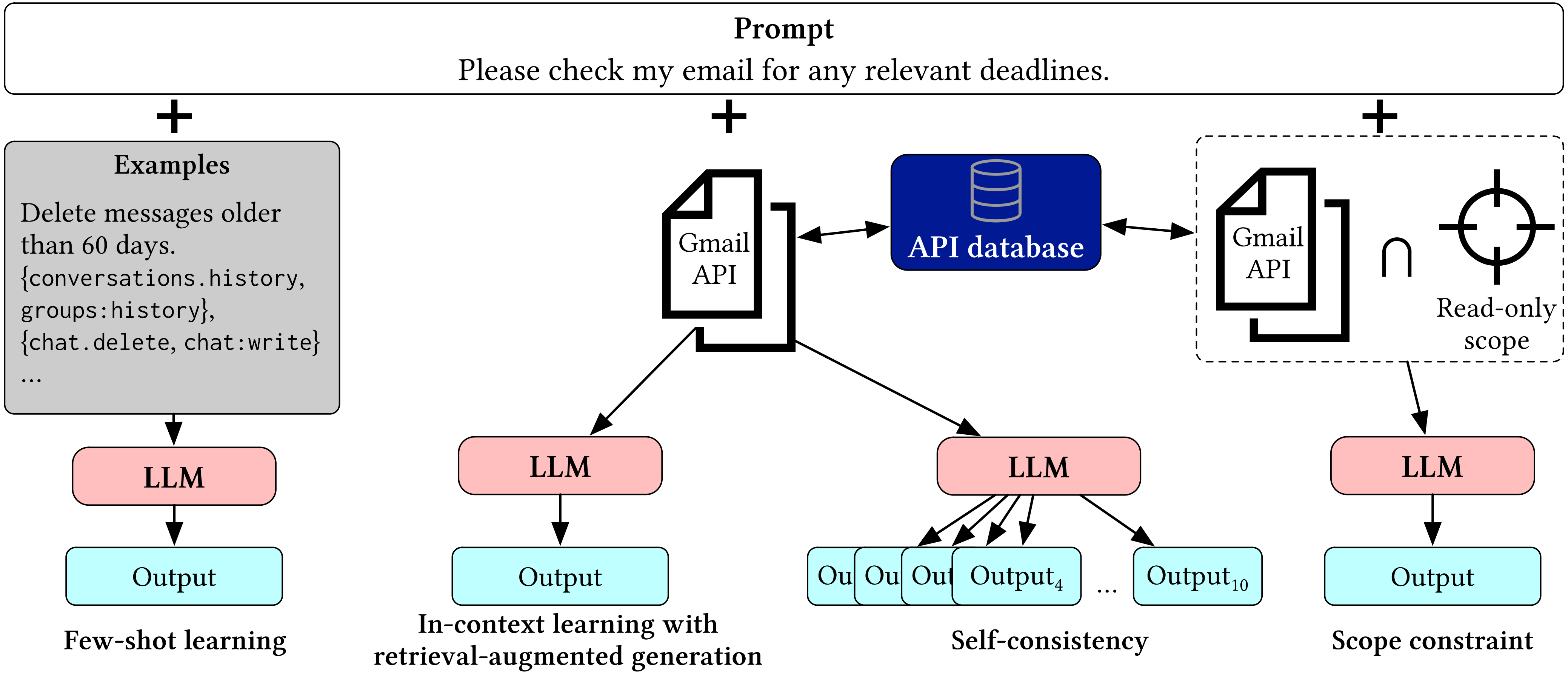
This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6749.



The OAuth Flow

Slack uses OAuth 2.0's authorization code grant flow to issue access tokens on behalf of users.



