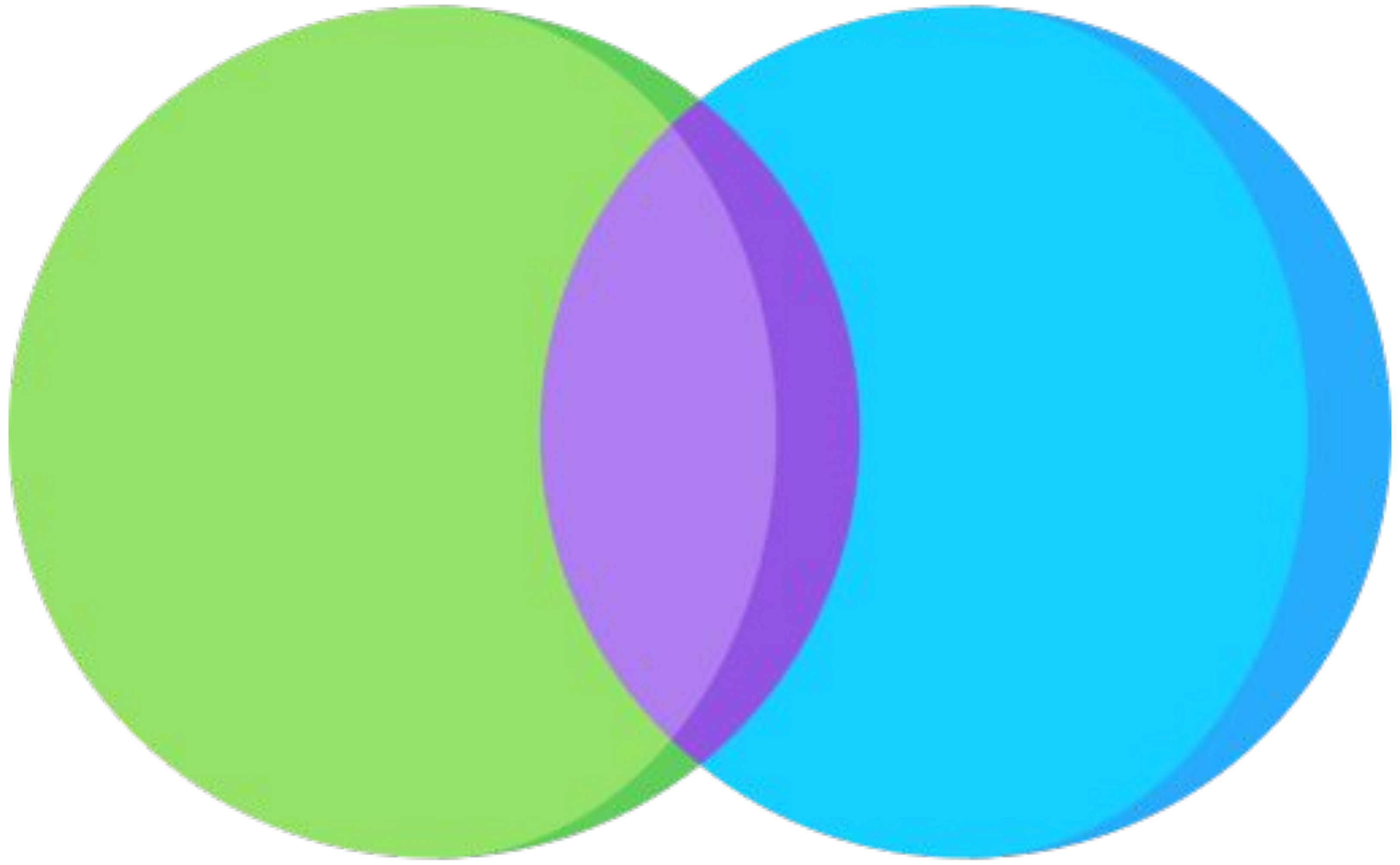
# Human-Centered Evaluation of Coding Agents

Valerie Chen

# Brief intro 👋

## My research



ML/NLP

HCI

I am a student at…

Carnegie Mellon University

ML
MACHINE LEARNING DEPARTMENT

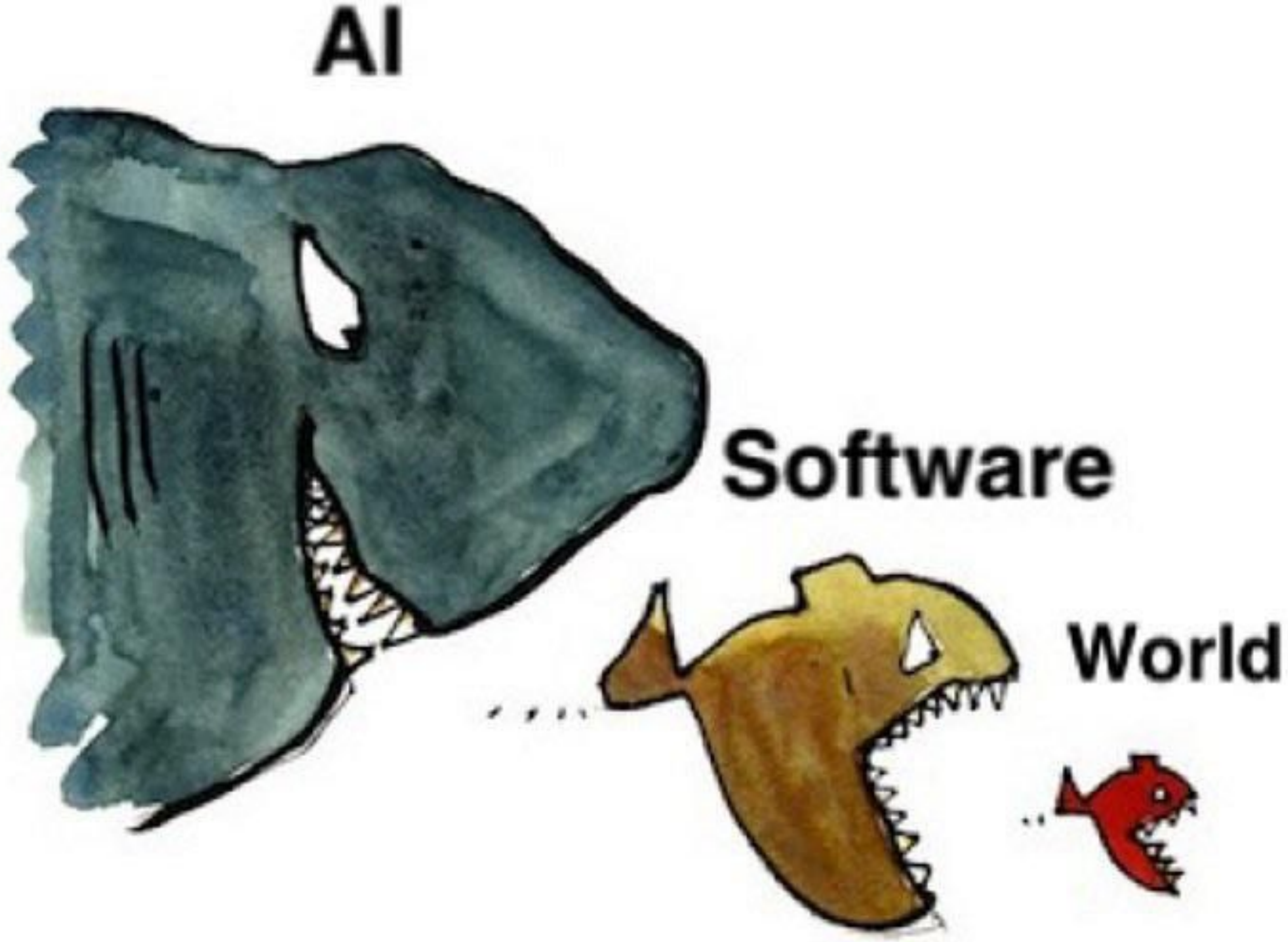
I collaborate with…

JET BRAINS

# “Software is eating the world”

Marc Andreessen (2011)

# Recently, a proliferation of AI tools for code



**A ripe opportunity to study human-AI interaction!**

# Coding assistants are evolving

# A growing ecosystem

GitHub Copilot

ChatGPT

Claude

BY ANTHROPIC

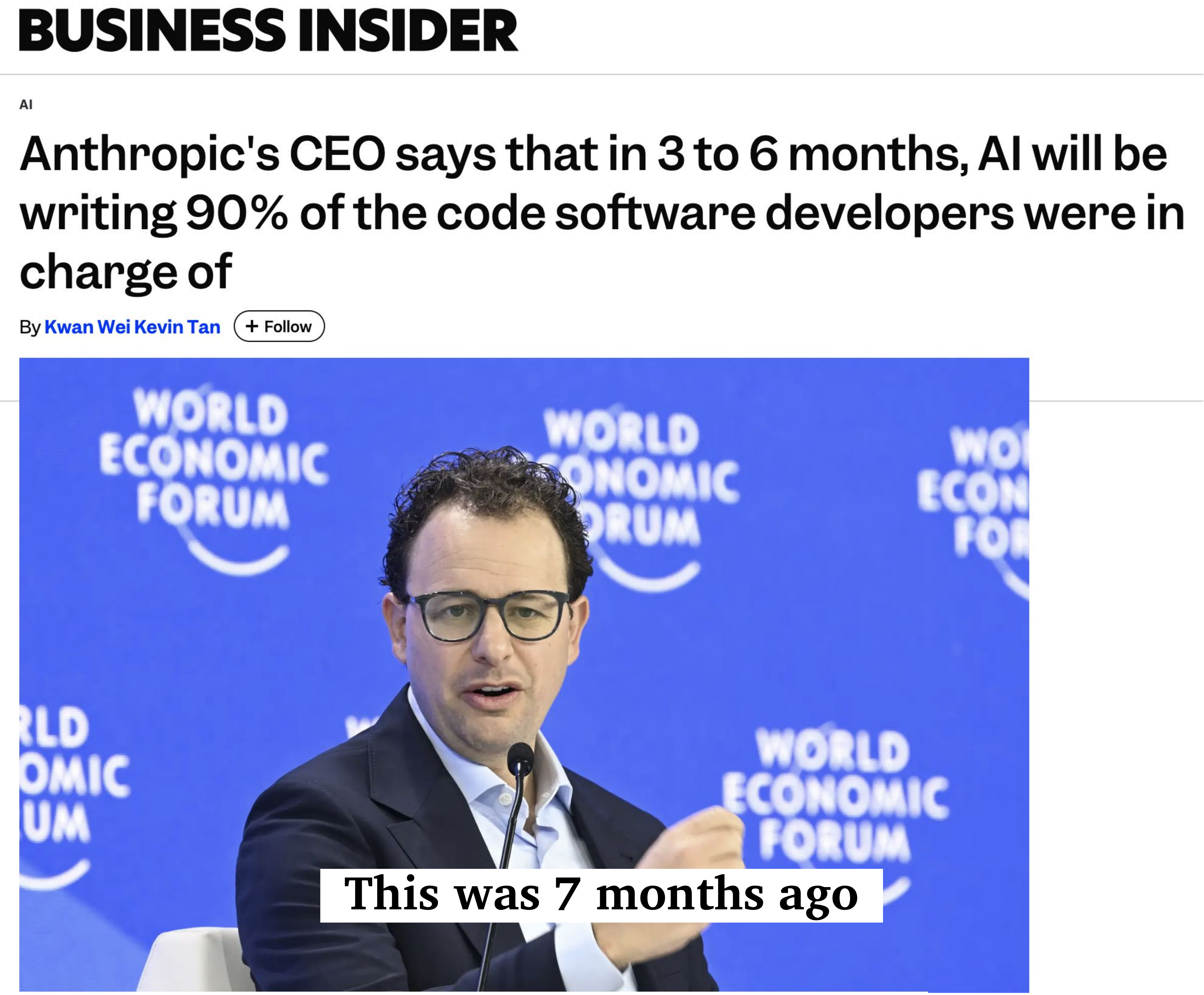
Devin

CURSOR

Continue

OpenHands

```
✻ Welcome to the Claude Code research preview!

CLAUDE
CODE

Login successful. Press Enter to continue
```



BUSINESS INSIDER

AI

# Anthropic's CEO says that in 3 to 6 months, AI will be writing 90% of the code software developers were in charge of

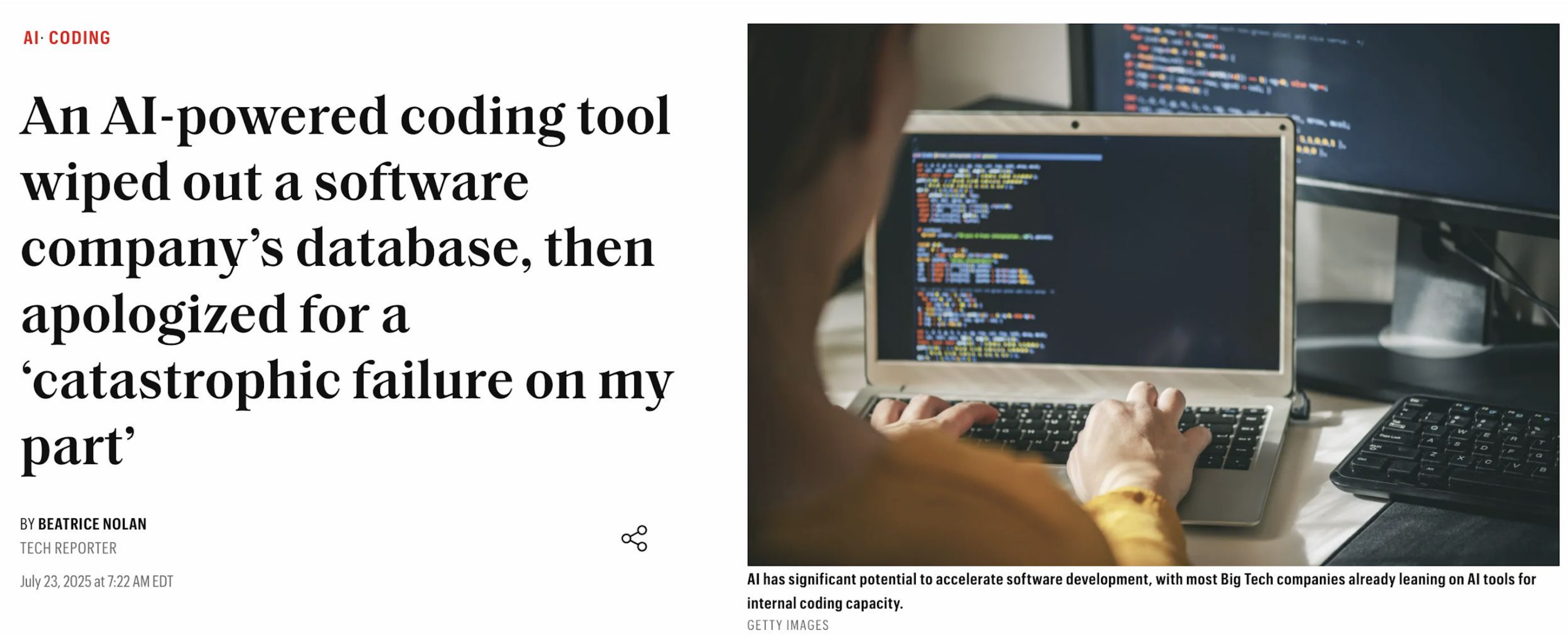
By **Kwan Wei Kevin Tan** + Follow

This was 7 months ago

Human Only

Level of AI Involvement

# Increased autonomy, increased risk



AI · CODING

## An AI-powered coding tool wiped out a software company's database, then apologized for a 'catastrophic failure on my part'

BY **BEATRICE NOLAN**
TECH REPORTER
July 23, 2025 at 7:22 AM EDT

**AI has significant potential to accelerate software development, with most Big Tech companies already leaning on AI tools for internal coding capacity.**

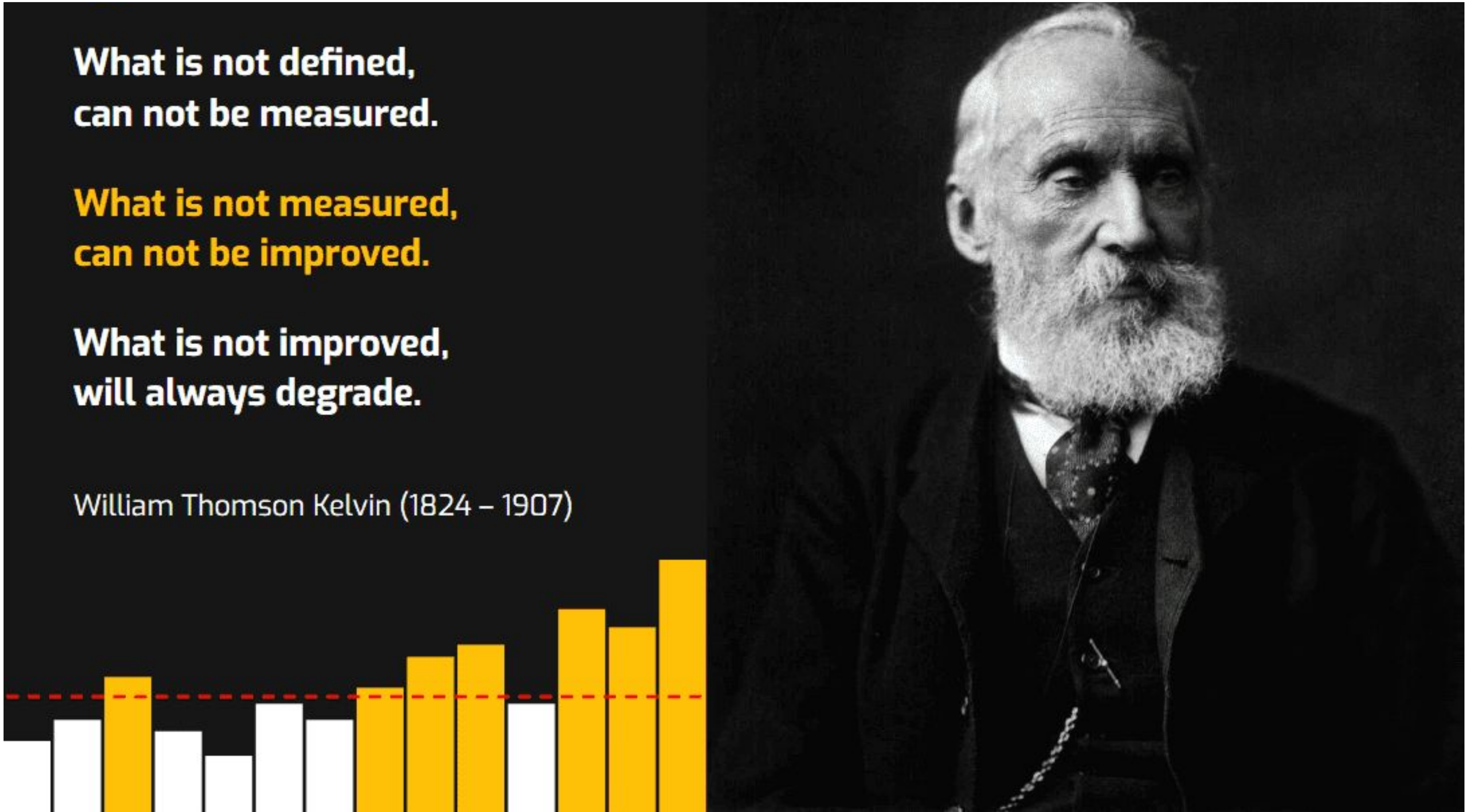
GETTY IMAGES

# These concerns are not limited to SWE

Customer Service and Chat Support
Sales & Marketing Automation
BANK
Finance & Banking
Healthcare
IOT
Manufacturing & IoT
Education

# So how do we design the next generation of (coding) agents?

# It starts with evaluation!



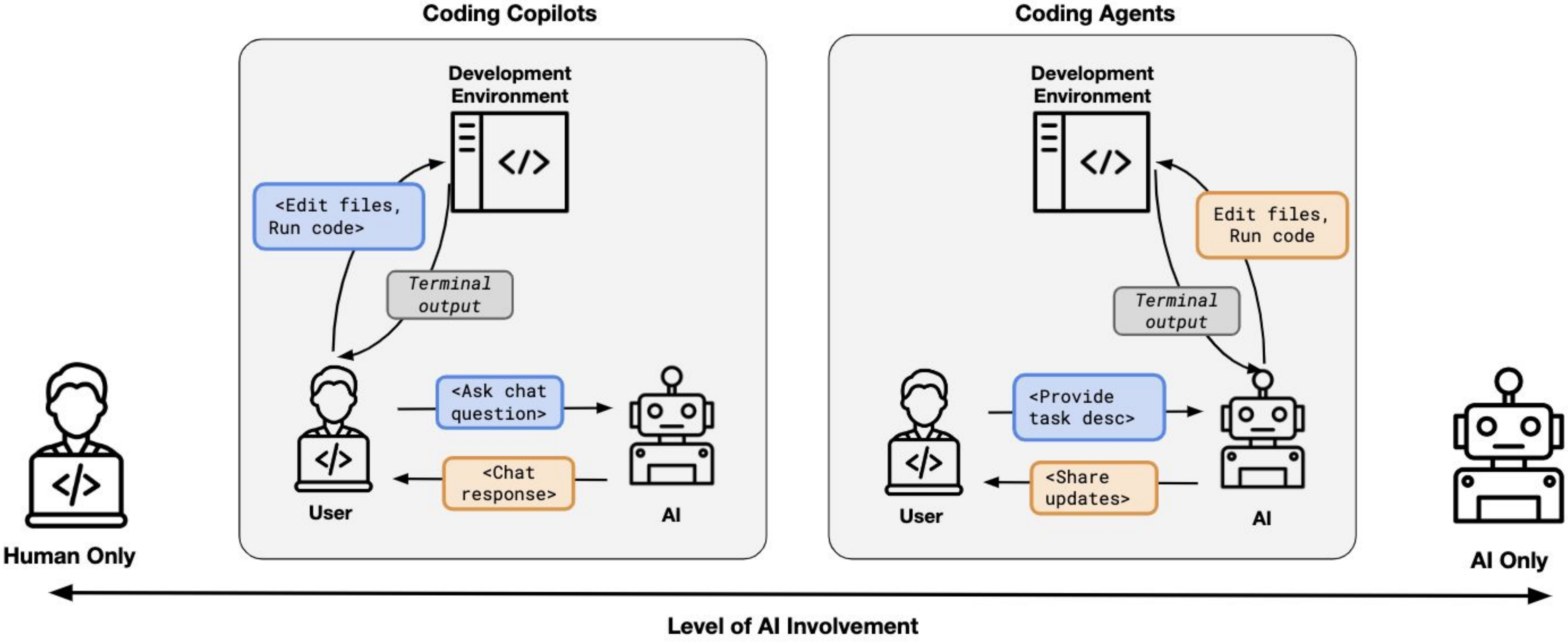
What is not defined,
can not be measured.

What is not measured,
can not be improved.

What is not improved,
will always degrade.

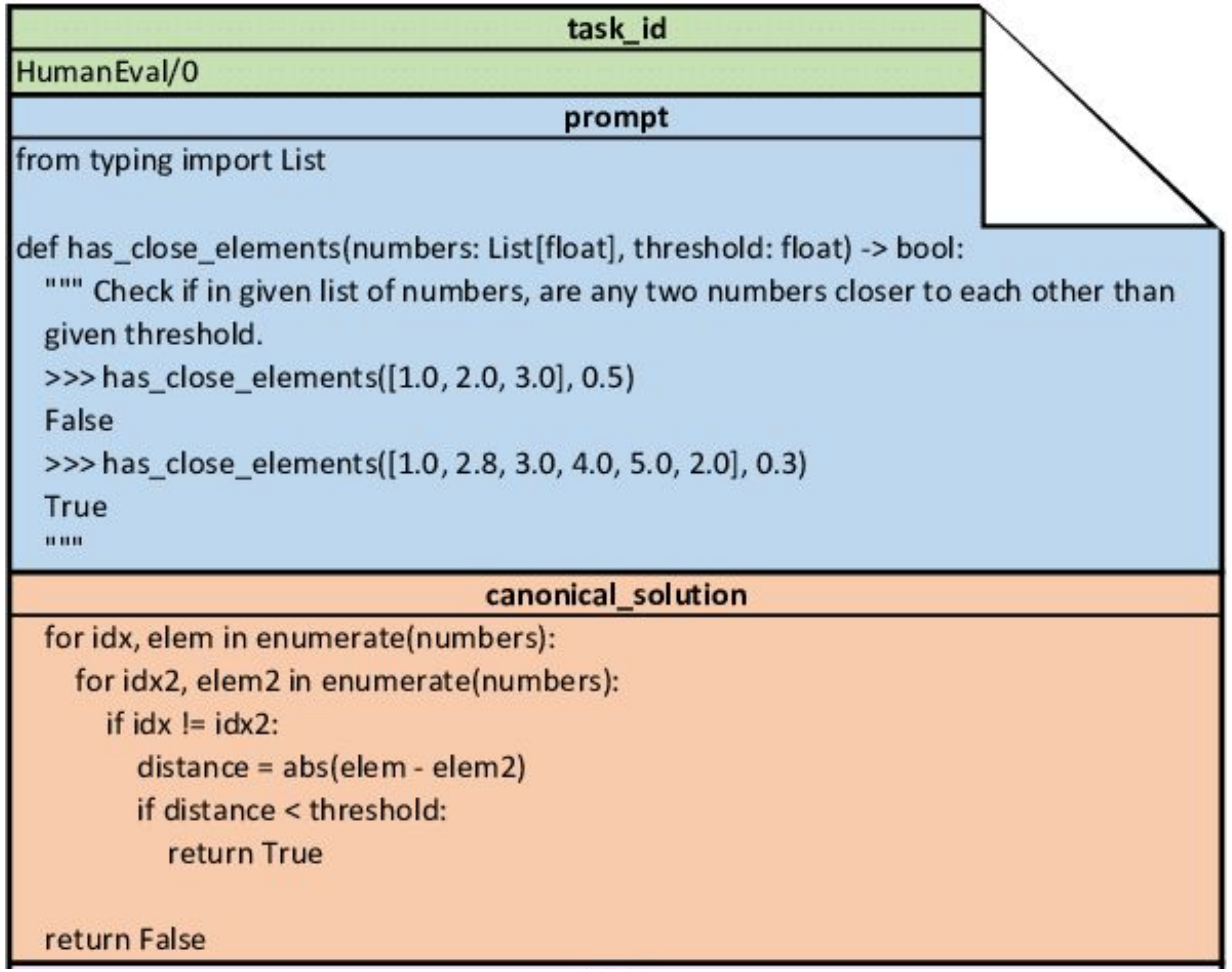
William Thomson Kelvin (1824 – 1907)

# Human-Centered Evaluations



# Evaluating copilots

# Prior Evaluations



**task_id**

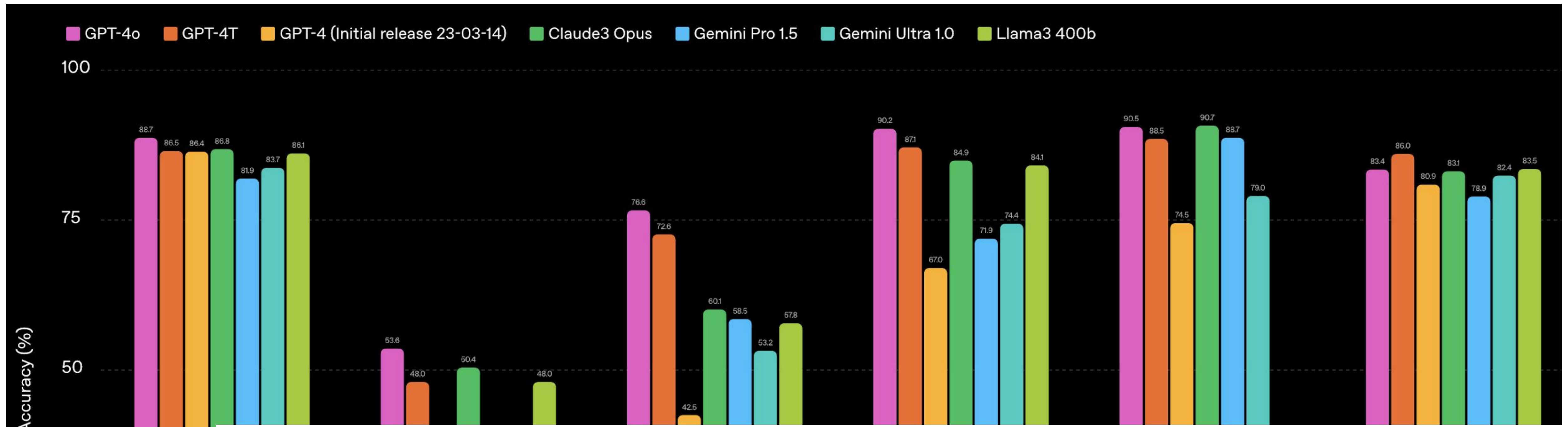
HumanEval/0

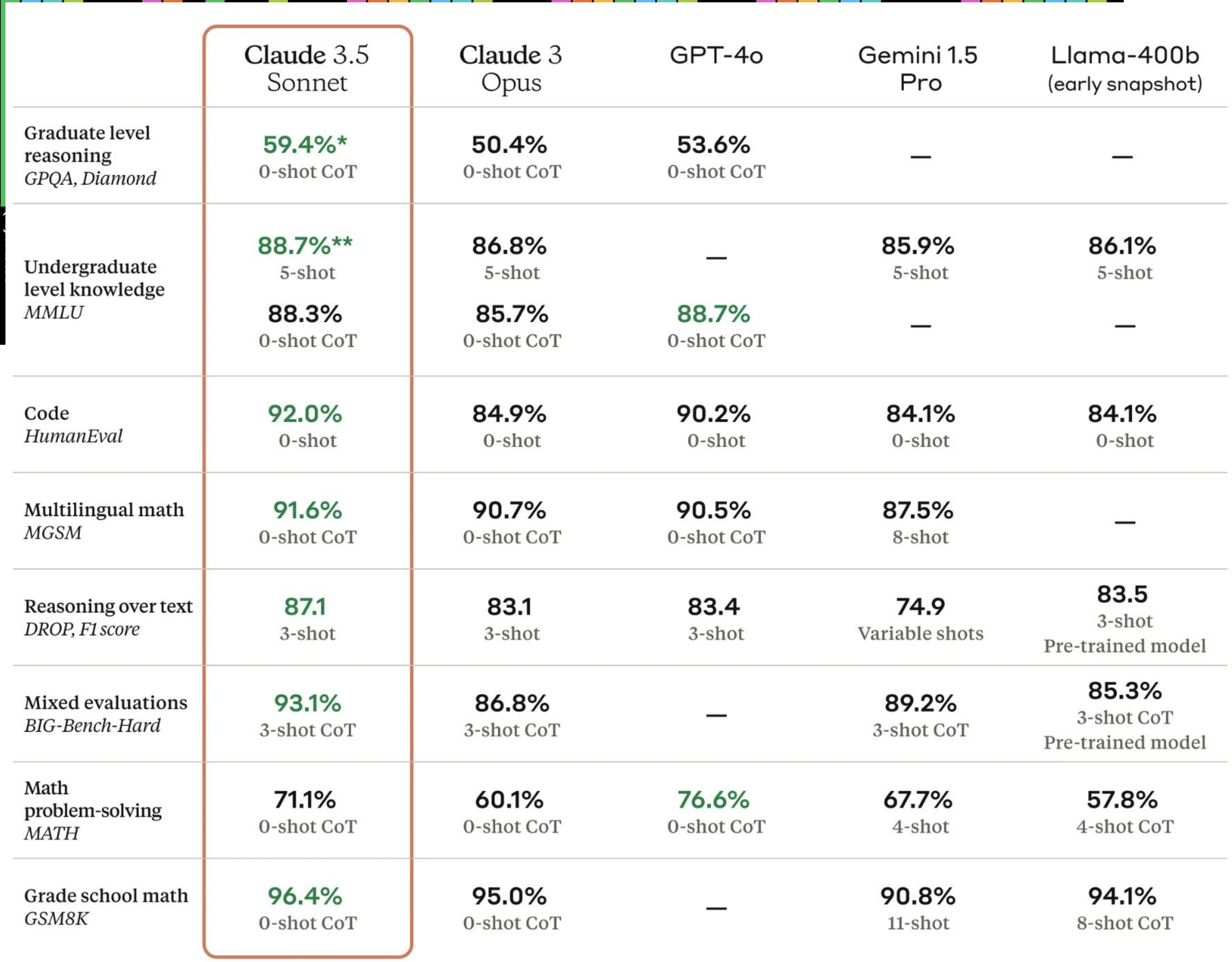
**prompt**

```
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
   """ Check if in given list of numbers, are any two numbers closer to each other than
   given threshold.
   >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
   False
   >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
   True
   """
```

**canonical_solution**

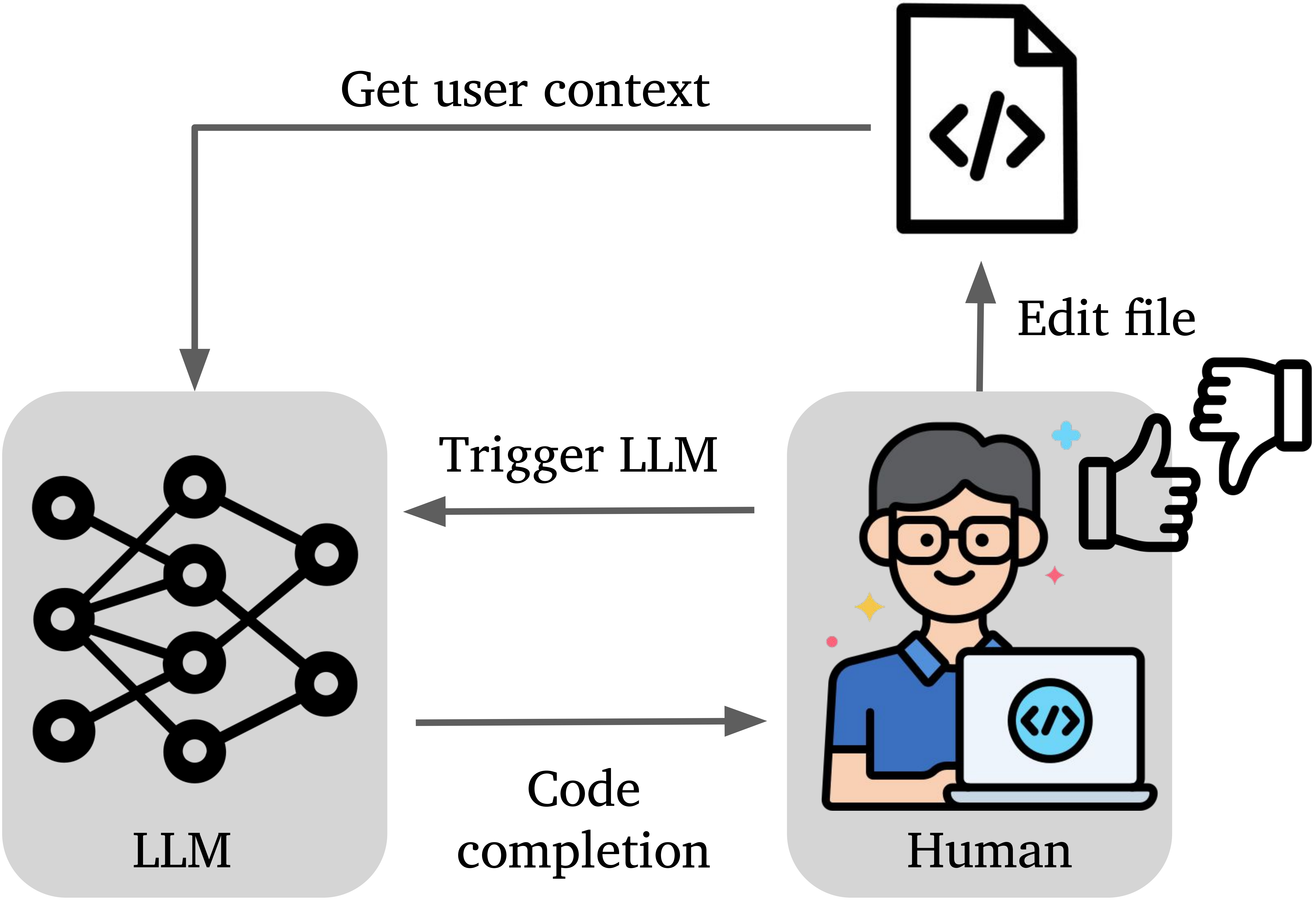
```
   for idx, elem in enumerate(numbers):
      for idx2, elem2 in enumerate(numbers):
         if idx != idx2:
            distance = abs(elem - elem2)
            if distance < threshold:
               return True

   return False
```



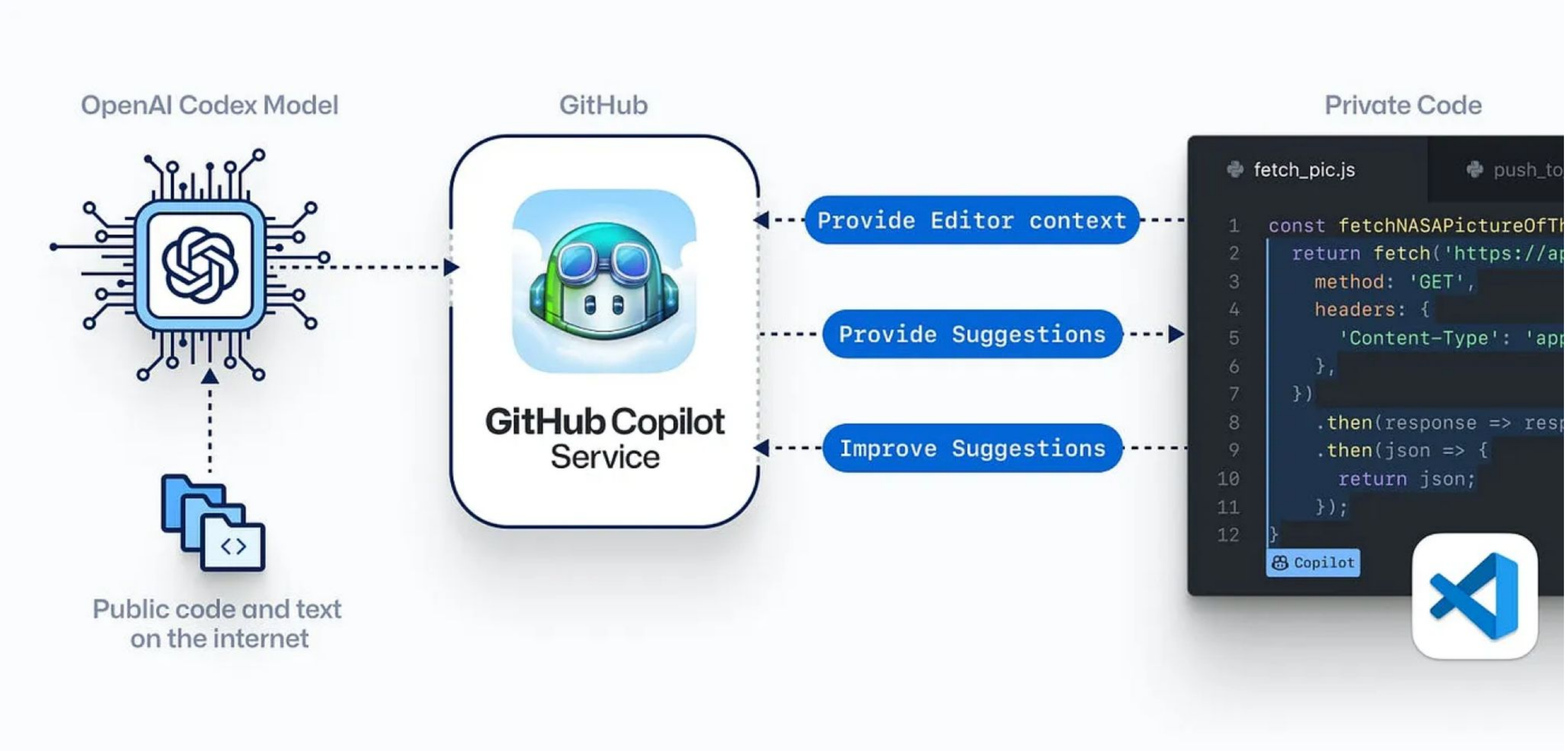


| | Claude 3.5 Sonnet | Claude 3 Opus | GPT-4o | Gemini 1.5 Pro | Llama-400b (early snapshot) |
|---|---|---|---|---|---|
| Graduate level reasoning<br>*GPQA, Diamond* | 59.4%*<br>0-shot CoT | 50.4%<br>0-shot CoT | 53.6%<br>0-shot CoT | — | — |
| Undergraduate level knowledge<br>*MMLU* | 88.7%**<br>5-shot<br>88.3%<br>0-shot CoT | 86.8%<br>5-shot<br>85.7%<br>0-shot CoT | —<br>88.7%<br>0-shot CoT | 85.9%<br>5-shot<br>— | 86.1%<br>5-shot<br>— |
| Code<br>*HumanEval* | 92.0%<br>0-shot | 84.9%<br>0-shot | 90.2%<br>0-shot | 84.1%<br>0-shot | 84.1%<br>0-shot |
| Multilingual math<br>*MGSM* | 91.6%<br>0-shot CoT | 90.7%<br>0-shot CoT | 90.5%<br>0-shot CoT | 87.5%<br>8-shot | — |
| Reasoning over text<br>*DROP, F1 score* | 87.1<br>3-shot | 83.1<br>3-shot | 83.4<br>3-shot | 74.9<br>Variable shots | 83.5<br>3-shot<br>Pre-trained model |
| Mixed evaluations<br>*BIG-Bench-Hard* | 93.1%<br>3-shot CoT | 86.8%<br>3-shot CoT | — | 89.2%<br>3-shot CoT | 85.3%<br>3-shot CoT<br>Pre-trained model |
| Math problem-solving<br>*MATH* | 71.1%<br>0-shot CoT | 60.1%<br>0-shot CoT | 76.6%<br>0-shot CoT | 67.7%<br>4-shot | 57.8%<br>4-shot CoT |
| Grade school math<br>*GSM8K* | 96.4%<br>0-shot CoT | 95.0%<br>0-shot CoT | — | 90.8%<br>11-shot | 94.1%<br>8-shot CoT |

# In practice, models don’t work on their own



# GitHub Copilot



```
const fetchNASAPictureOfTh
  return fetch('https://ap
    method: 'GET',
    headers: {
      'Content-Type': 'app
    },
  })
    .then(response => resp
    .then(json => {
      return json;
    });
}
```