Prompt

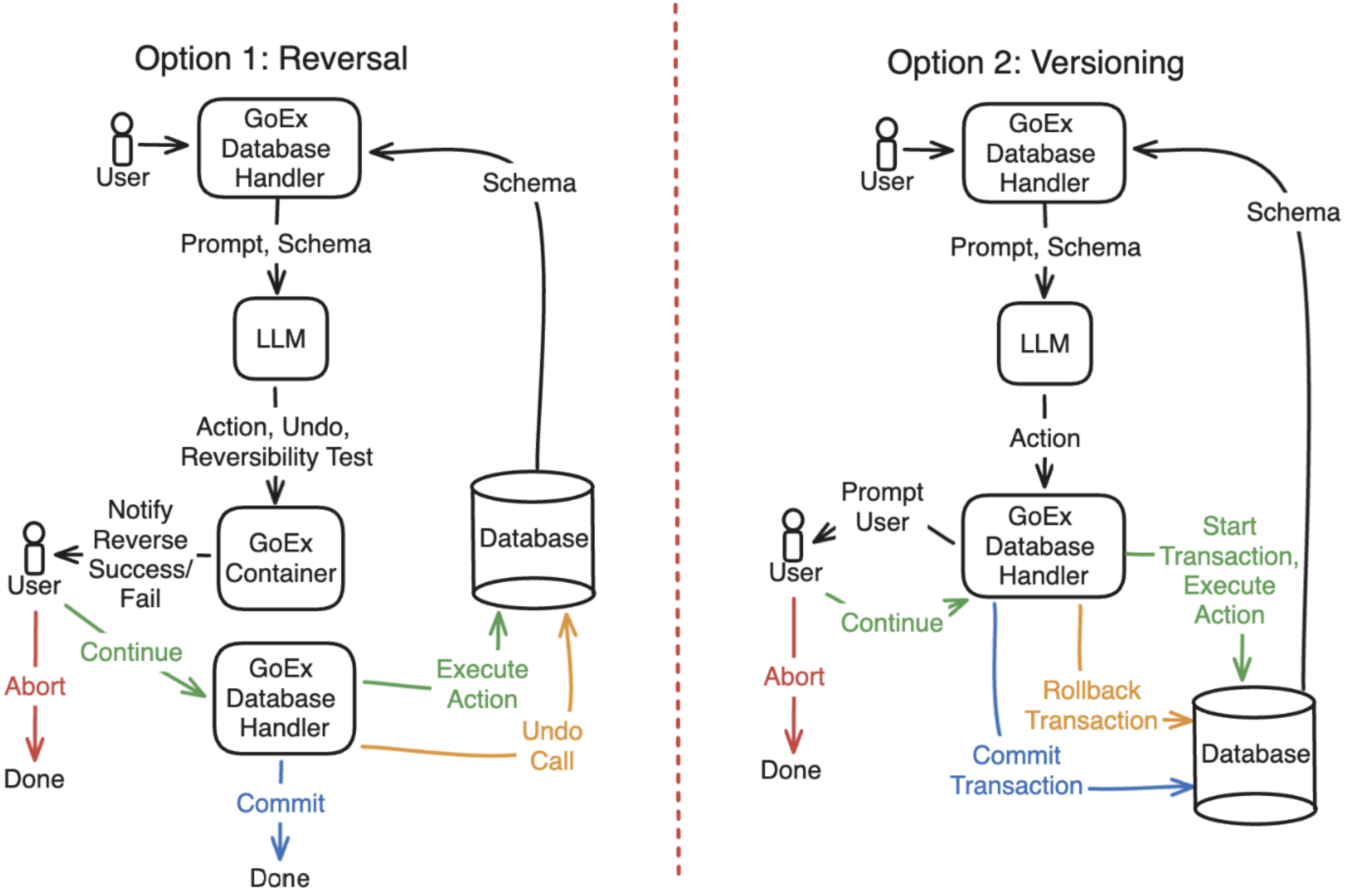
Please check my email for any relevant deadlines.

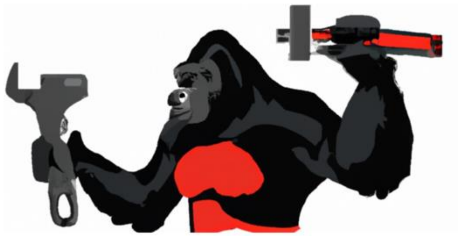
Examples

Delete messages older than 60 days.
{`conversations.history`, `groups:history`}, {`chat.delete`, `chat:write`}
…

Execution Runtime

GoEx: Database handlers

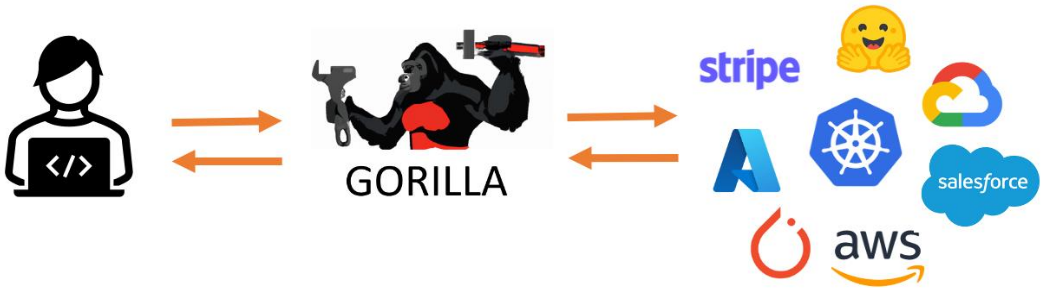




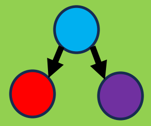
GORILLA

gorilla.cs.berkeley.edu

Retrieval Aware Training (RAT)



Measure Hallucination!