# How do we pick which LLM to use?

Open code-specific models

Codestral

deepseek coder

Open models

Llama 3
From Meta

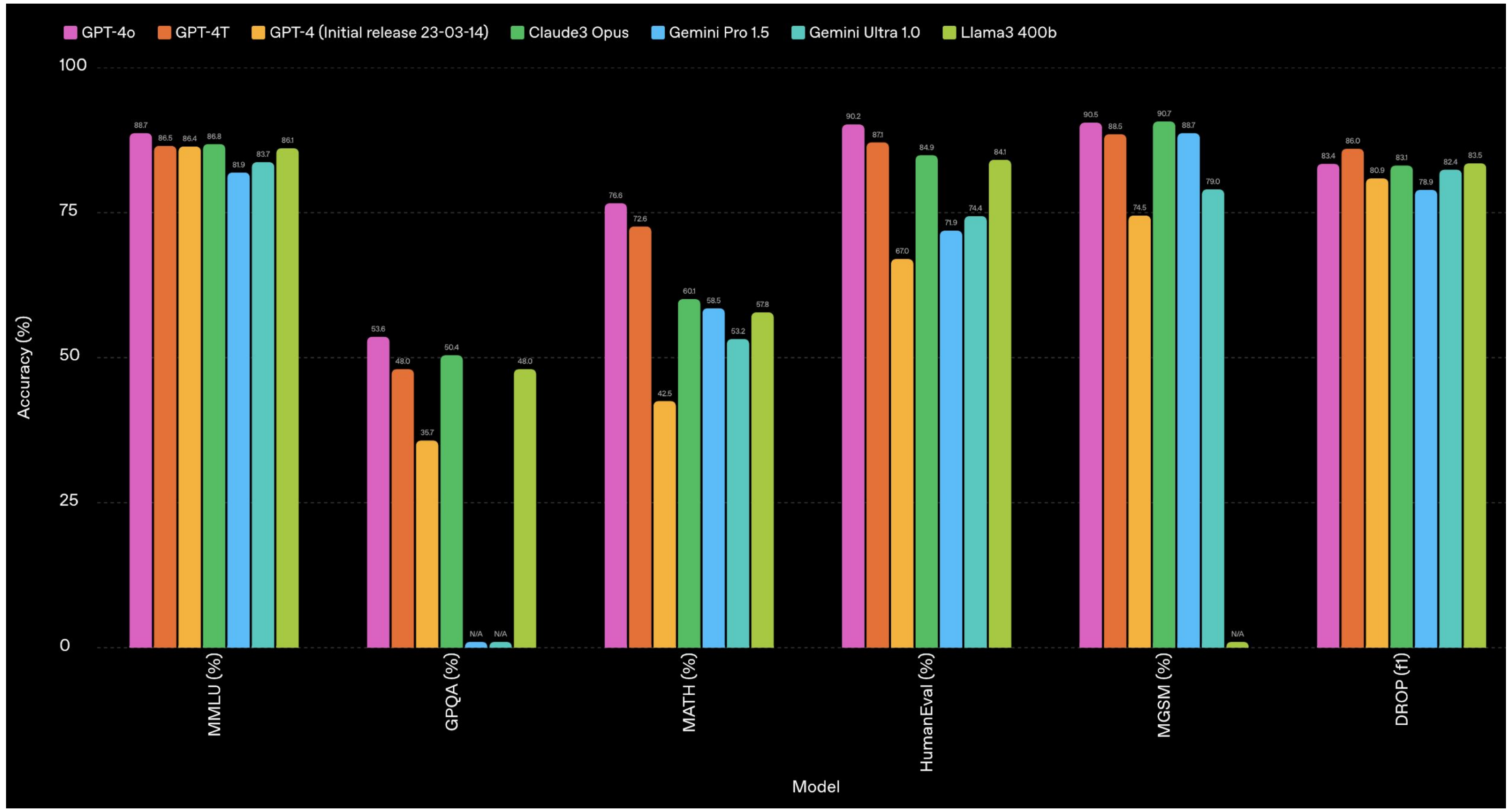
Commercial Models

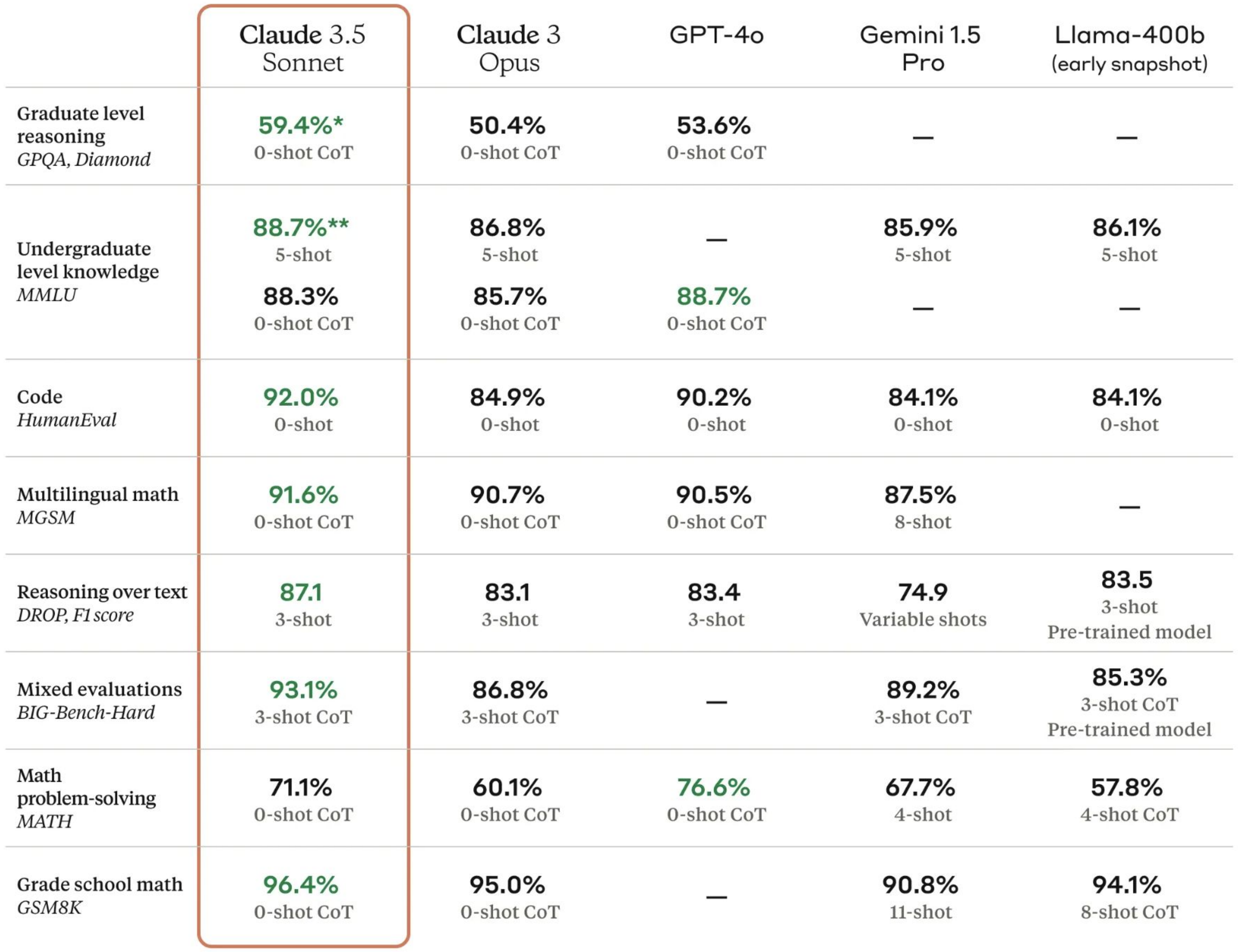
Gemini

GPT-4o

Claude 3.5 Sonnet
ANTHROP\C

# What about benchmark performance?

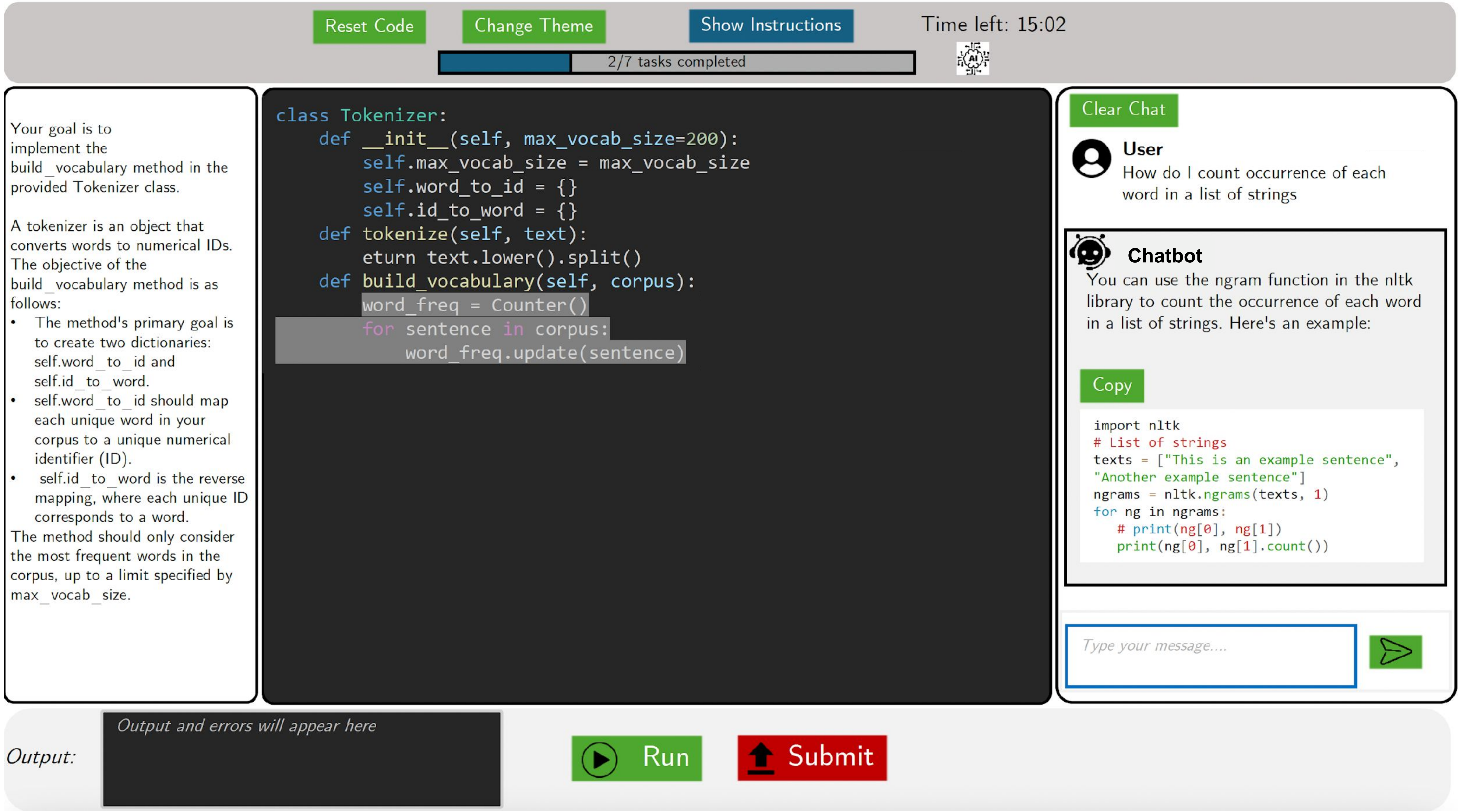




| | Claude 3.5 Sonnet | Claude 3 Opus | GPT-4o | Gemini 1.5 Pro | Llama-400b (early snapshot) |
|---|---|---|---|---|---|
| **Graduate level reasoning**<br>*GPQA, Diamond* | 59.4%*<br>0-shot CoT | 50.4%<br>0-shot CoT | 53.6%<br>0-shot CoT | — | — |
| **Undergraduate level knowledge**<br>*MMLU* | 88.7%**<br>5-shot | 86.8%<br>5-shot | — | 85.9%<br>5-shot | 86.1%<br>5-shot |
| | 88.3%<br>0-shot CoT | 85.7%<br>0-shot CoT | 88.7%<br>0-shot CoT | — | — |
| **Code**<br>*HumanEval* | 92.0%<br>0-shot | 84.9%<br>0-shot | 90.2%<br>0-shot | 84.1%<br>0-shot | 84.1%<br>0-shot |
| **Multilingual math**<br>*MGSM* | 91.6%<br>0-shot CoT | 90.7%<br>0-shot CoT | 90.5%<br>0-shot CoT | 87.5%<br>8-shot | — |
| **Reasoning over text**<br>*DROP, F1 score* | 87.1<br>3-shot | 83.1<br>3-shot | 83.4<br>3-shot | 74.9<br>Variable shots | 83.5<br>3-shot<br>Pre-trained model |
| **Mixed evaluations**<br>*BIG-Bench-Hard* | 93.1%<br>3-shot CoT | 86.8%<br>3-shot CoT | — | 89.2%<br>3-shot CoT | 85.3%<br>3-shot CoT<br>Pre-trained model |
| **Math problem-solving**<br>*MATH* | 71.1%<br>0-shot CoT | 60.1%<br>0-shot CoT | 76.6%<br>0-shot CoT | 67.7%<br>4-shot | 57.8%<br>4-shot CoT |
| **Grade school math**<br>*GSM8K* | 96.4%<br>0-shot CoT | 95.0%<br>0-shot CoT | — | 90.8%<br>11-shot | 94.1%<br>8-shot CoT |

We test this hypothesis by running a user study where people program with models of varying benchmark performance.

# We create a web interface to evaluate two forms of AI support



Your goal is to implement the build_vocabulary method in the provided Tokenizer class.

A tokenizer is an object that converts words to numerical IDs. The objective of the build_vocabulary method is as follows:

- The method's primary goal is to create two dictionaries: self.word_to_id and self.id_to_word.
- self.word_to_id should map each unique word in your corpus to a unique numerical identifier (ID).
- self.id_to_word is the reverse mapping, where each unique ID corresponds to a word.

The method should only consider the most frequent words in the corpus, up to a limit specified by max_vocab_size.

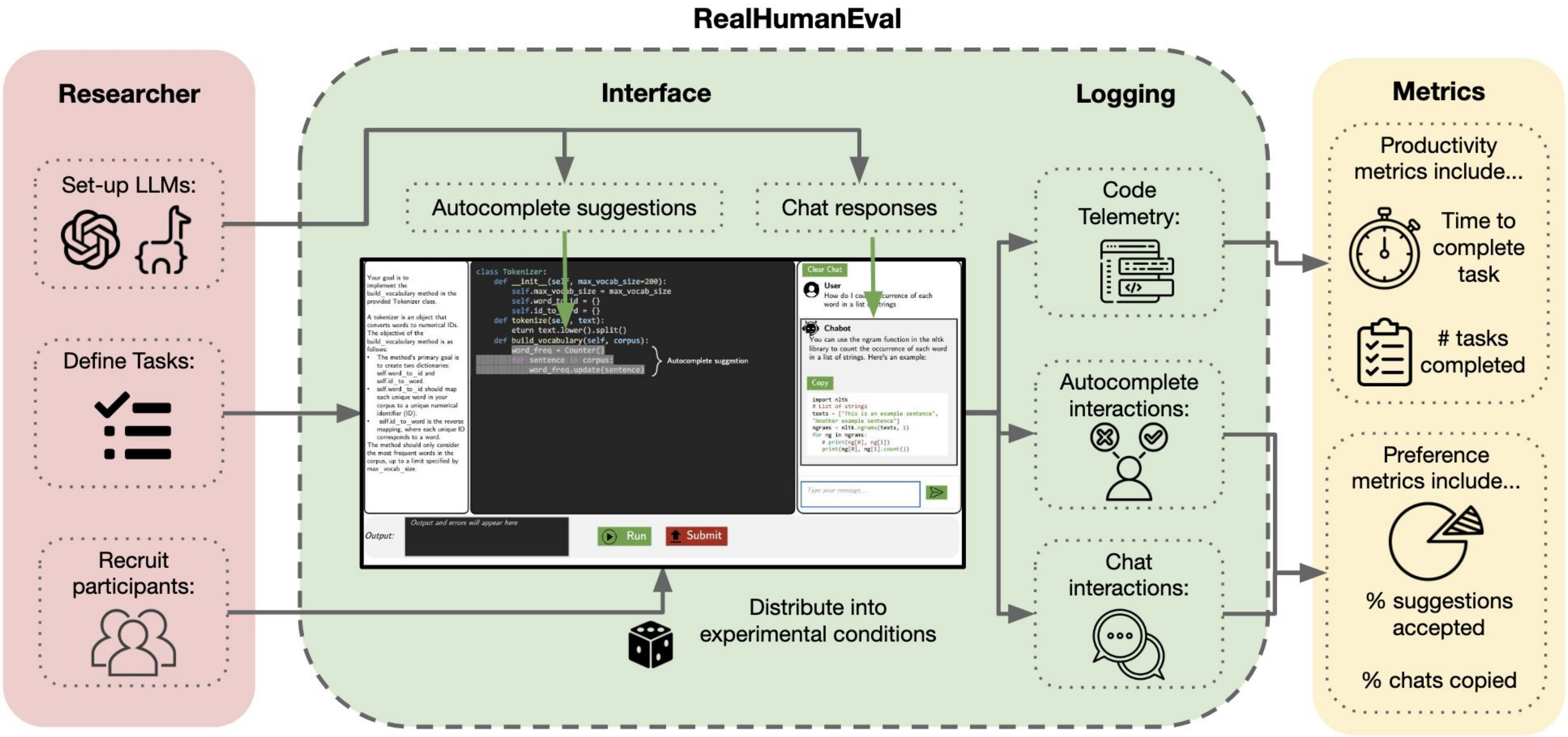
```
class Tokenizer:
    def __init__(self, max_vocab_size=200):
        self.max_vocab_size = max_vocab_size
        self.word_to_id = {}
        self.id_to_word = {}
    def tokenize(self, text):
        eturn text.lower().split()
    def build_vocabulary(self, corpus):
        word_freq = Counter()
        for sentence in corpus:
            word_freq.update(sentence)
```

**User**
How do I count occurrence of each word in a list of strings

**Chatbot**
You can use the ngram function in the nltk library to count the occurrence of each word in a list of strings. Here's an example:

```
import nltk
# List of strings
texts = ["This is an example sentence",
"Another example sentence"]
ngrams = nltk.ngrams(texts, 1)
for ng in ngrams:
   # print(ng[0], ng[1])
   print(ng[0], ng[1].count())
```

# The interface is part of an end-to-end pipeline to evaluate LLMs for coding



Your goal is to implement the build_vocabulary method in the provided Tokenizer class.

A tokenizer is an object that converts words to numerical IDs. The objective of the build_vocabulary method is as follows:

- The method's primary goal is to create two dictionaries: self.word_to_id and self.id_to_word.
- self.word_to_id should map each unique word in your corpus to a unique numerical identifier (ID).
- self.id_to_word is the reverse mapping, where each unique ID corresponds to a word.

The method should only consider the most frequent words in the corpus, up to a limit specified by max_vocab_size.

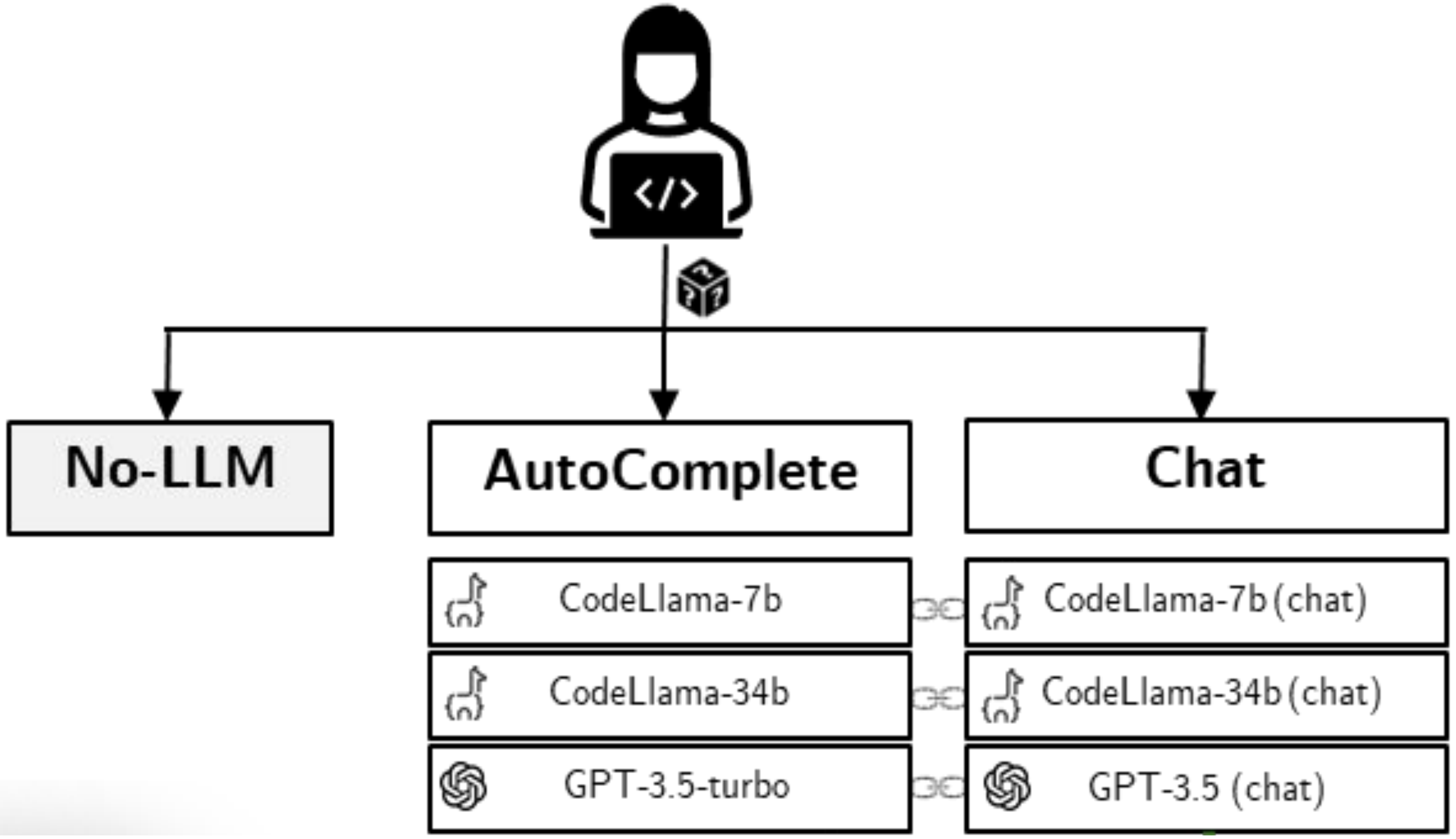
```
class Tokenizer:
    def __init__(sel , max_vocab_size=200):
        self.max_vo ab_size = max_vocab_size
        self.word_t  id = {}
        self.id_to_  d = {}
    def tokenize(se  , text):
        eturn text.lower().split()
    def build_vocabulary(self, corpus):
        word_freq = Counter()
        for sentence in corpus:
            word_freq.update(sentence)
```

**User**
How do I cou  currence of each word in a list   trings

**Chabot**
You can use the ngram function in the nltk library to count the occurrence of each word in a list of strings. Here's an example:

```
import nltk
# List of strings
texts = ["This is an example sentence",
"Another example sentence"]
ngrams = nltk.ngrams(texts, 1)
for ng in ngrams:
    # print(ng[0], ng[1])
    print(ng[0], ng[1].count())
```