Agent-Arena

`{} Open Functions`

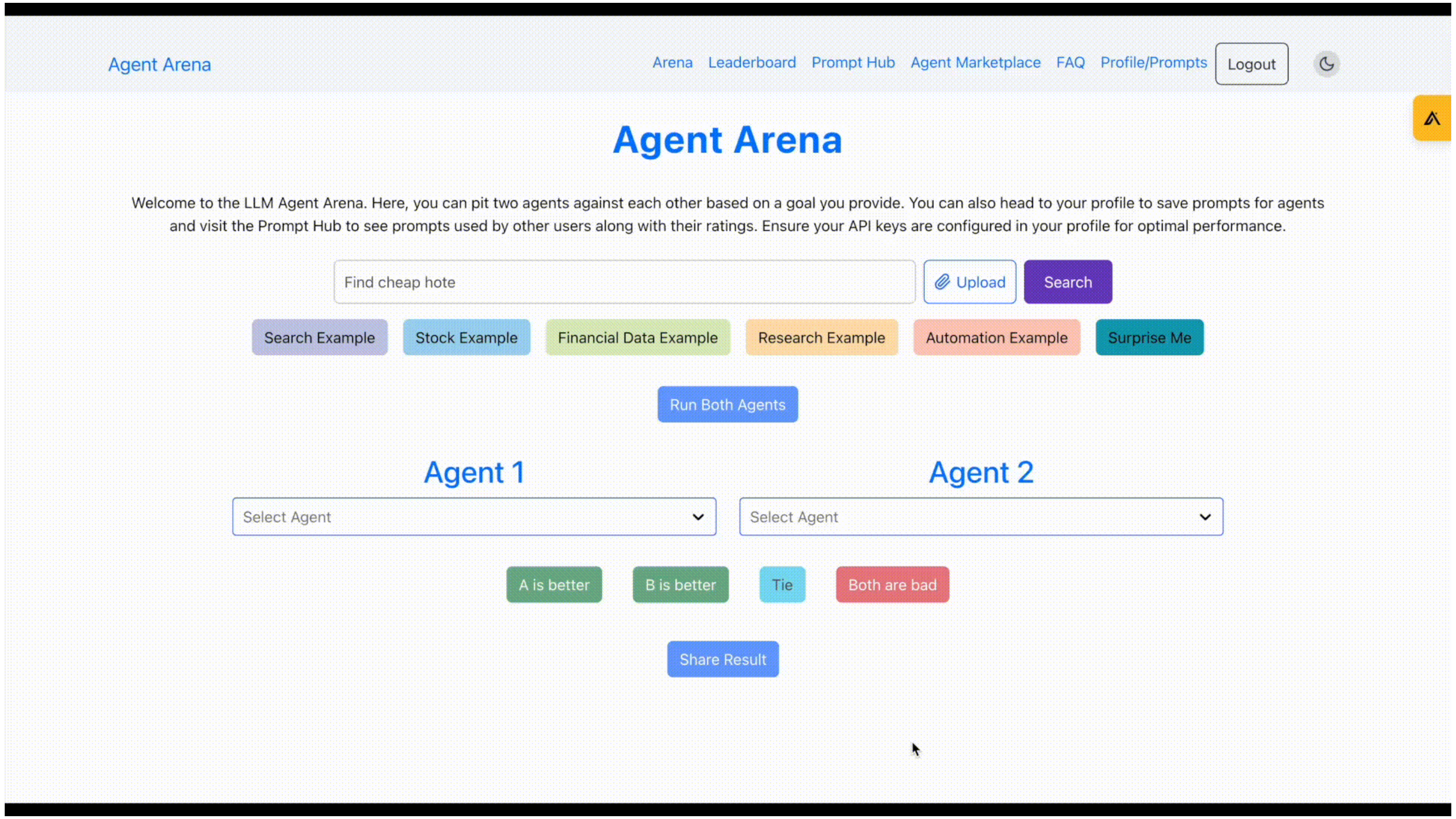
Berkeley Function-Calling Leaderboard

Berkeley Function-Calling Leaderboard

Leaderboard

GoEx: Execution Engine

Overview



Agent Arena

Welcome to the LLM Agent Arena. Here, you can pit two agents against each other based on a goal you provide. You can also head to your profile to save prompts for agents and visit the Prompt Hub to see prompts used by other users along with their ratings. Ensure your API keys are configured in your profile for optimal performance.

Find cheap hote

An interactive sandbox where users can compare, visualize, and rate agentic workflows **personalized to their needs**

What are agents?

LLM Model + Framework + Tool

Models / Frameworks / Tools

We currently support the following agents:

Frameworks 🔗

- **Langchain**
- **OpenAI Assistants**
- **Anthropic Tool Use**
- **CrewAI**
- **Llamaindex**
- **Composio**

Models 🧠

- **OpenAI:** gpt-4o-2024-08-06, gpt-4o-2024-05-13, gpt-4-turbo-2024-04-09, gpt-4-0613
- **Anthropic:** claude-3-5-sonnet-20240620, claude-3-opus-20240229, claude-3-haiku-20240307
- **Google:** gemini-1.5-pro-001, gemini-1.5-flash-001, gemini-1.5-pro-002, gemini-1.5-flash-002
- **Mistral:** open-mixtral-8x7b, mistral-large-2407, open-mixtral-8x22b
- **Meta:** llama-3.1-405B-instruct, llama-3.1-8B-instruct, llama-3.1-70B-instruct

Tools ⚙️

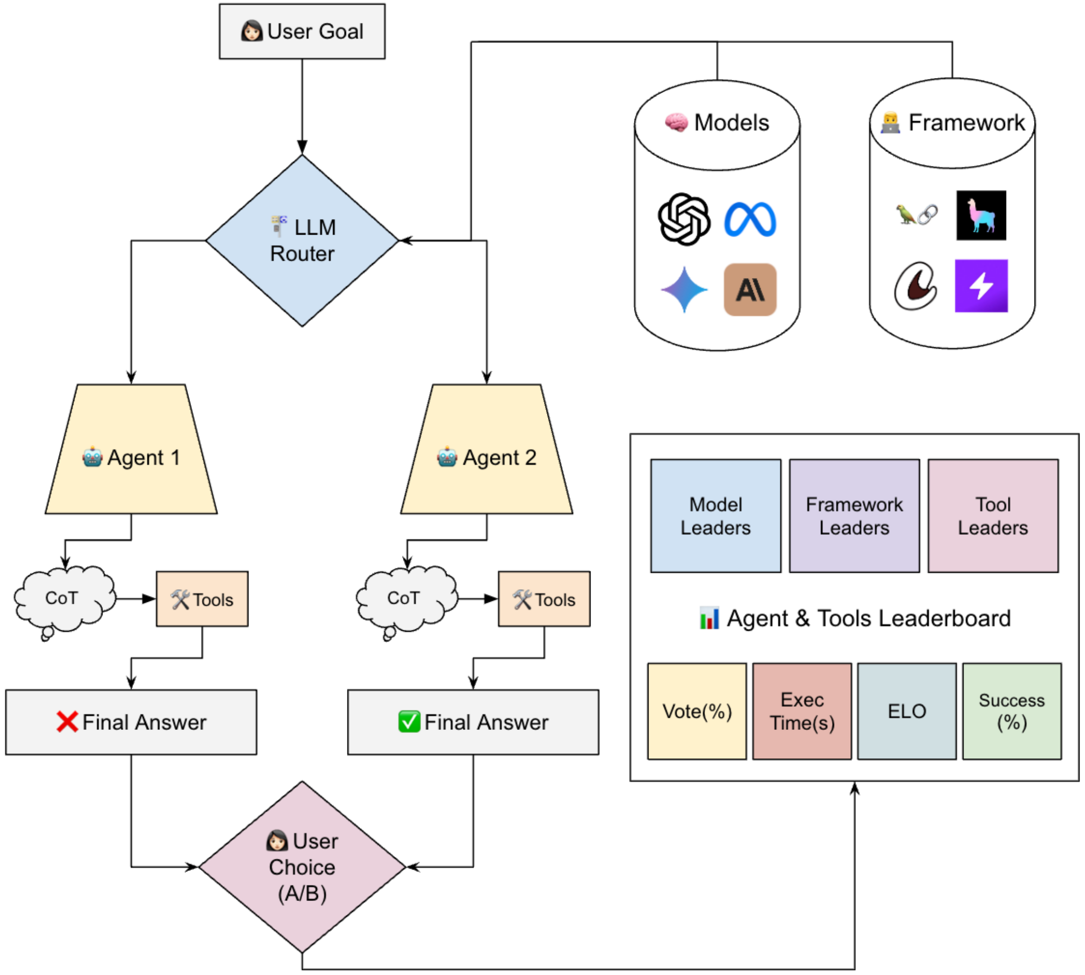
- **Brave Search**
- **Google Serper**
- **Alpha Vantage**
- **Google Jobs**
- **Arxiv**
- **+ 33 more**

The Agent Arena Platform

1. User gives task
2. Router finds 2 Agents
3. Agents attempt task
4. User declares winner

Components:

- Arena
- Leaderboard
- Prompt Hub



Introduction to the Bradley-Terry Model

What is the Bradley-Terry Model?

- The Bradley-Terry model is a statistical model used to estimate the probability that one entity will win over another in pairwise comparisons. It's particularly useful in ranking items (like agents, players, or teams) based on the outcomes of their head-to-head matchups.

Core Idea

- Each entity (e.g., an agent) is assigned a skill rating (β).The probability that Agent A beats Agent B depends on their respective skill ratings.

- **Formula**:
  - The probability that Agent A wins against Agent B (with β_A and β_B being the skill rating of Agent A and Agent B) is given by:

$$P(\text{A wins over B}) = \frac{e^{\beta_A}}{e^{\beta_A} + e^{\beta_B}}$$

A higher β means a stronger agent.

Bradley-Terry Model Example

langchain google-serper-search agent (claude-3-5-sonnet-20240620)

VS

llamaindex brave-search agent (gpt-4o-2024-08-06)

- Framework: langchain
- Tool: google-serper-search
- Model: (claude-3-5-sonnet-20240620)

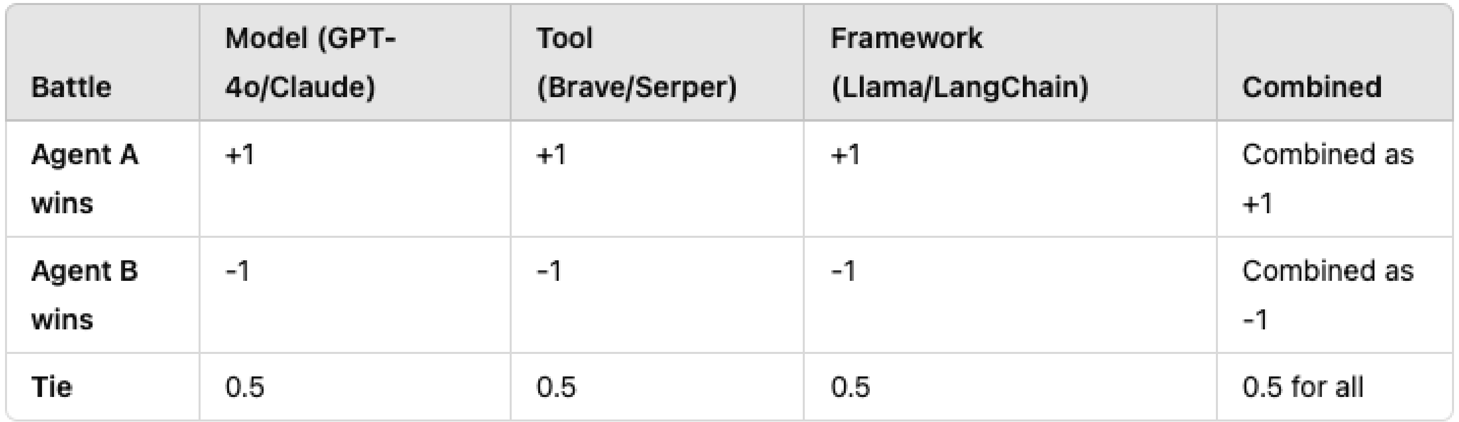
- Framework: llamaindex
- Tool: brave-search
- Model: (gpt-4o-2024-08-06)

$$L = -\sum_{i=1}^{n} [Y_i \log(\sigma(X_i \beta)) + (1 - Y_i) \log(1 - \sigma(X_i \beta))]$$