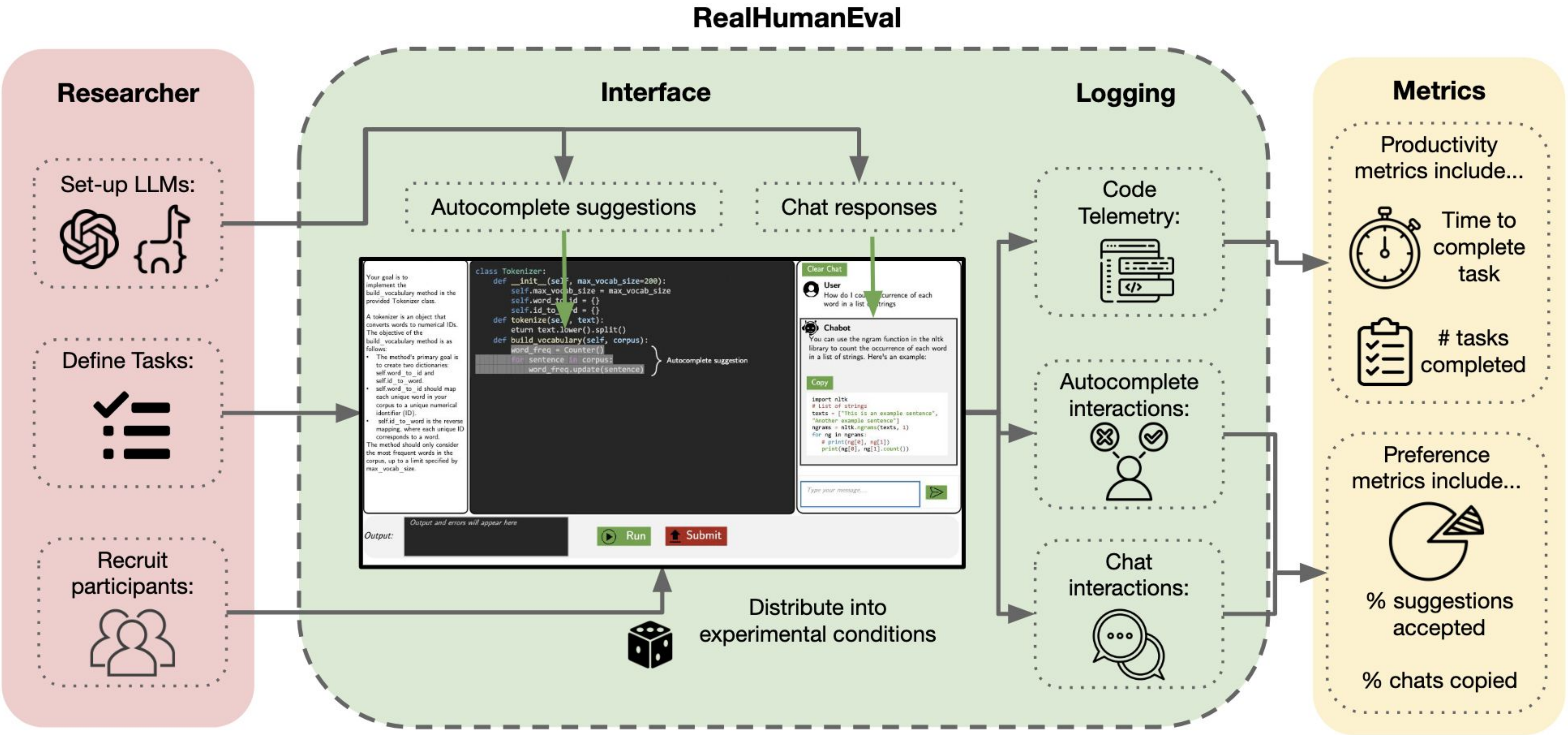
# Our results highlight the importance of human-in-the-loop evaluation



Notice that none of these models are used anymore for code!

- 213 participants assigned to 7 conditions
- Gaps in benchmark performance do not match differences in human performance
- The benefits that a human gets from LLM support vary by task type (e.g., data science vs interview-style problems)

# What’s the best way to do human-in-the-loop evaluation?



Your goal is to implement the build_vocabulary method in the provided Tokenizer class.

A tokenizer is an object that converts words to numerical IDs. The objective of the build_vocabulary method is as follows:

- The method's primary goal is to create two dictionaries: self.word_to_id and self.id_to_word.
- self.word_to_id should map each unique word in your corpus to a unique numerical identifier (ID).
- self.id_to_word is the reverse mapping, where each unique ID corresponds to a word.

The method should only consider the most frequent words in the corpus, up to a limit specified by max_vocab_size.

```
class Tokenizer:
    def __init__(sel[illegible], max_vocab_size=200):
        self.max_voc[illegible]b_size = max_vocab_size
        self.word_t[illegible]id = {}
        self.id_to_[illegible]d = {}
    def tokenize(se[illegible], text):
        eturn text.lower().split()
    def build_vocabulary(self, corpus):
        word_freq = Counter()
        for sentence in corpus:
            word_freq.update(sentence)
```

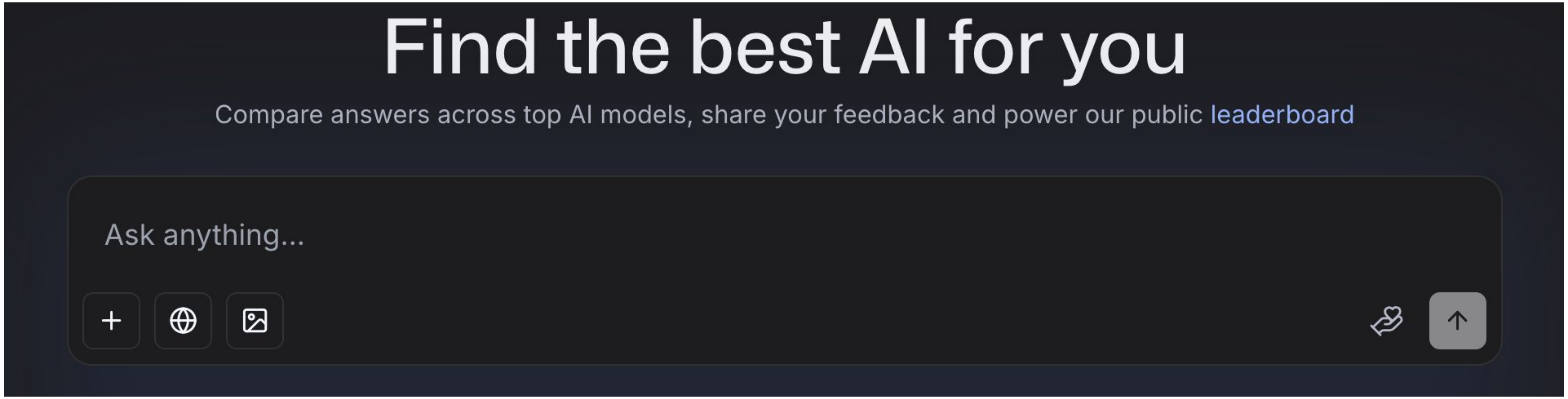
**User**
How do I cou[illegible]currence of each word in a list [illegible]trings

**Chabot**
You can use the ngram function in the nltk library to count the occurrence of each word in a list of strings. Here's an example:

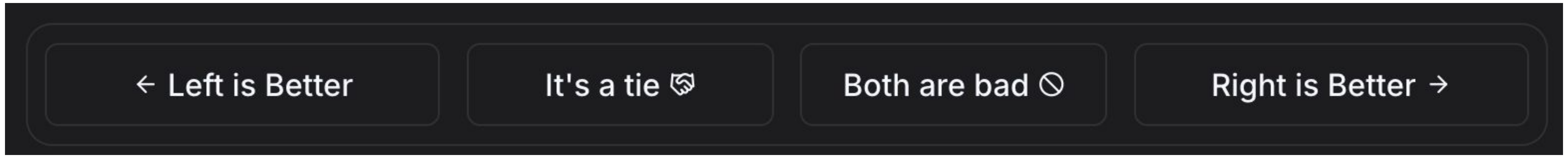
```
import nltk
# List of strings
texts = ["This is an example sentence",
"Another example sentence"]
ngrams = nltk.ngrams(texts, 1)
for ng in ngrams:
   # print(ng[0], ng[1])
   print(ng[0], ng[1].count())
```

Running studies involve user recruitment and a user only interacts with one model. **Realistic usage, but not scalable!**

# What’s the best way to do human-in-the-loop evaluation?


Find the best AI for you

Compare answers across top AI models, share your feedback and power our public leaderboard



As of September 2025, the platform has collected over 3.5 million head-to-head votes across more than 400 models

# Demo

https://lmarena.ai/

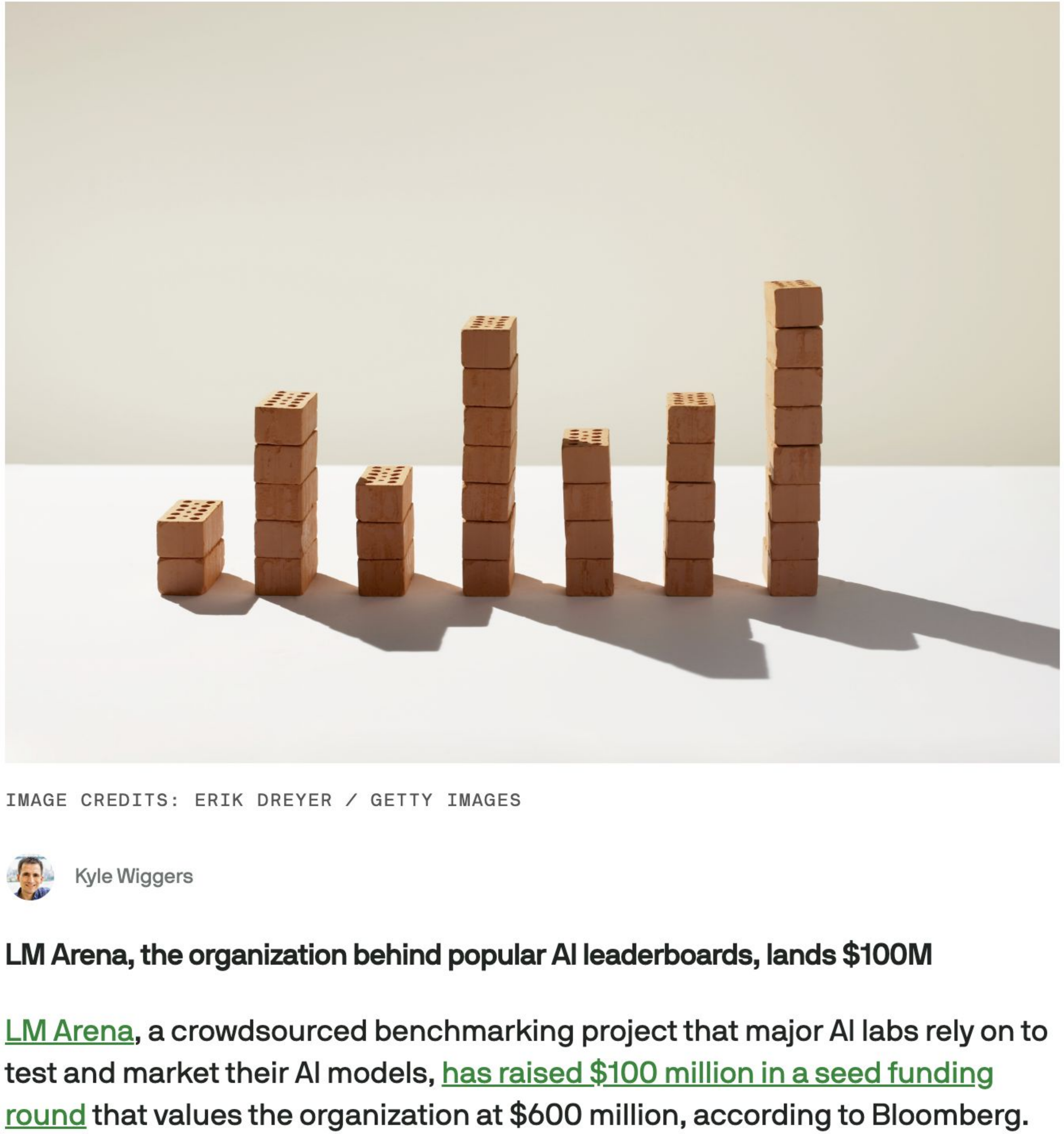


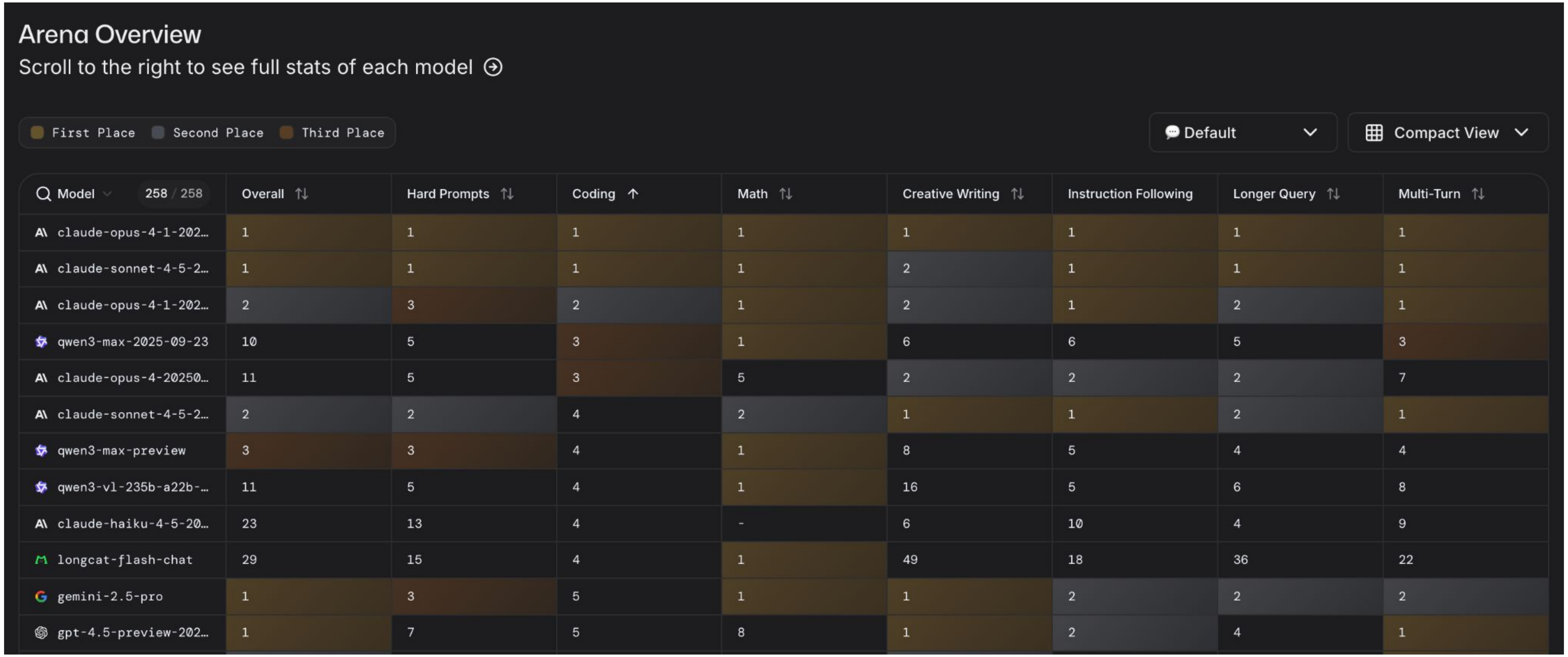
IMAGE CREDITS: ERIK DREYER / GETTY IMAGES

Kyle Wiggers

**LM Arena, the organization behind popular AI leaderboards, lands $100M**

LM Arena, a crowdsourced benchmarking project that major AI labs rely on to test and market their AI models, has raised $100 million in a seed funding round that values the organization at $600 million, according to Bloomberg.

# What’s the best way to do human-in-the-loop evaluation?



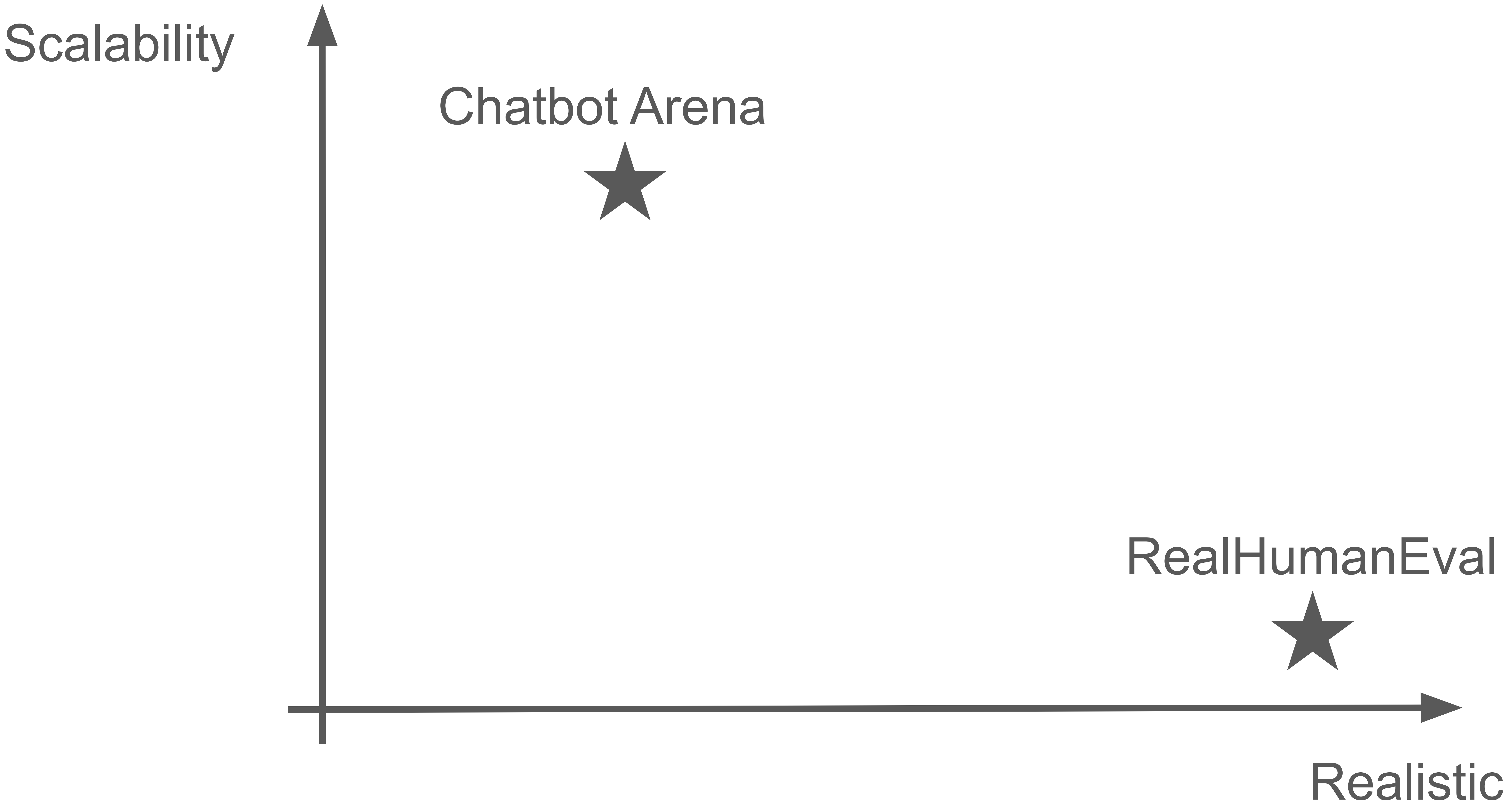
Arena Overview

Scroll to the right to see full stats of each model

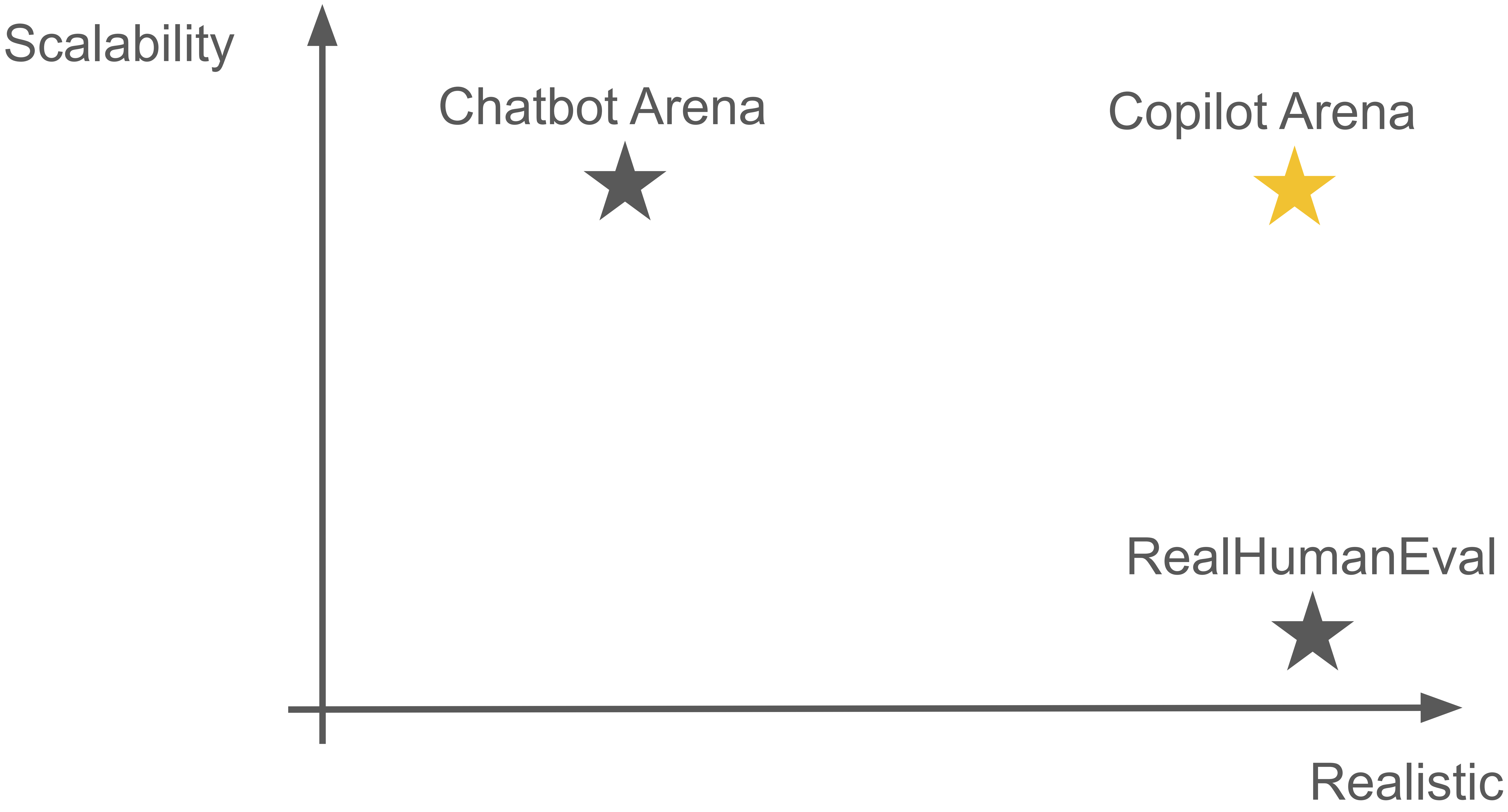
| Model 258 / 258 | Overall | Hard Prompts | Coding | Math | Creative Writing | Instruction Following | Longer Query | Multi-Turn |
|---|---|---|---|---|---|---|---|---|
| claude-opus-4-1-202… | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| claude-sonnet-4-5-2… | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 |
| claude-opus-4-1-202… | 2 | 3 | 2 | 1 | 2 | 1 | 2 | 1 |
| qwen3-max-2025-09-23 | 10 | 5 | 3 | 1 | 6 | 6 | 5 | 3 |
| claude-opus-4-20250… | 11 | 5 | 3 | 5 | 2 | 2 | 2 | 7 |
| claude-sonnet-4-5-2… | 2 | 2 | 4 | 2 | 1 | 1 | 2 | 1 |
| qwen3-max-preview | 3 | 3 | 4 | 1 | 8 | 5 | 4 | 4 |
| qwen3-vl-235b-a22b-… | 11 | 5 | 4 | 1 | 16 | 5 | 6 | 8 |
| claude-haiku-4-5-20… | 23 | 13 | 4 | - | 6 | 10 | 4 | 9 |
| longcat-flash-chat | 29 | 15 | 4 | 1 | 49 | 18 | 36 | 22 |
| gemini-2.5-pro | 1 | 3 | 5 | 1 | 1 | 2 | 2 | 2 |
| gpt-4.5-preview-202… | 1 | 7 | 5 | 8 | 1 | 2 | 4 | 1 |

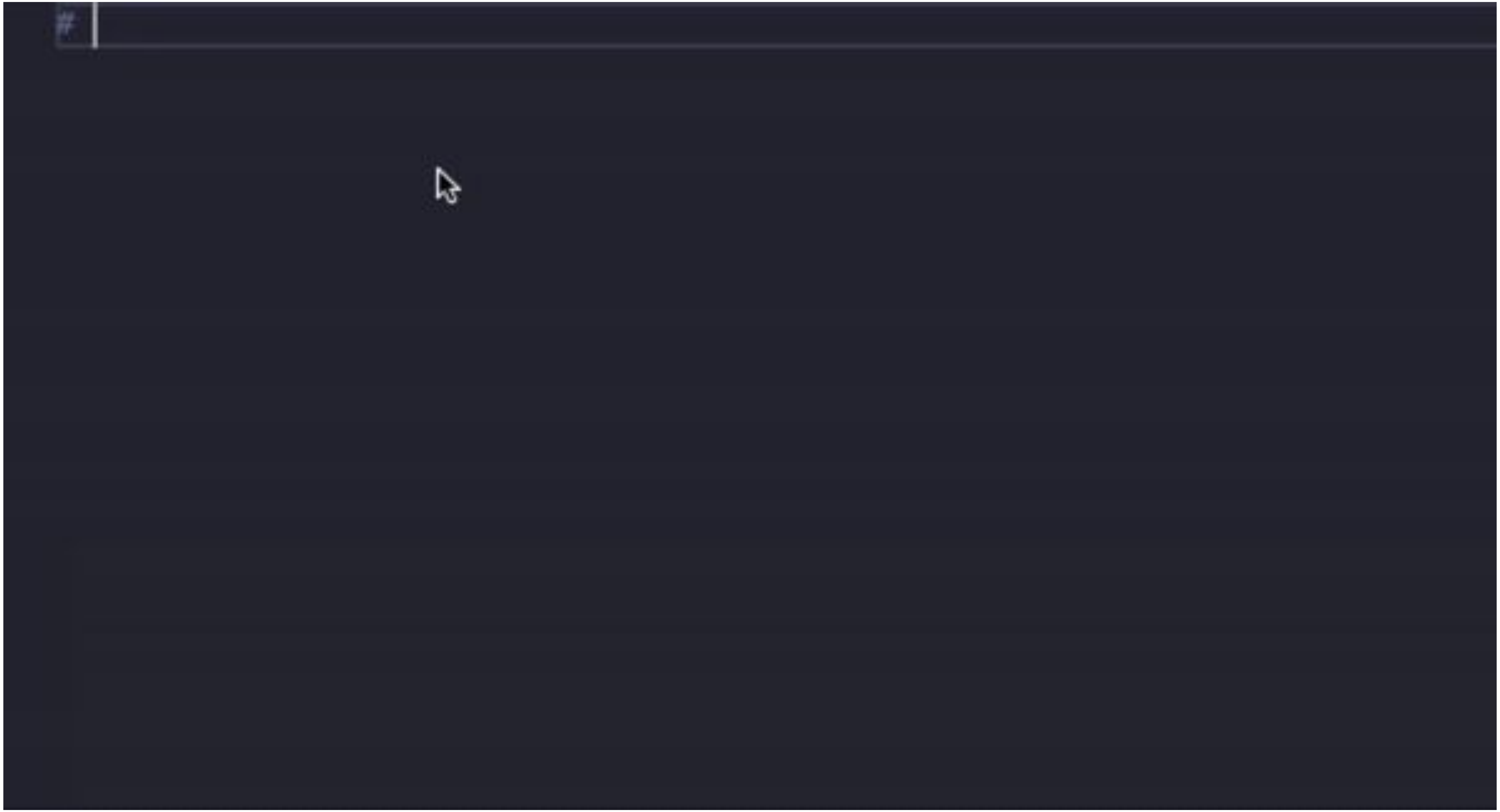
Chatbot Arena has introduced a coding category to evaluate models with coding capabilites. **Scalable, but not realistic!**

# Existing evaluations are flawed

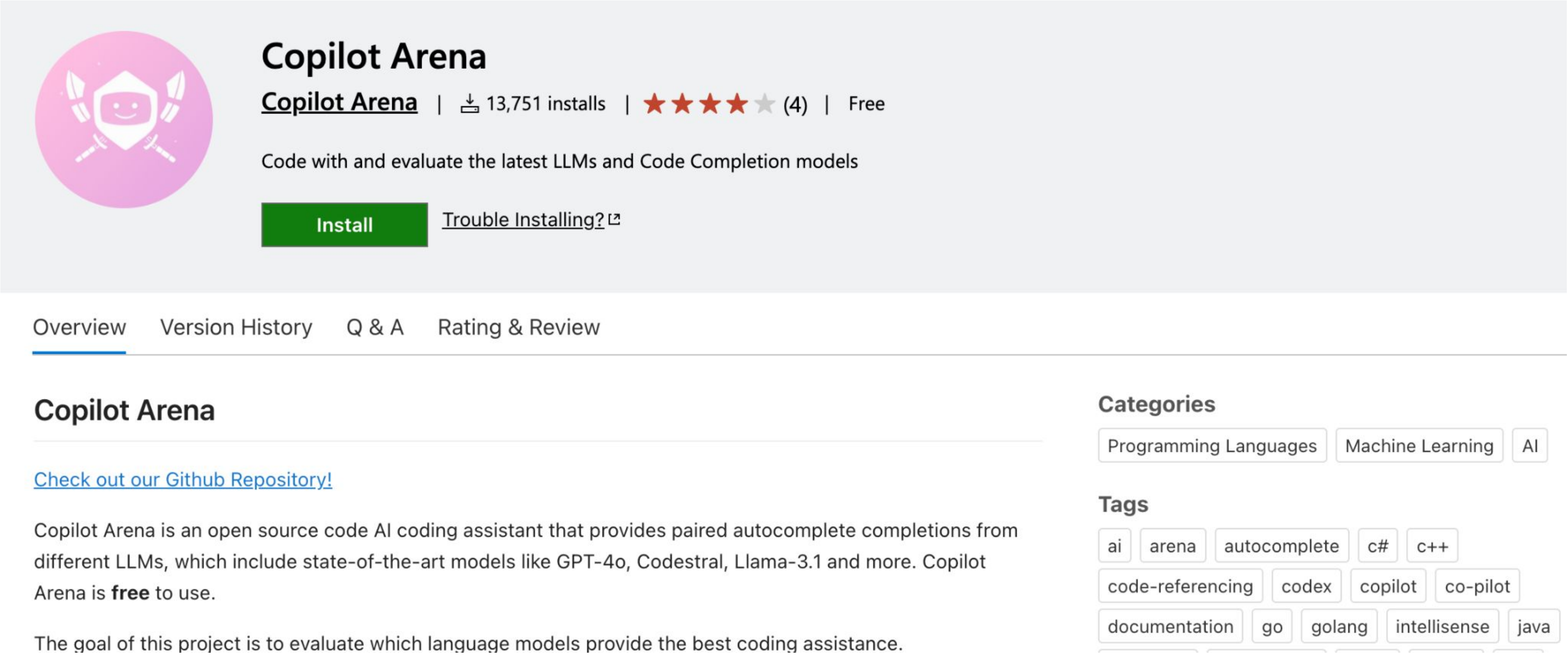


# Copilot Arena aims to achieve the best of both worlds





# Copilot Arena is a real VSCode Extension



**Copilot Arena**

Copilot Arena | 13,751 installs | ★★★★☆ (4) | Free

Code with and evaluate the latest LLMs and Code Completion models

## Copilot Arena

Check out our Github Repository!

Copilot Arena is an open source code AI coding assistant that provides paired autocomplete completions from different LLMs, which include state-of-the-art models like GPT-4o, Codestral, Llama-3.1 and more. Copilot Arena is **free** to use.

The goal of this project is to evaluate which language models provide the best coding assistance.

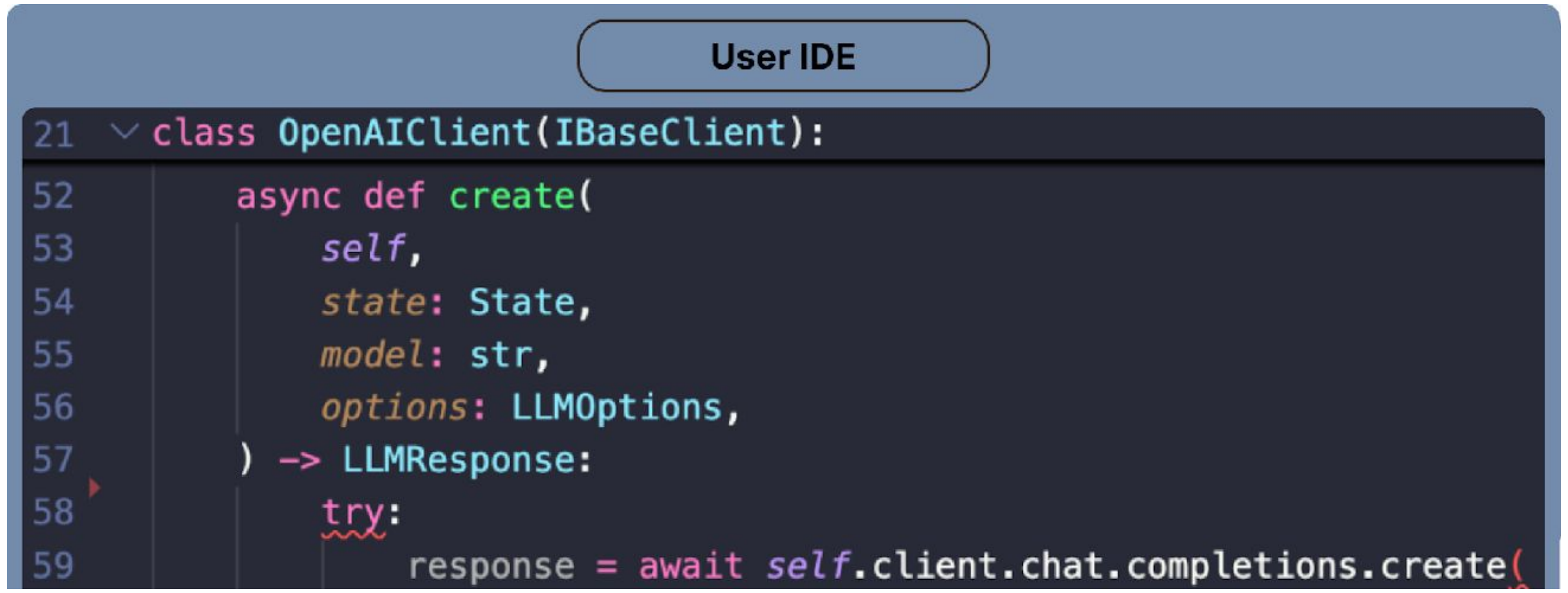
**Categories**

Programming Languages | Machine Learning | AI

**Tags**

ai | arena | autocomplete | c# | c++
code-referencing | codex | copilot | co-pilot
documentation | go | golang | intellisense | java

# Copilot Arena workflow



```
class OpenAIClient(IBaseClient):
    async def create(
        self,
        state: State,
        model: str,
        options: LLMOptions,
    ) -> LLMResponse:
        try:
            response = await self.client.chat.completions.create(
```

# What models does Copilot Arena support?

Open code-specific models

{ Codestral }

deepseek coder

Open models

Llama 3
From Meta

Commercial Models

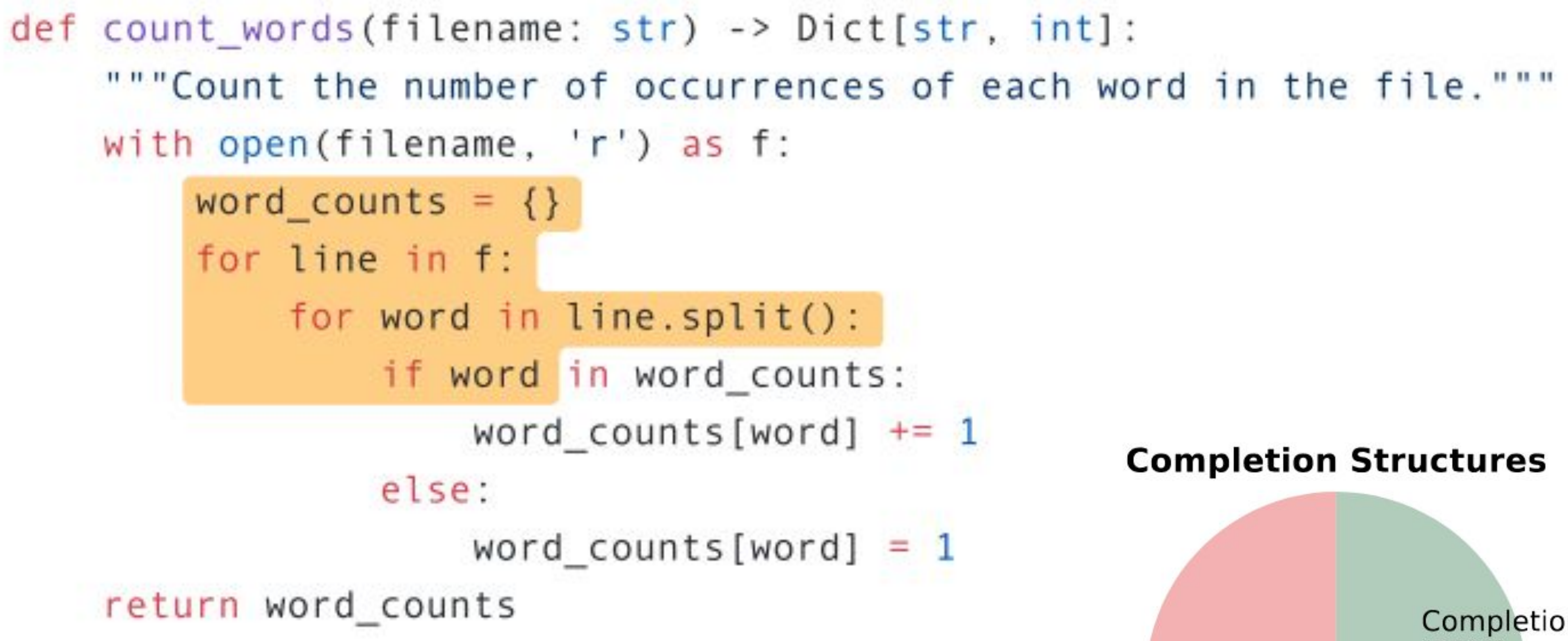
Gemini

GPT-4o

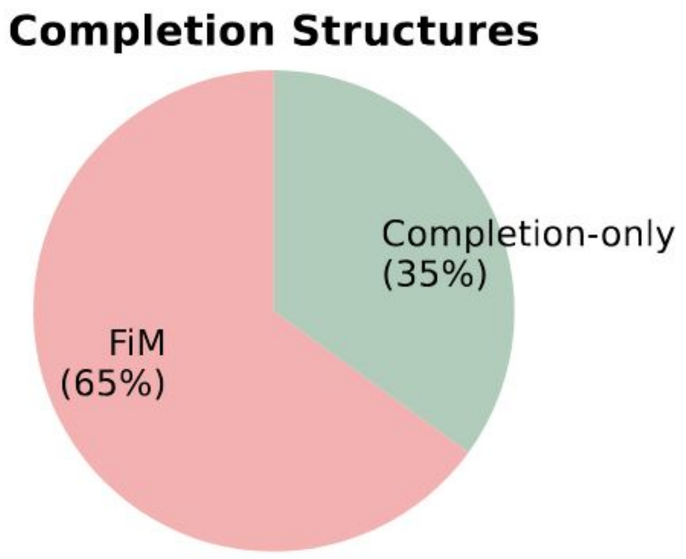
Claude 3.5 Sonnet
ANTHROP\C

# How are chat models performing code completions?

Need to be able to
“fill-in-the-middle”
(FiM)



```
def count_words(filename: str) -> Dict[str, int]:
    """Count the number of occurrences of each word in the file."""
    with open(filename, 'r') as f:
        word_counts = {}
        for line in f:
            for word in line.split():
                if word in word_counts:
                    word_counts[word] += 1
                else:
                    word_counts[word] = 1
    return word_counts
```



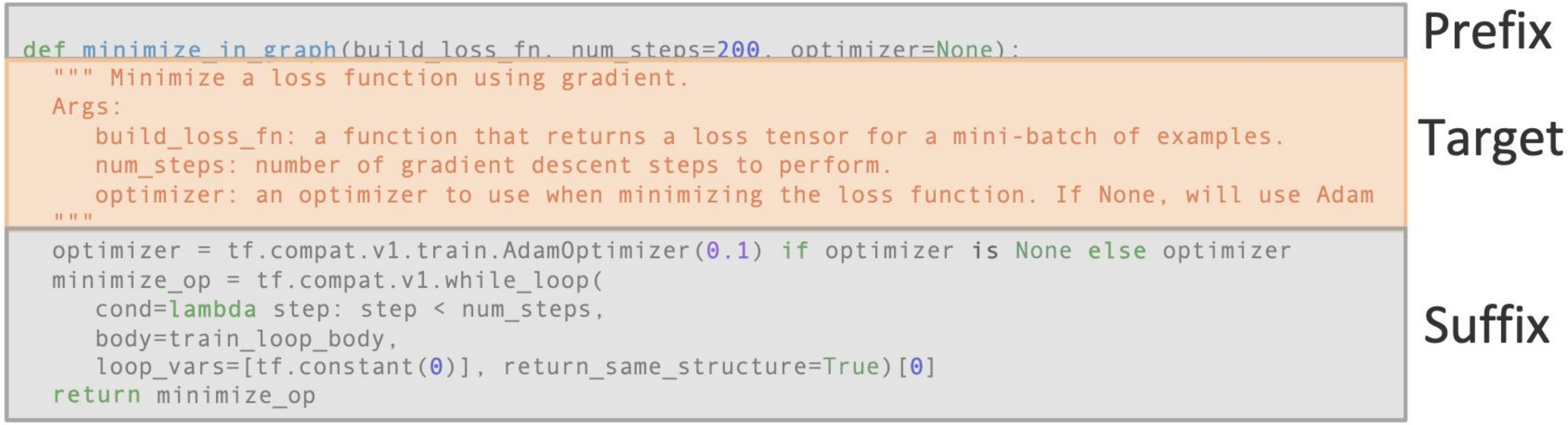
Not trained to do FiM!