Next Step: Setting Up Features and Parameters

Extending Bradley-Terry (Combining Subcomponents)

- **Motivation for Extension:**
  - In some cases, agents are not monolithic entities but are composed of subcomponents that contribute to their performance. For example, in Agent Arena:
- **Models:** Different language models (e.g., GPT-4, Claude)
- **Tools:** Different tools or plugins the agents can use (e.g., search engines).
- **Frameworks:** Different agent frameworks (e.g., LangChain, LlamaIndex).
- We want to estimate the skill ratings of these subcomponents together, not just individually with separate design matrices.
- **Extended Model:**
  - Instead of assigning a single β per agent, assign βs to each subcomponent.
  - The overall skill of an agent is the combination of its subcomponents.



Battle	Model (GPT-4o/Claude)	Tool (Brave/Serper)	Framework (Llama/LangChain)	Combined
Agent A wins	+1	+1	+1	Combined as +1
Agent B wins	-1	-1	-1	Combined as -1
Tie	0.5	0.5	0.5	0.5 for all

Example Cont… (Setting Up Skill Parameter)

β: The Parameter Vector: β is a vector of parameters, with each entry corresponding to the "strength" or "rating" of a particular subcomponent (model, tool, or framework). This vector is what we optimize during model training to minimize the loss function and find the best-fit ratings.

Extended Approach (Combining subcomponents into one battle)

$$\beta = \begin{bmatrix} \text{skill}_{\text{brave}} & \text{skill}_{\text{google}} & \text{skill}_{\text{gpt-4o}} & \text{skill}_{\text{claude}} & \text{skill}_{\text{llama}} & \text{skill}_{\text{langchain}} \end{bmatrix}$$

Traditional Approach: Treat Agent, Framework, Models, and Tools as separate battles

$$\beta = [\text{skill}_{\text{brave}}, \text{skill}_{\text{google}}]$$

Individual Tool Battle

$$\beta = [\text{skill}_{\text{gpt-4o}}, \text{skill}_{\text{claude}}]$$

Individual Model Battle

$$\beta = [\text{skill}_{\text{llama}}, \text{skill}_{\text{langchain}}]$$

Individual Framework Battle

Interpreting the Results

Subcomponent Ratings:

- Suppose after fitting the model, we obtain:
- **Models:**
    - GPT-4: β = 1.5
    - Claude: β = 0.5
- **Tools:**
    - Brave Search: β = 1.0
    - Google Serper: β = 0.8
- **Frameworks:**
    - LangChain: β = 1.2
    - LlamaIndex: β = 0.9