Gemini

GPT-4o

Claude 3.5 Sonnet
ANTHROP\C

# What’s unique about code?

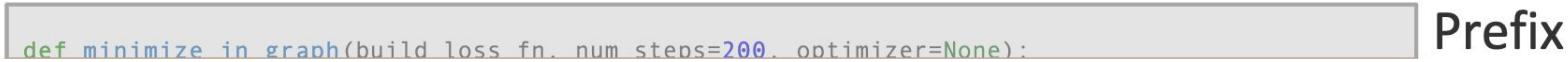


```
def minimize_in_graph(build_loss_fn, num_steps=200, optimizer=None):
   """ Minimize a loss function using gradient.
   Args:
      build_loss_fn: a function that returns a loss tensor for a mini-batch of examples.
      num_steps: number of gradient descent steps to perform.
      optimizer: an optimizer to use when minimizing the loss function. If None, will use Adam
   """
   optimizer = tf.compat.v1.train.AdamOptimizer(0.1) if optimizer is None else optimizer
   minimize_op = tf.compat.v1.while_loop(
      cond=lambda step: step < num_steps,
      body=train_loop_body,
      loop_vars=[tf.constant(0)], return_same_structure=True)[0]
   return minimize_op
```

Slide from CMU Neural Code Generation
(11-891)

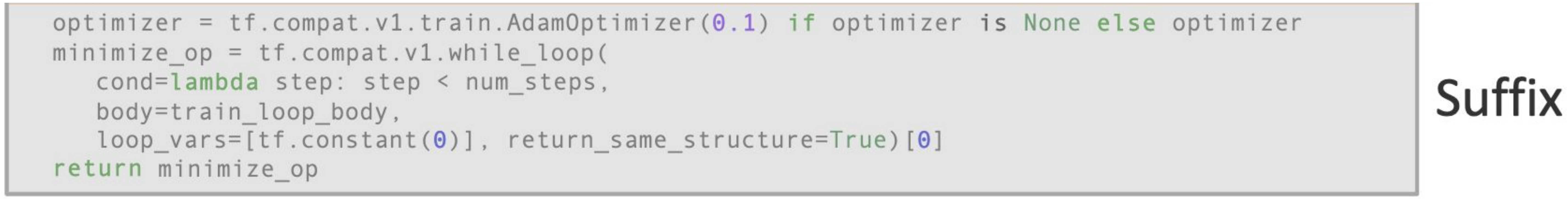
# Training models to fill in the middle

<PRE>



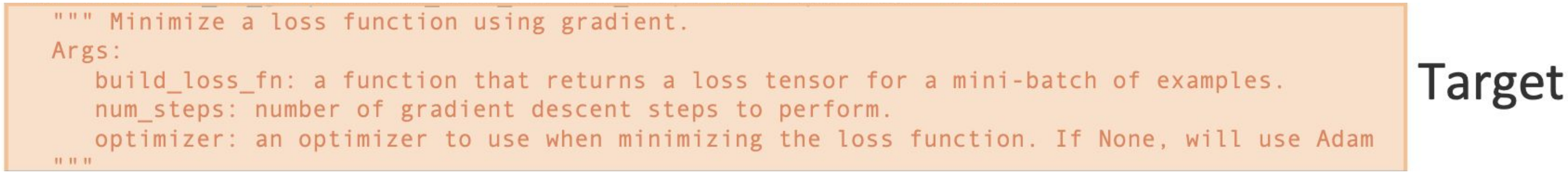
```
def minimize_in_graph(build_loss_fn, num_steps=200, optimizer=None):
```

<SUF>



```
  optimizer = tf.compat.v1.train.AdamOptimizer(0.1) if optimizer is None else optimizer
  minimize_op = tf.compat.v1.while_loop(
     cond=lambda step: step < num_steps,
     body=train_loop_body,
     loop_vars=[tf.constant(0)], return_same_structure=True)[0]
  return minimize_op
```

<MID>



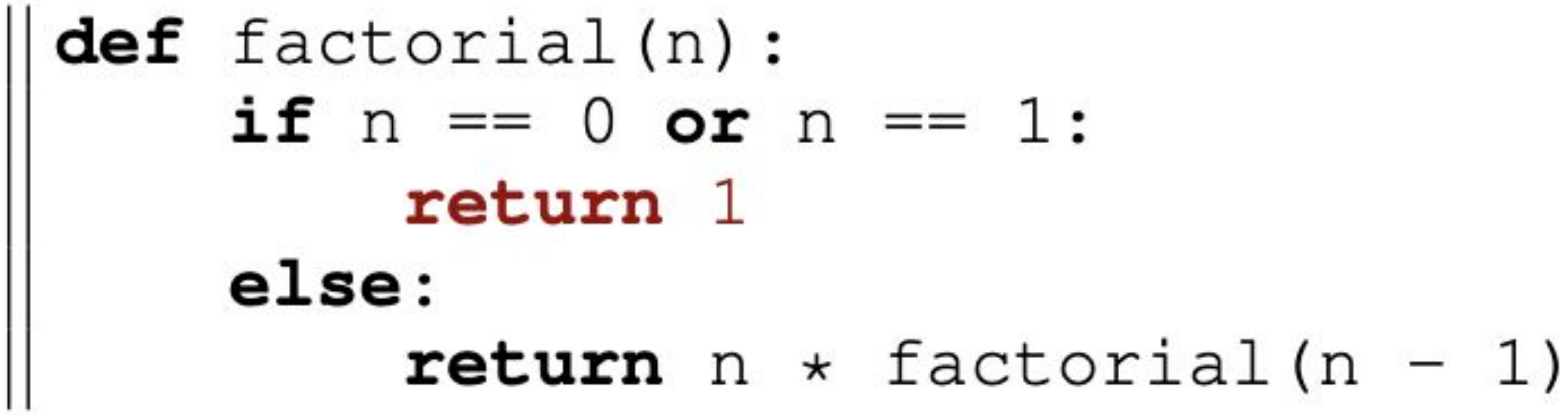
```
  """ Minimize a loss function using gradient.
  Args:
     build_loss_fn: a function that returns a loss tensor for a mini-batch of examples.
     num_steps: number of gradient descent steps to perform.
     optimizer: an optimizer to use when minimizing the loss function. If None, will use Adam
  """
```

Encode code in the same way during inference

https://arxiv.org/pdf/2207.14255

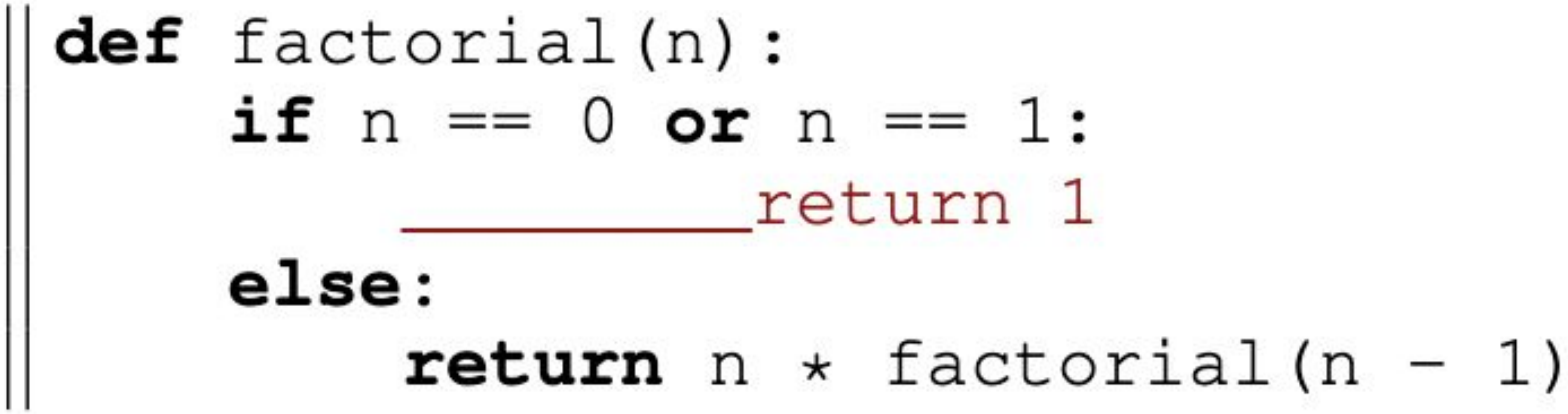
# Many chat models struggle with correct formatting

Correct Outcome



```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
```

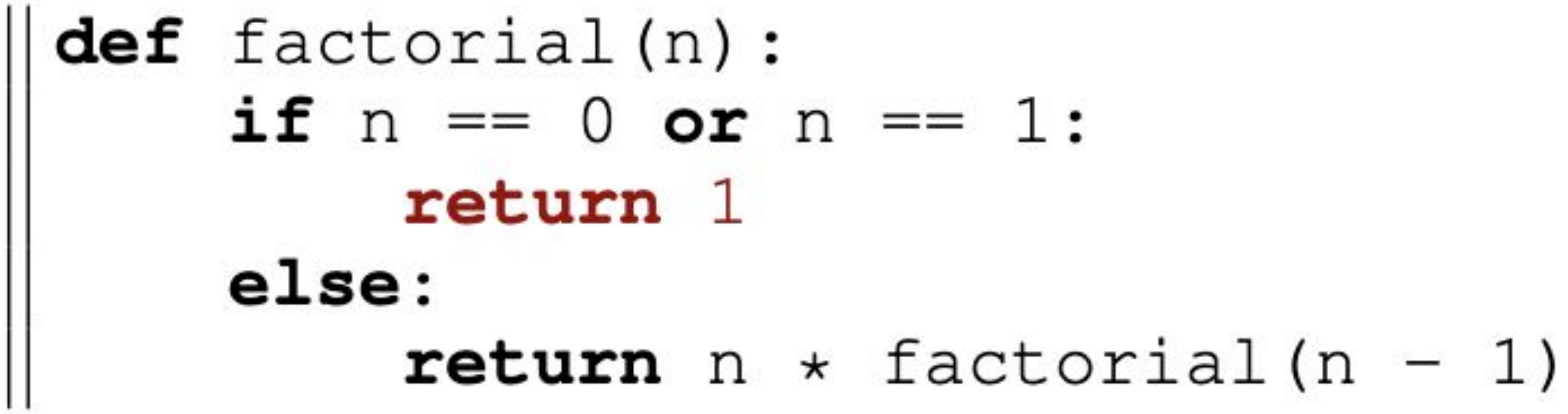
Indent Error



```
def factorial(n):
    if n == 0 or n == 1:
        ________return 1
    else:
        return n * factorial(n - 1)
```

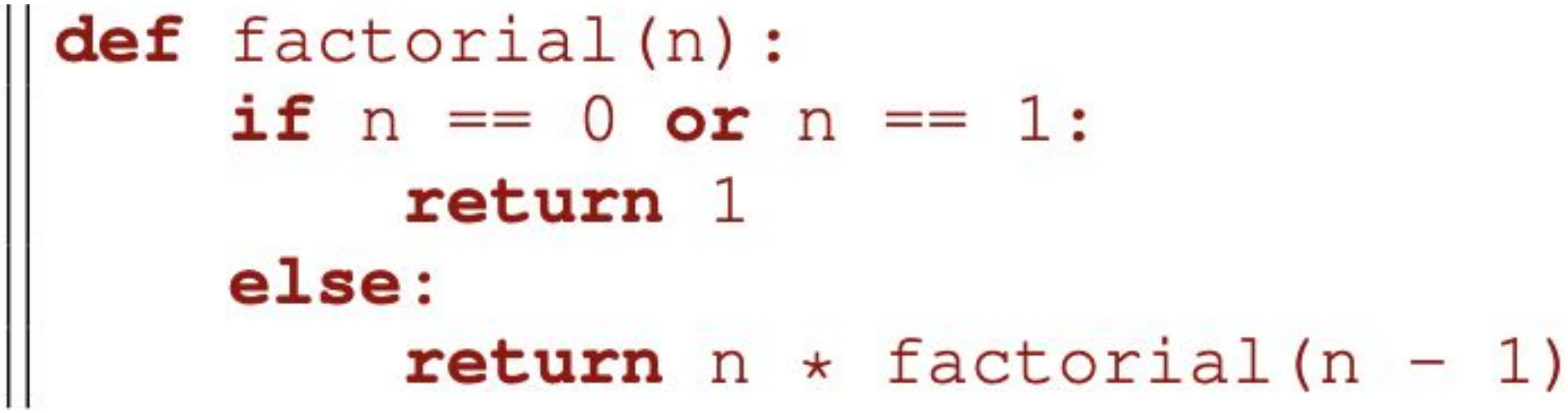
# Instead of generating completions, we ask models to generate code snippets

Correct Outcome



```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
```

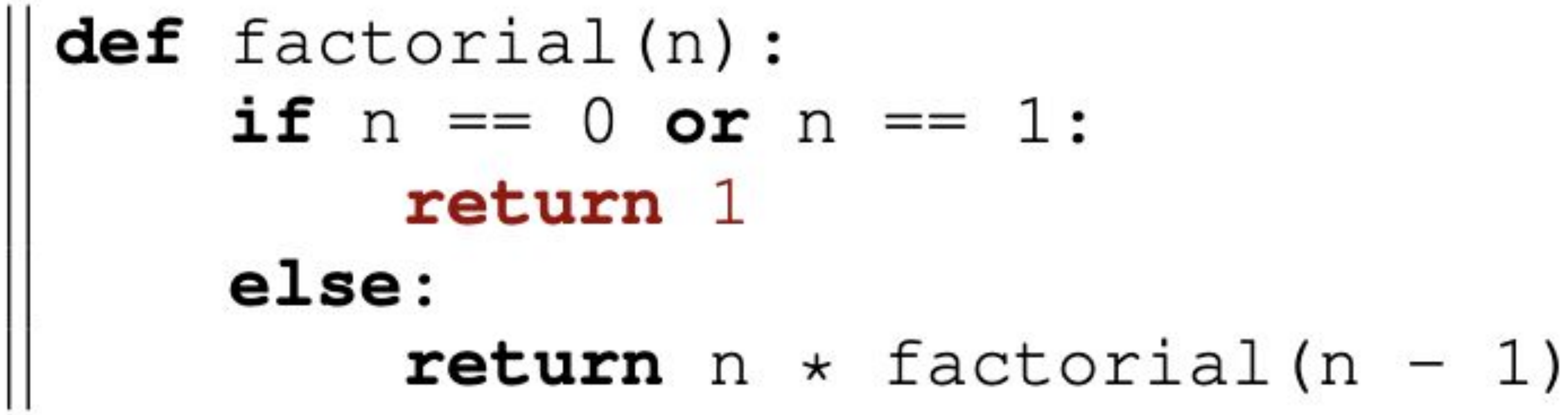
Generate the entire snippet



```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
```

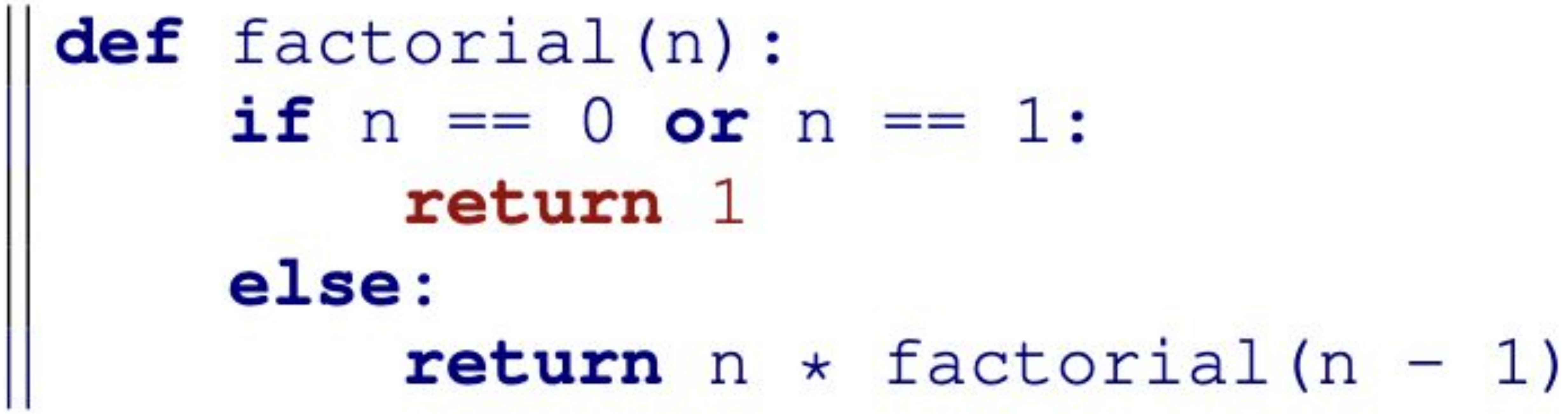
# We post-process the generated snippet to remove overlaps

Correct Outcome



```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
```

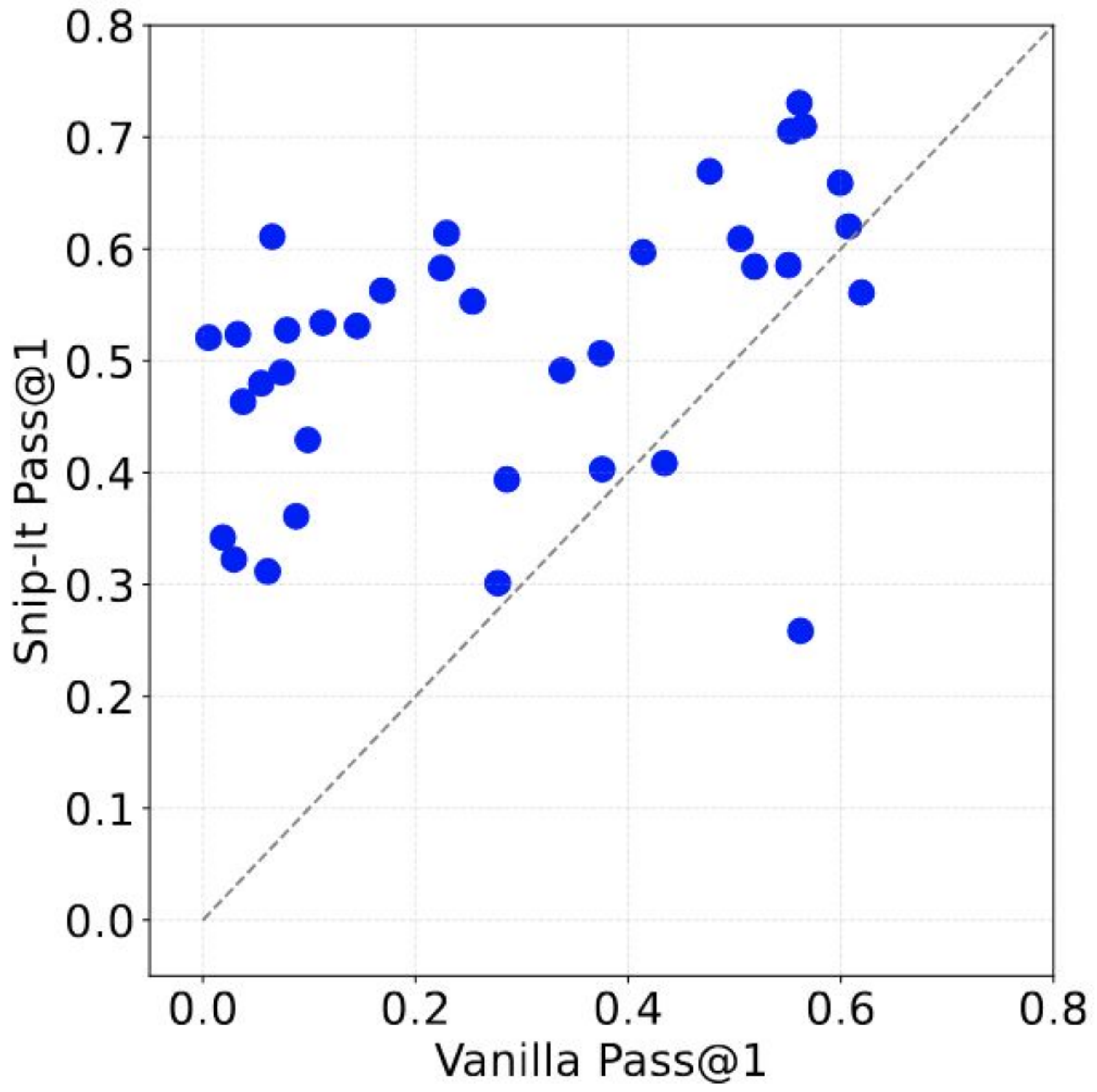
Remove any overlapping text



```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
```

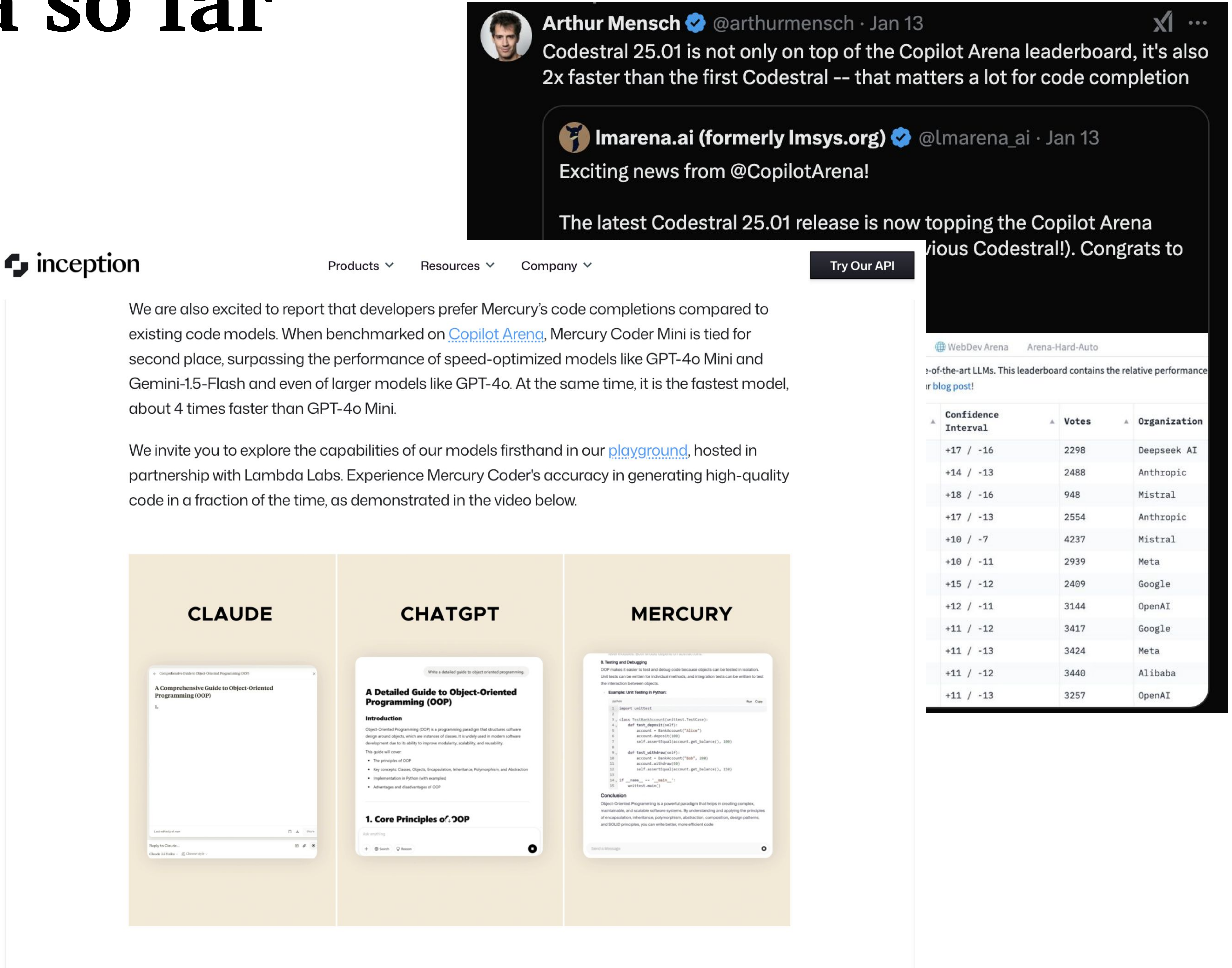
# Simple solution to allow chat models to perform FiM!

- Performance on benchmarks like HumanEval infilling drastically improves model performance (and reduces indentation or formatting issues).



# Copilot Arena so far

- >30k battles
- >4.5k users contribute
- Dozen models in the a
- We even help evaluate



**Arthur Mensch** @arthurmensch · Jan 13

Codestral 25.01 is not only on top of the Copilot Arena leaderboard, it's also 2x faster than the first Codestral -- that matters a lot for code completion

**lmarena.ai (formerly lmsys.org)** @lmarena_ai · Jan 13

Exciting news from @CopilotArena!

The latest Codestral 25.01 release is now topping the Copilot Arena

vious Codestral!). Congrats to

-of-the-art LLMs. This leaderboard contains the relative performance

r blog post!

| Confidence Interval | Votes | Organization |
|---|---|---|
| +17 / -16 | 2298 | Deepseek AI |
| +14 / -13 | 2488 | Anthropic |
| +18 / -16 | 948 | Mistral |
| +17 / -13 | 2554 | Anthropic |
| +10 / -7 | 4237 | Mistral |
| +10 / -11 | 2939 | Meta |
| +15 / -12 | 2409 | Google |
| +12 / -11 | 3144 | OpenAI |
| +11 / -12 | 3417 | Google |
| +11 / -13 | 3424 | Meta |
| +11 / -12 | 3440 | Alibaba |
| +11 / -13 | 3257 | OpenAI |

We are also excited to report that developers prefer Mercury's code completions compared to existing code models. When benchmarked on Copilot Arena, Mercury Coder Mini is tied for second place, surpassing the performance of speed-optimized models like GPT-4o Mini and Gemini-1.5-Flash and even of larger models like GPT-4o. At the same time, it is the fastest model, about 4 times faster than GPT-4o Mini.

We invite you to explore the capabilities of our models firsthand in our playground, hosted in partnership with Lambda Labs. Experience Mercury Coder's accuracy in generating high-quality code in a fraction of the time, as demonstrated in the video below.

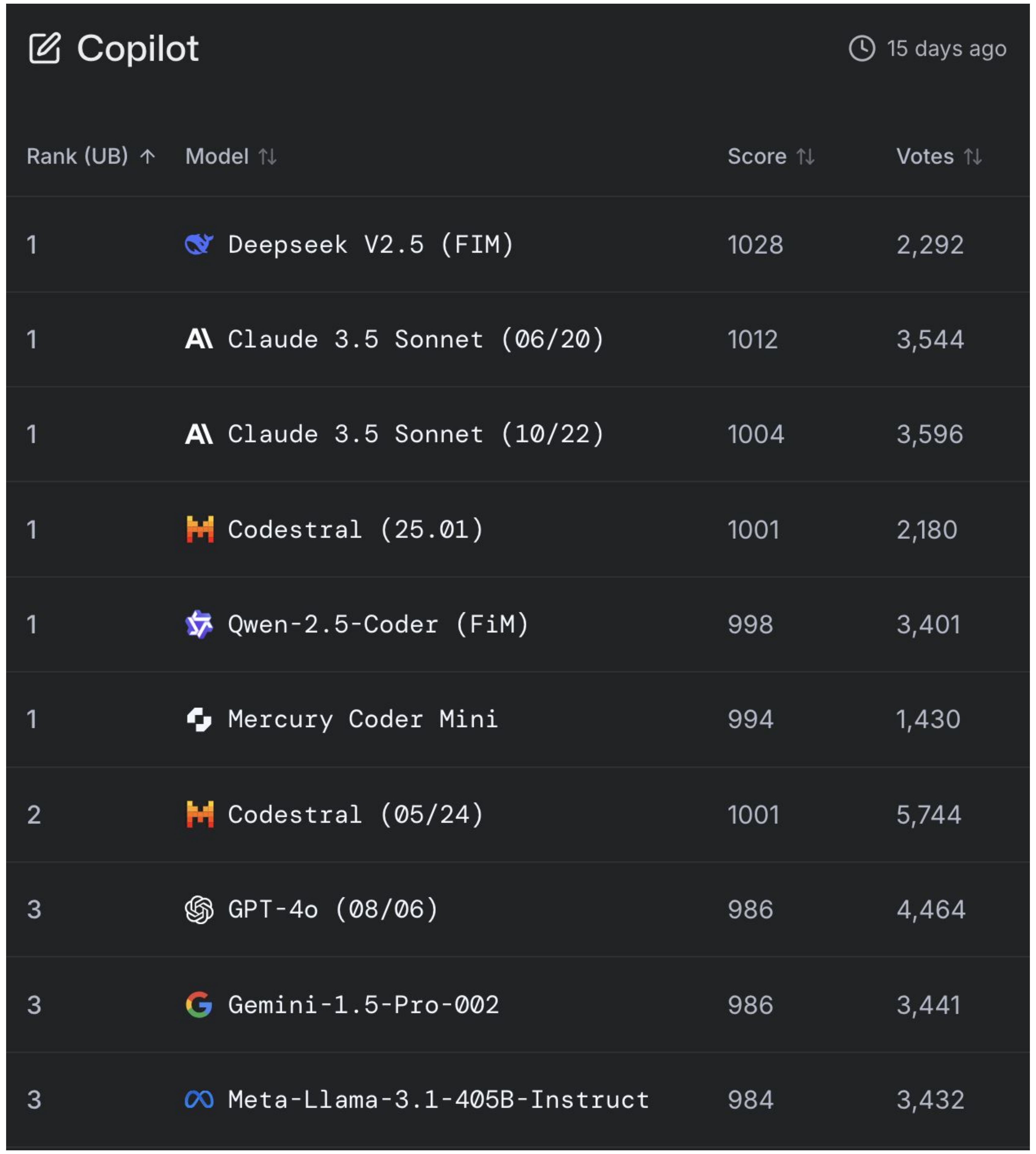
CLAUDE
CHATGPT
MERCURY
A Comprehensive Guide to Object-Oriented Programming (OOP)
A Detailed Guide to Object-Oriented Programming (OOP)
Introduction
1. Core Principles of OOP

# Leaderboard Computation

- **Data**
  - Models: $i, j \in [M]$ (top, bottom)
  - Battle: $X_i = (0,0,\underset{\substack{\uparrow \\ i}}{1}, 0, \underset{\substack{\uparrow \\ j}}{-1}, \dots)$
  - Vote:
    $$Y_i = \begin{cases} 1 & \text{vote for model i} \\ 0 & \text{vote for model j} \end{cases}$$

- **Modeling**
  - Fit Bradley-Terry model (probability model $i$ beats $j$): $p_{ij} = \dfrac{e^{\beta_i}}{e^{\beta_i} + e^{\beta_j}}$
  - Logistic regression:
    $$\hat{\beta} = \arg \min_{\beta \in \mathbb{R}^M} \frac{1}{n} \sum_{i=1}^{n} \mathrm{CE}(\sigma(X_i^\top \beta), Y_i)$$
  - Bootstrap samples to estimate CIs

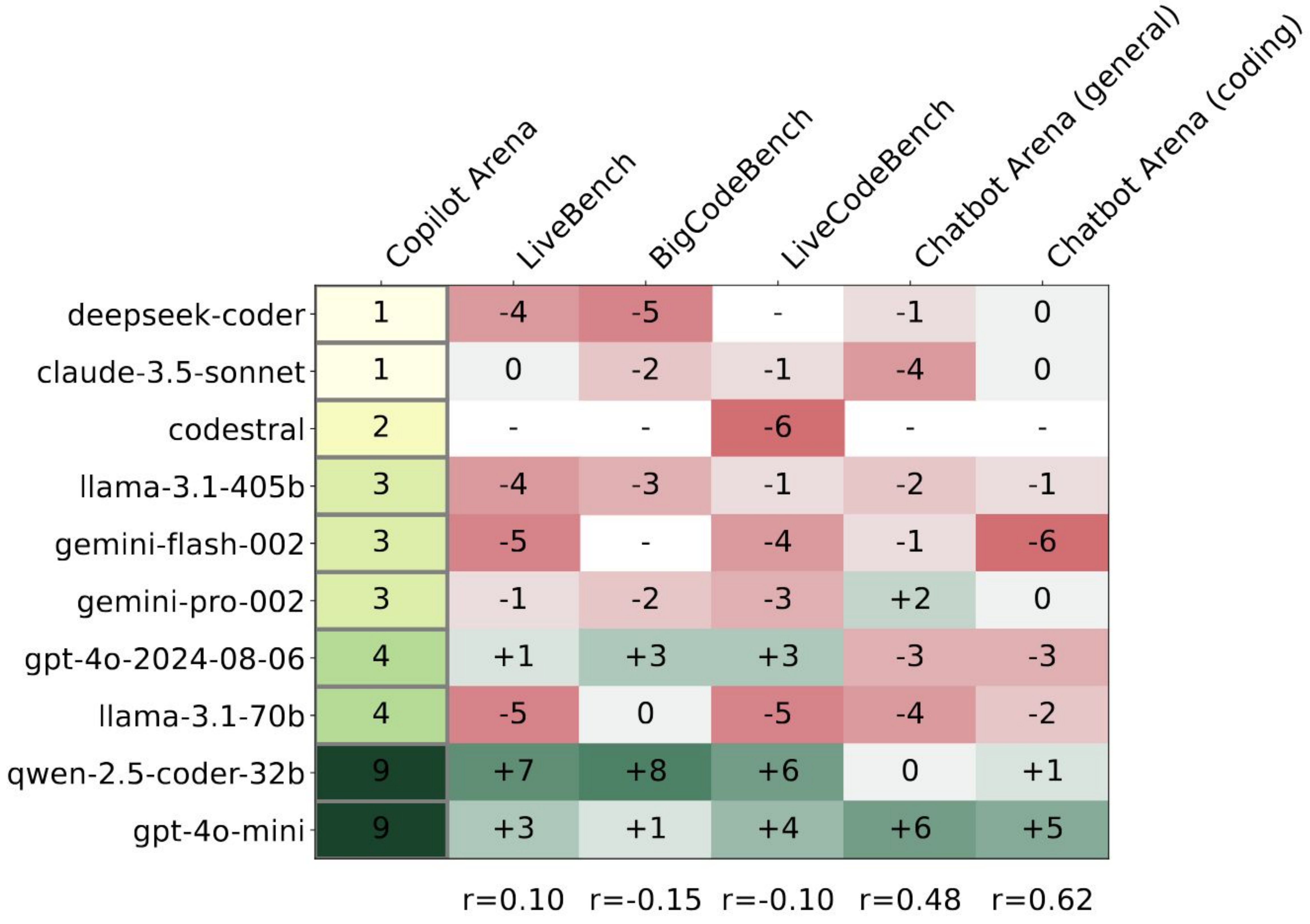
# Copilot Arena Leaderboard



Copilot

15 days ago

| Rank (UB) | Model | Score | Votes |
|---|---|---|---|
| 1 | Deepseek V2.5 (FIM) | 1028 | 2,292 |
| 1 | Claude 3.5 Sonnet (06/20) | 1012 | 3,544 |
| 1 | Claude 3.5 Sonnet (10/22) | 1004 | 3,596 |
| 1 | Codestral (25.01) | 1001 | 2,180 |
| 1 | Qwen-2.5-Coder (FiM) | 998 | 3,401 |
| 1 | Mercury Coder Mini | 994 | 1,430 |
| 2 | Codestral (05/24) | 1001 | 5,744 |
| 3 | GPT-4o (08/06) | 986 | 4,464 |
| 3 | Gemini-1.5-Pro-002 | 986 | 3,441 |
| 3 | Meta-Llama-3.1-405B-Instruct | 984 | 3,432 |



| | Copilot Arena | LiveBench | BigCodeBench | LiveCodeBench | Chatbot Arena (general) | Chatbot Arena (coding) |
|---|---|---|---|---|---|---|
| deepseek-coder | 1 | -4 | -5 | - | -1 | 0 |
| claude-3.5-sonnet | 1 | 0 | -2 | -1 | -4 | 0 |
| codestral | 2 | - | - | -6 | - | - |
| llama-3.1-405b | 3 | -4 | -3 | -1 | -2 | -1 |
| gemini-flash-002 | 3 | -5 | - | -4 | -1 | -6 |
| gemini-pro-002 | 3 | -1 | -2 | -3 | +2 | 0 |
| gpt-4o-2024-08-06 | 4 | +1 | +3 | +3 | -3 | -3 |
| llama-3.1-70b | 4 | -5 | 0 | -5 | -4 | -2 |
| qwen-2.5-coder-32b | 9 | +7 | +8 | +6 | 0 | +1 |
| gpt-4o-mini | 9 | +3 | +1 | +4 | +6 | +5 |
| | | r=0.10 | r=-0.15 | r=-0.10 | r=0.48 | r=0.62 |

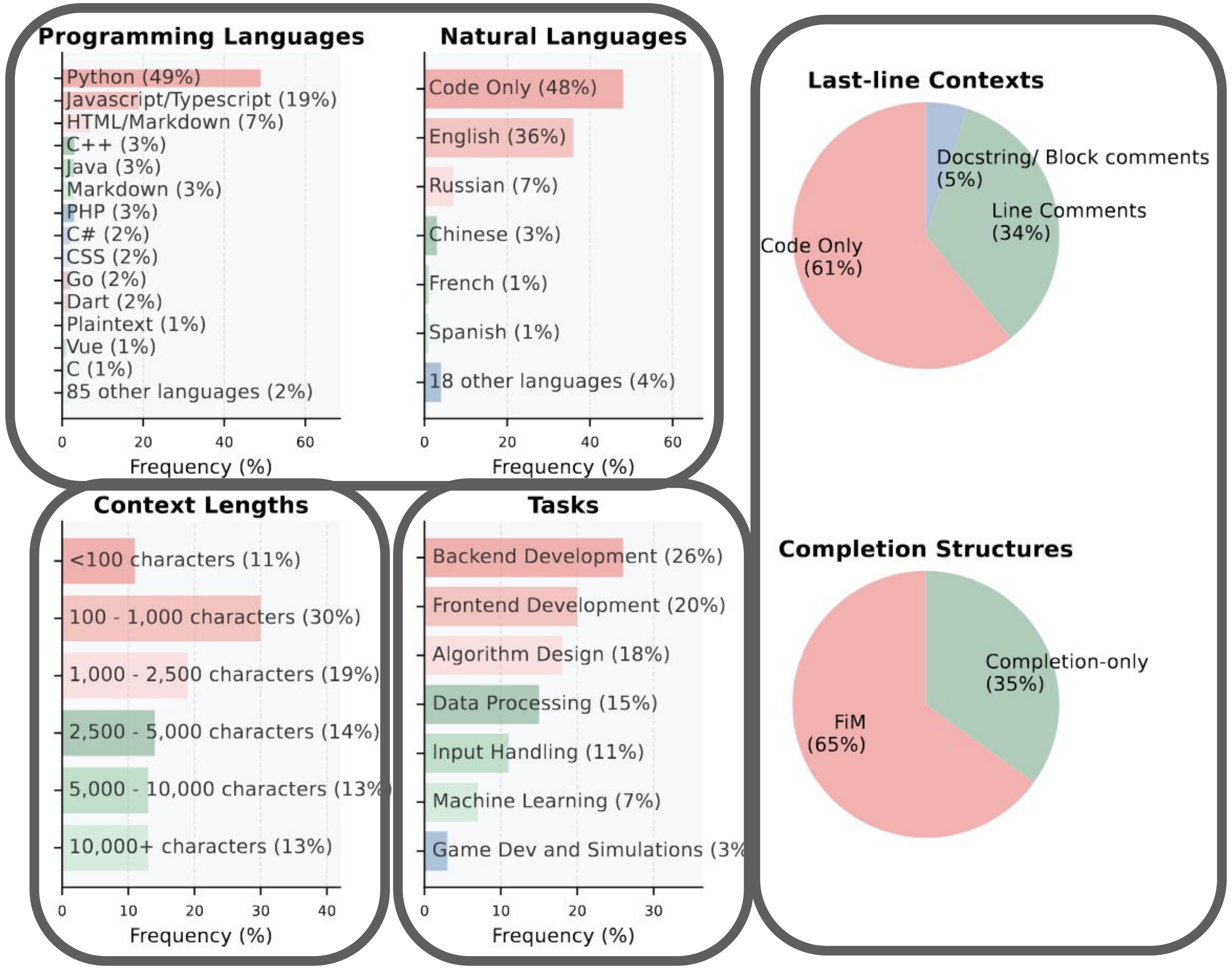
Live at lmarena.ai

# Real-world “data distribution”

# Comparison to prior evals

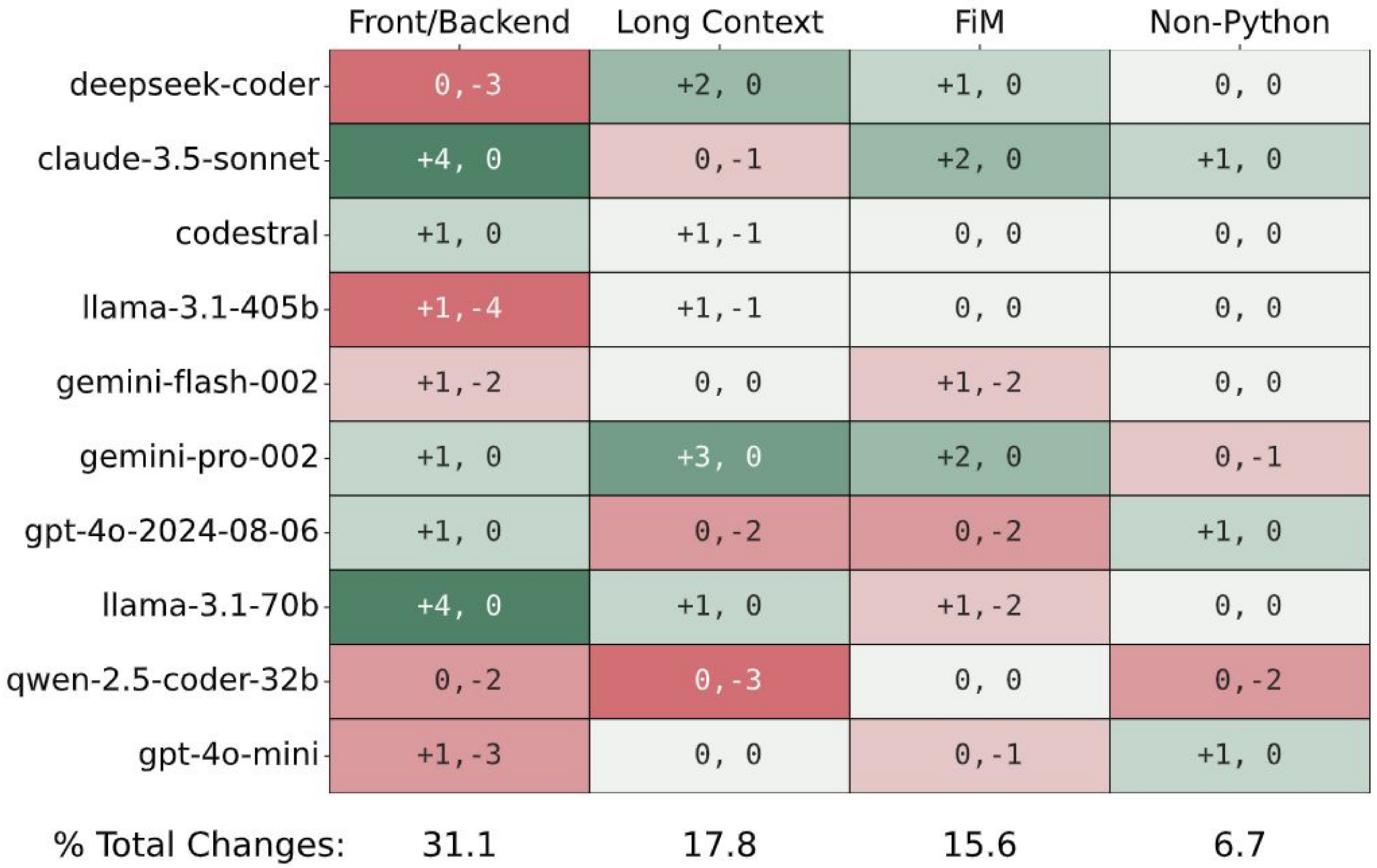
Many existing static benchmarks only evaluate Python, interview-style coding problems that are written in English.