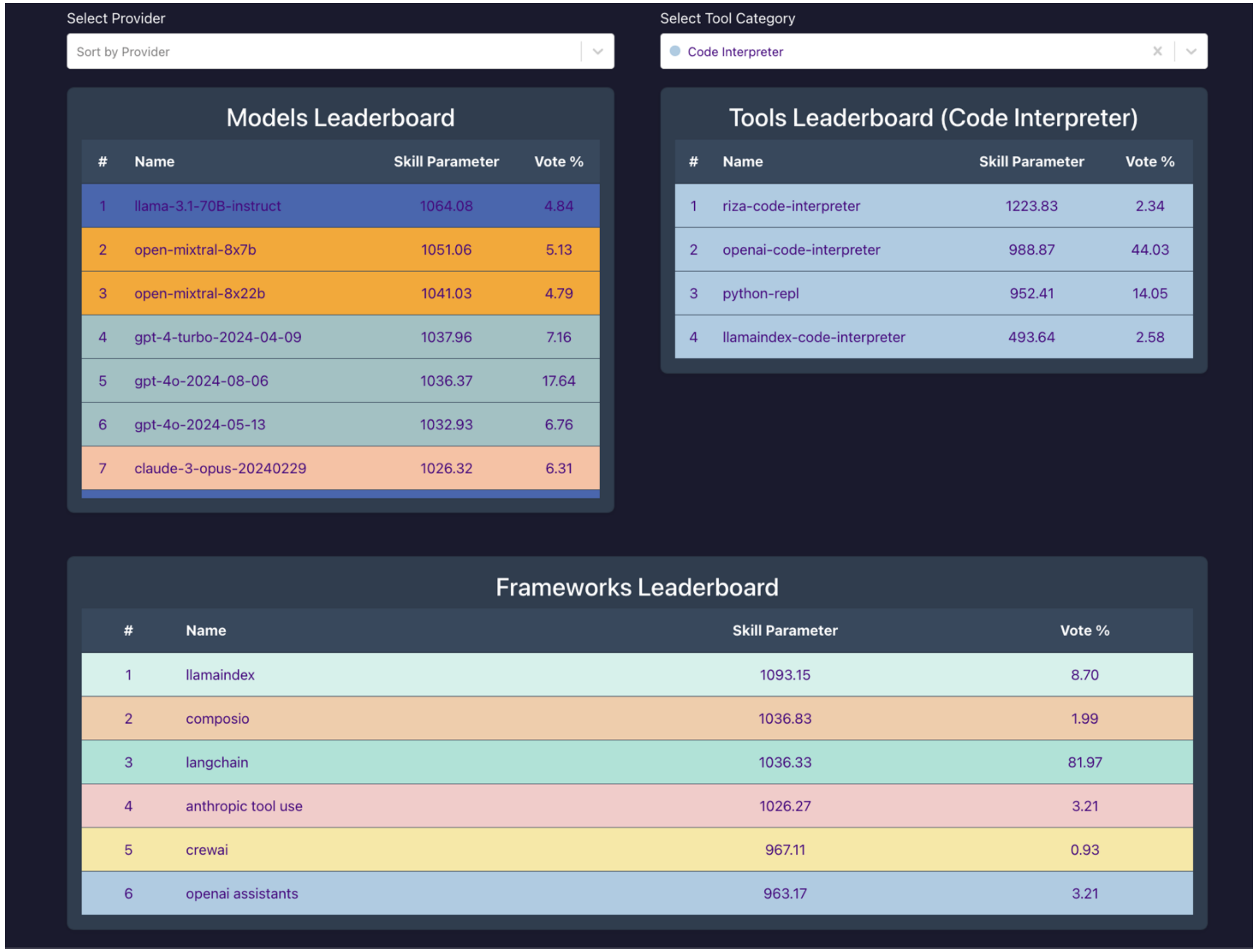
Interpretation:

- **Models:** GPT-4 has a higher rating than Claude, suggesting it contributes more to winning.
- **Tools:** Brave Search and Google Serper have similar ratings, but Brave Search is slightly higher.
- **Frameworks:** LangChain has a higher rating than LlamaIndex.

Predicting Outcomes:

- For any future battle, we can compute P(A wins over B) using the βs of their subcomponents.
- Agents with stronger subcomponents are more likely to win

Leaderboard



Select Provider: Sort by Provider

Select Tool Category: Code Interpreter

Models Leaderboard

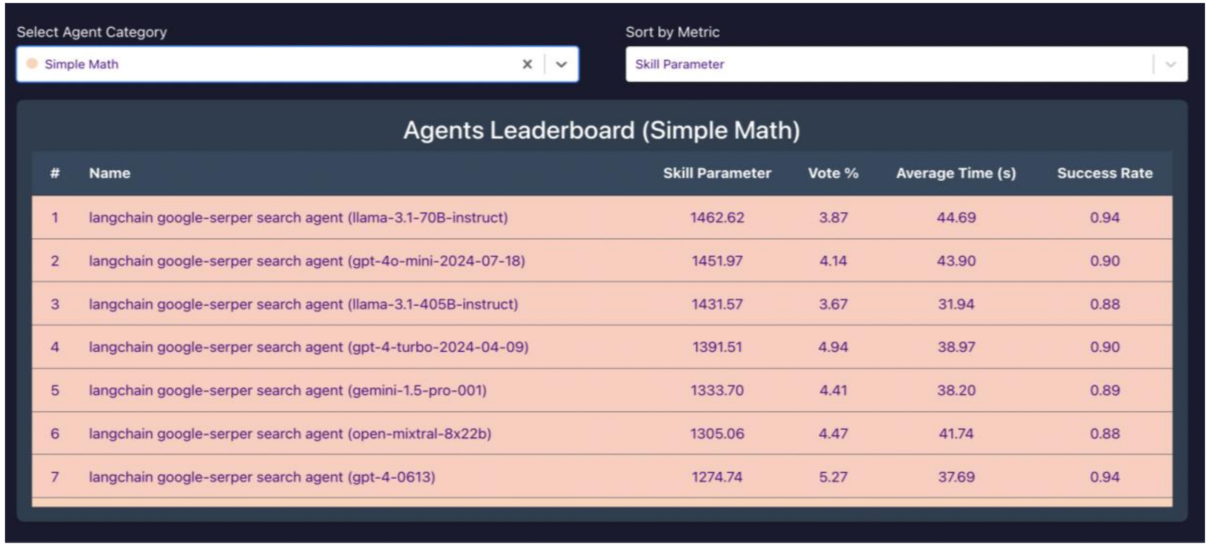
#	Name	Skill Parameter	Vote %
1	llama-3.1-70B-instruct	1064.08	4.84
2	open-mixtral-8x7b	1051.06	5.13
3	open-mixtral-8x22b	1041.03	4.79
4	gpt-4-turbo-2024-04-09	1037.96	7.16
5	gpt-4o-2024-08-06	1036.37	17.64
6	gpt-4o-2024-05-13	1032.93	6.76
7	claude-3-opus-20240229	1026.32	6.31

Tools Leaderboard (Code Interpreter)

#	Name	Skill Parameter	Vote %
1	riza-code-interpreter	1223.83	2.34
2	openai-code-interpreter	988.87	44.03
3	python-repl	952.41	14.05
4	llamaindex-code-interpreter	493.64	2.58

Frameworks Leaderboard

#	Name	Skill Parameter	Vote %
1	llamaindex	1093.15	8.70
2	composio	1036.83	1.99
3	langchain	1036.33	81.97
4	anthropic tool use	1026.27	3.21
5	crewai	967.11	0.93
6	openai assistants	963.17	3.21



Select Agent Category: Simple Math

Sort by Metric: Skill Parameter

Agents Leaderboard (Simple Math)

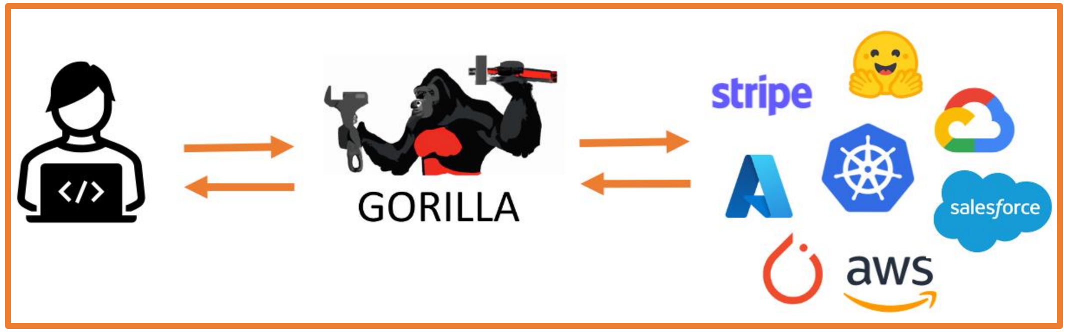
#	Name	Skill Parameter	Vote %	Average Time (s)	Success Rate
1	langchain google-serper search agent (llama-3.1-70B-instruct)	1462.62	3.87	44.69	0.94
2	langchain google-serper search agent (gpt-4o-mini-2024-07-18)	1451.97	4.14	43.90	0.90
3	langchain google-serper search agent (llama-3.1-405B-instruct)	1431.57	3.67	31.94	0.88
4	langchain google-serper search agent (gpt-4-turbo-2024-04-09)	1391.51	4.94	38.97	0.90
5	langchain google-serper search agent (gemini-1.5-pro-001)	1333.70	4.41	38.20	0.89
6	langchain google-serper search agent (open-mixtral-8x22b)	1305.06	4.47	41.74	0.88
7	langchain google-serper search agent (gpt-4-0613)	1274.74	5.27	37.69	0.94

Leaderboards rating model, framework, and tool subcomponents as well as composite agents using Bradley Terry Elo rating methods

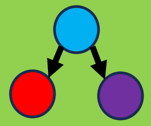
GORILLA

gorilla.cs.berkeley.edu

Retrieval Aware Training (RAT)



Measure Hallucination!



Agent-Arena

{} Open Functions

Berkeley Function-Calling Leaderboard

Berkeley Function-Calling Leaderboard

Leaderboard

GoEx: Execution Engine