Copilot Arena captures the long tail of context lengths



40% of Chatbot Arena's coding tasks contain code context and only 2.6% focus on infilling

# Which models best align with user preferences?



| | Front/Backend | Long Context | FiM | Non-Python |
|---|---|---|---|---|
| deepseek-coder | 0,-3 | +2, 0 | +1, 0 | 0, 0 |
| claude-3.5-sonnet | +4, 0 | 0,-1 | +2, 0 | +1, 0 |
| codestral | +1, 0 | +1,-1 | 0, 0 | 0, 0 |
| llama-3.1-405b | +1,-4 | +1,-1 | 0, 0 | 0, 0 |
| gemini-flash-002 | +1,-2 | 0, 0 | +1,-2 | 0, 0 |
| gemini-pro-002 | +1, 0 | +3, 0 | +2, 0 | 0,-1 |
| gpt-4o-2024-08-06 | +1, 0 | 0,-2 | 0,-2 | +1, 0 |
| llama-3.1-70b | +4, 0 | +1, 0 | +1,-2 | 0, 0 |
| qwen-2.5-coder-32b | 0,-2 | 0,-3 | 0, 0 | 0,-2 |
| gpt-4o-mini | +1,-3 | 0, 0 | 0,-1 | +1, 0 |
| % Total Changes: | 31.1 | 17.8 | 15.6 | 6.7 |

$$\Delta_{i,j} = \mathbb{1}[(W_{i,j}(X) - W_{i,j}(\tilde{X})) > \epsilon]$$

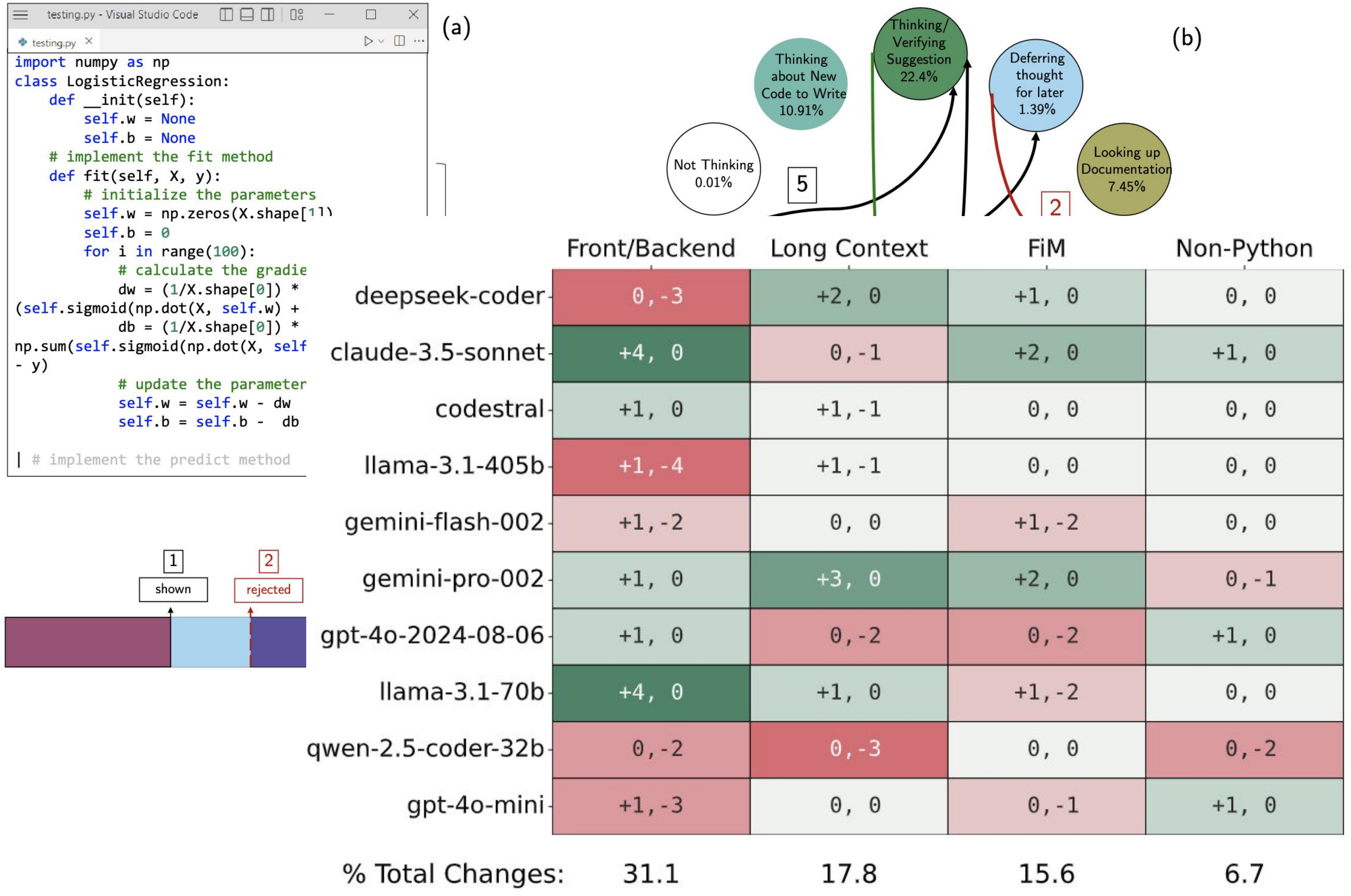
- Downstream task significantly affects user preference, while programming languages have little effect.
- Smaller models tend to perform better on data similar to static benchmarks.

# TLDR

- Existing evaluations do not necessarily correlate well with in-the-wild preferences.
- Model performance is affected by task, context, and code structure. No model that is “one-size-fits-all.”
- Diverse and realistic human preference data is essential for effective code generation models.
- We also now have a lot of interesting data to dig into!

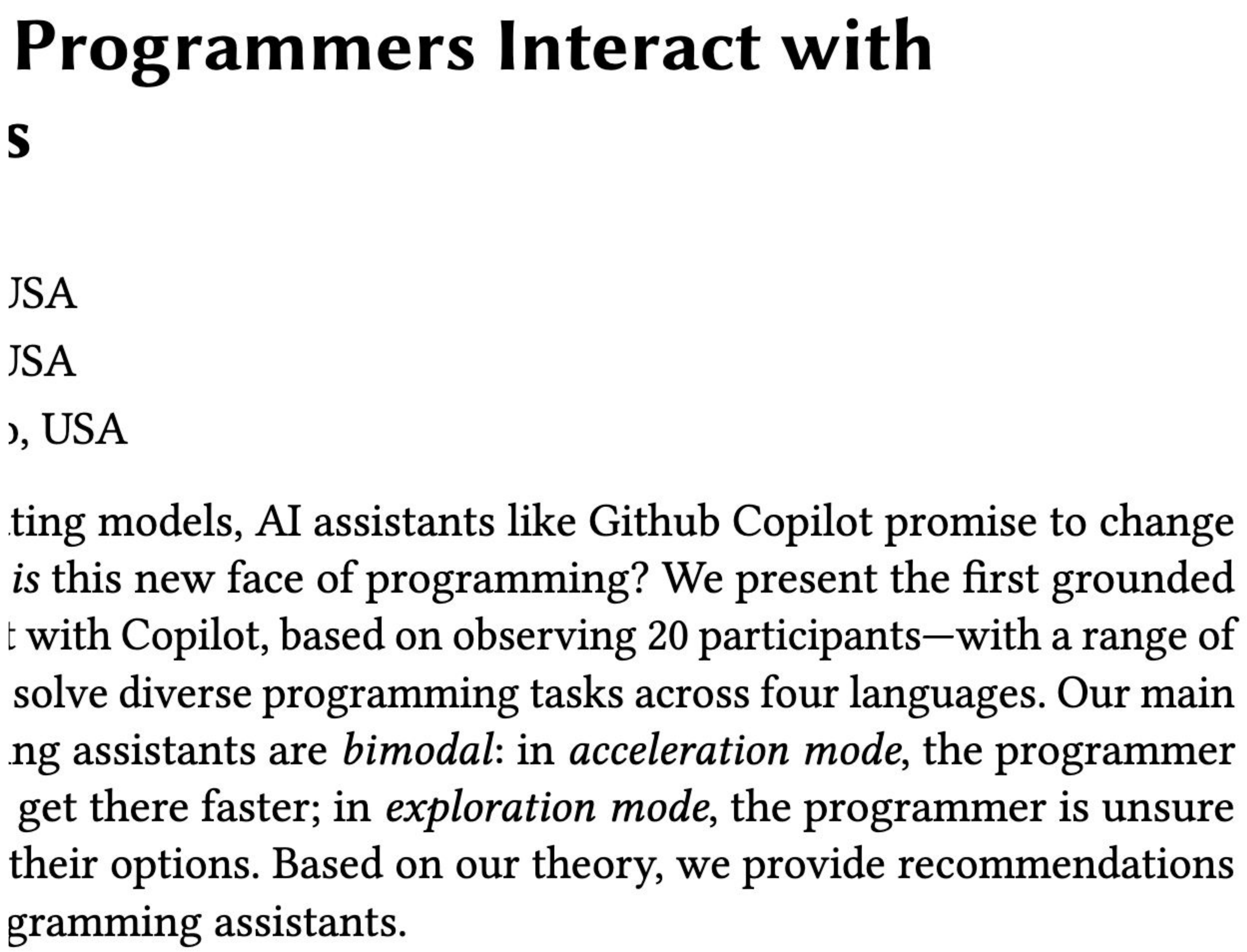
# Evaluating agents

# A lot of work has been done to understand copilot usage



```
import numpy as np
class LogisticRegression:
    def __init(self):
        self.w = None
        self.b = None
    # implement the fit method
    def fit(self, X, y):
        # initialize the parameters
        self.w = np.zeros(X.shape[1])
        self.b = 0
        for i in range(100):
            # calculate the gradie
            dw = (1/X.shape[0]) *
(self.sigmoid(np.dot(X, self.w) +
            db = (1/X.shape[0]) *
np.sum(self.sigmoid(np.dot(X, self
- y)

            # update the parameter
            self.w = self.w - dw
            self.b = self.b -  db

| # implement the predict method
```

| | Front/Backend | Long Context | FiM | Non-Python |
|---|---|---|---|---|
| deepseek-coder | 0, -3 | +2, 0 | +1, 0 | 0, 0 |
| claude-3.5-sonnet | +4, 0 | 0, -1 | +2, 0 | +1, 0 |
| codestral | +1, 0 | +1, -1 | 0, 0 | 0, 0 |
| llama-3.1-405b | +1, -4 | +1, -1 | 0, 0 | 0, 0 |
| gemini-flash-002 | +1, -2 | 0, 0 | +1, -2 | 0, 0 |
| gemini-pro-002 | +1, 0 | +3, 0 | +2, 0 | 0, -1 |
| gpt-4o-2024-08-06 | +1, 0 | 0, -2 | 0, -2 | +1, 0 |
| llama-3.1-70b | +4, 0 | +1, 0 | +1, -2 | 0, 0 |
| qwen-2.5-coder-32b | 0, -2 | 0, -3 | 0, 0 | 0, -2 |
| gpt-4o-mini | +1, -3 | 0, 0 | 0, -1 | +1, 0 |
| % Total Changes: | 31.1 | 17.8 | 15.6 | 6.7 |

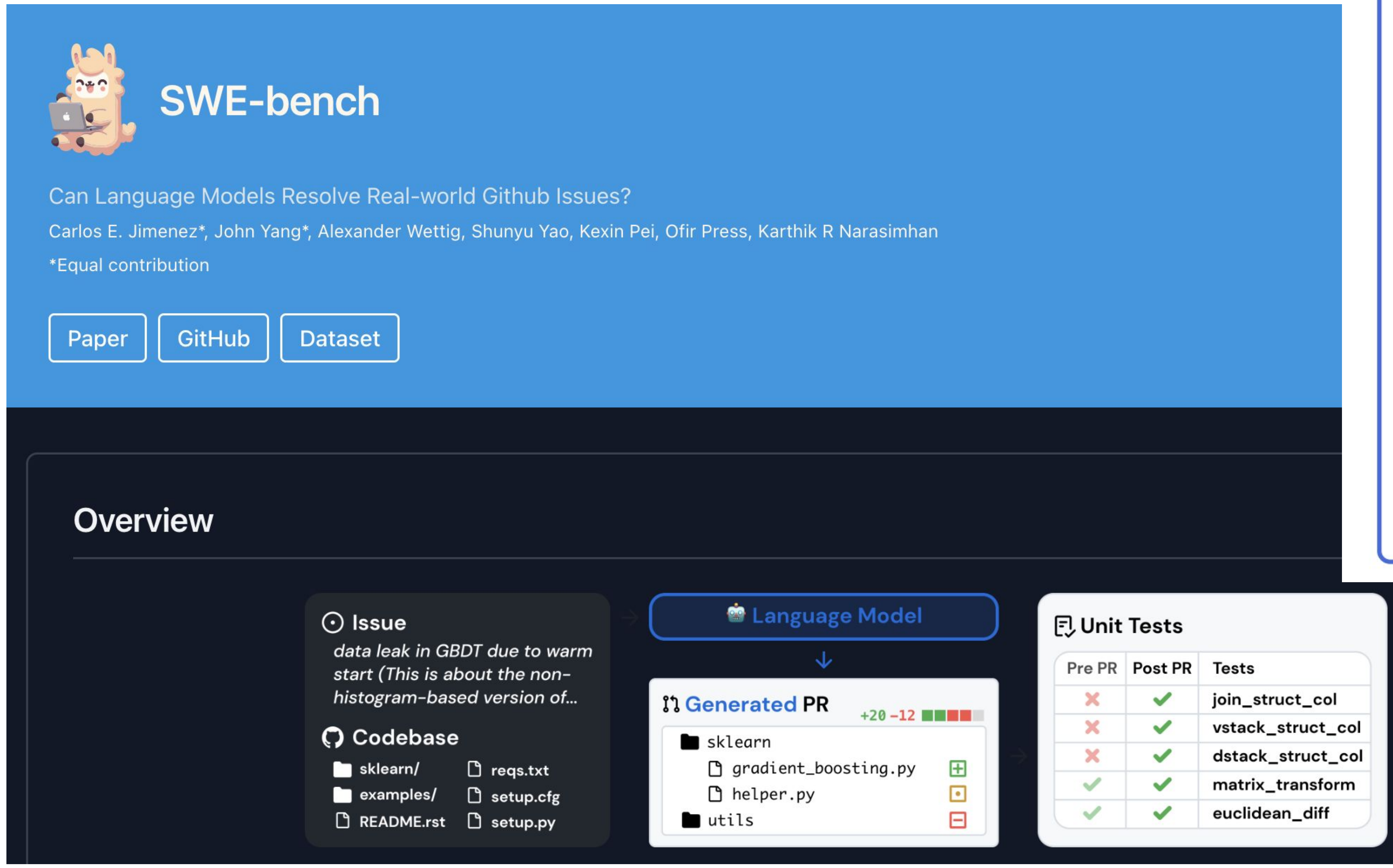


## Programmers Interact with
s

JSA
JSA
), USA

ting models, AI assistants like Github Copilot promise to change
*is* this new face of programming? We present the first grounded
t with Copilot, based on observing 20 participants—with a range of
solve diverse programming tasks across four languages. Our main
ng assistants are *bimodal*: in *acceleration mode*, the programmer
get there faster; in *exploration mode*, the programmer is unsure
their options. Based on our theory, we provide recommendations
gramming assistants.

# Current understanding of agent usage



## SWE-bench

Can Language Models Resolve Real-world Github Issues?

Carlos E. Jimenez*, John Yang*, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, Karthik R Narasimhan

*Equal contribution

### Overview

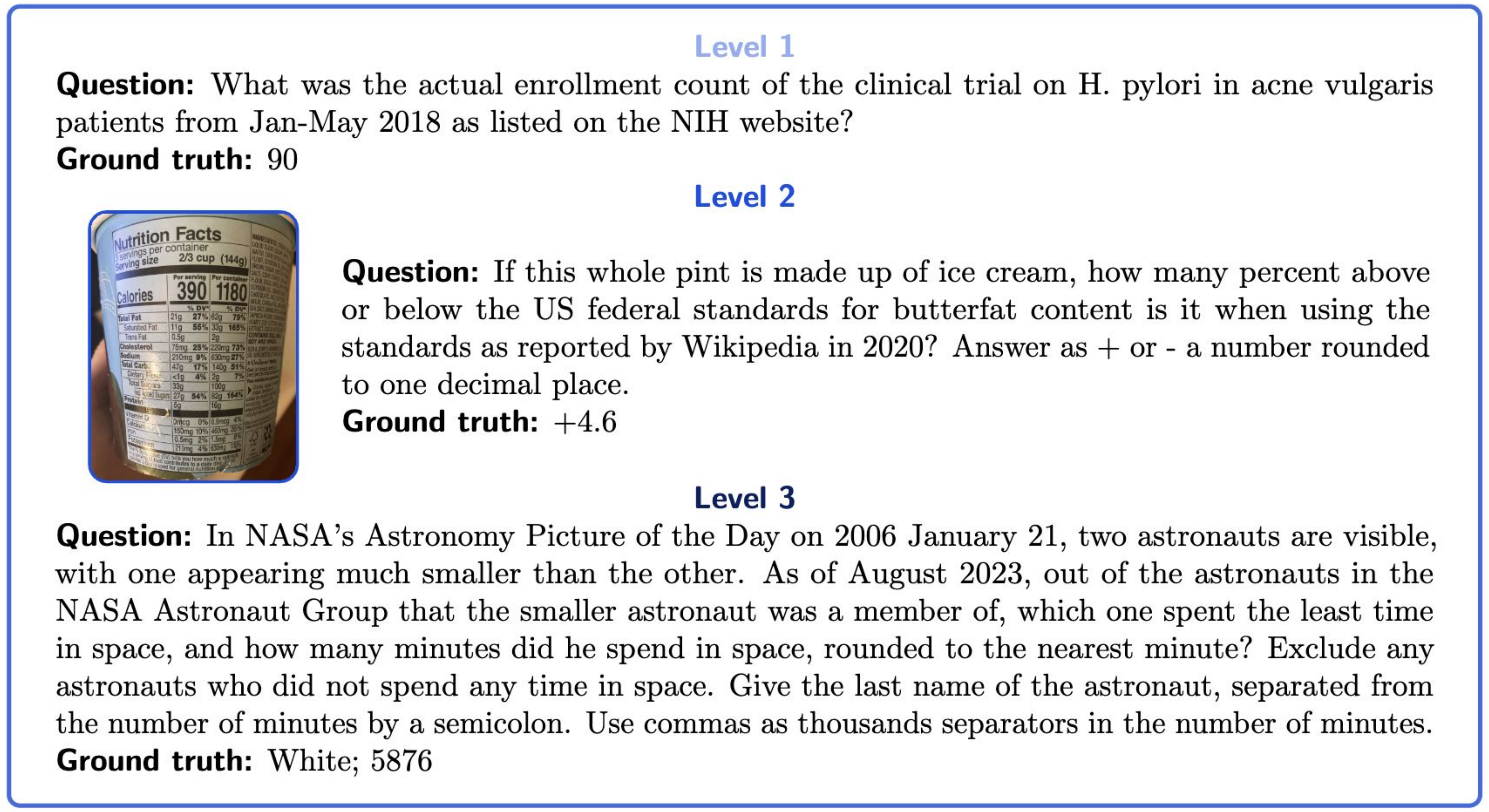
**Issue**

*data leak in GBDT due to warm start (This is about the non-histogram-based version of...*

**Codebase**

**Unit Tests**

| Pre PR | Post PR | Tests |
|---|---|---|
| ✗ | ✓ | join_struct_col |
| ✗ | ✓ | vstack_struct_col |
| ✗ | ✓ | dstack_struct_col |
| ✓ | ✓ | matrix_transform |
| ✓ | ✓ | euclidean_diff |



**Level 1**

**Question:** What was the actual enrollment count of the clinical trial on H. pylori in acne vulgaris patients from Jan-May 2018 as listed on the NIH website?
**Ground truth:** 90

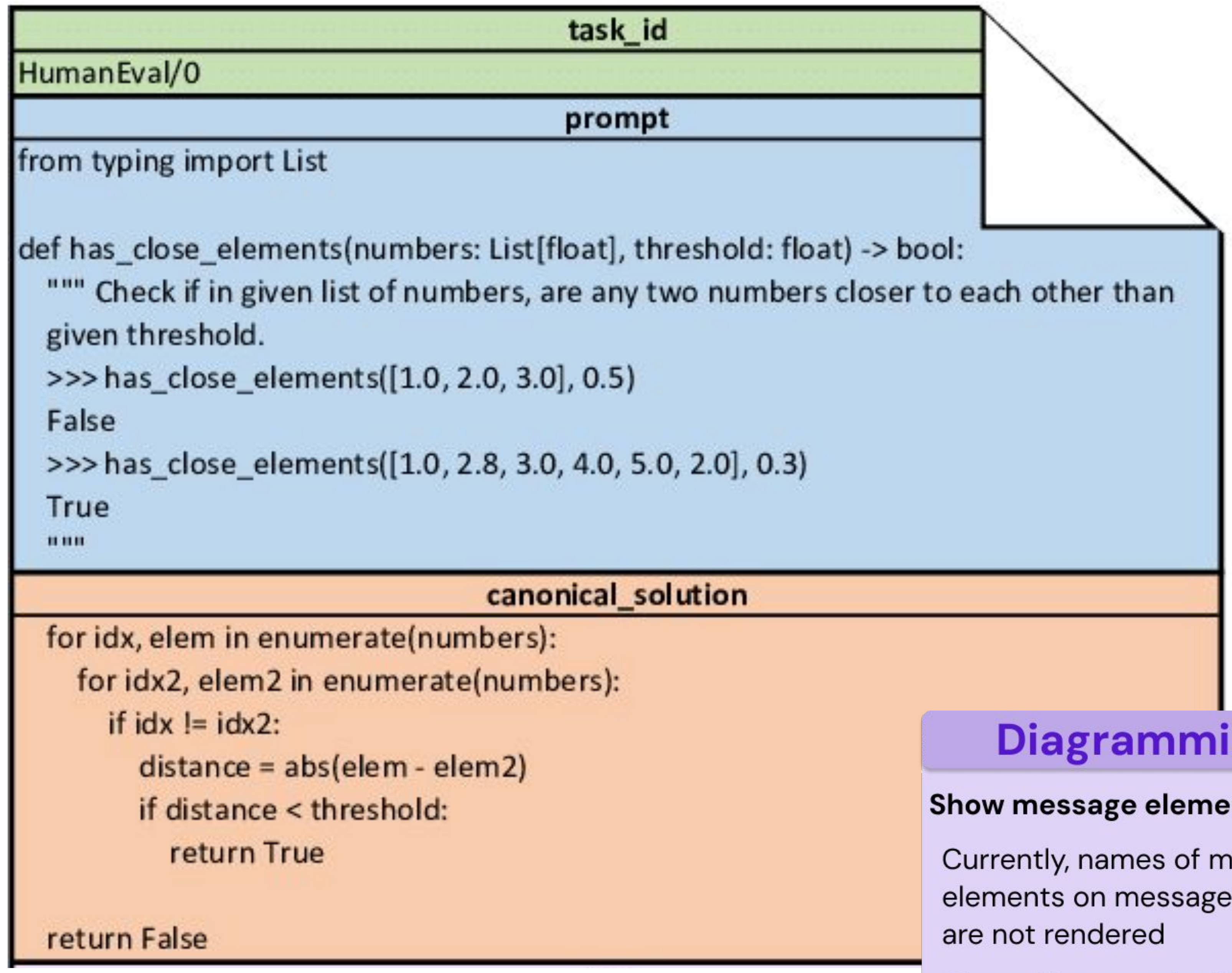
**Level 2**

**Question:** If this whole pint is made up of ice cream, how many percent above or below the US federal standards for butterfat content is it when using the standards as reported by Wikipedia in 2020? Answer as + or - a number rounded to one decimal place.
**Ground truth:** +4.6

**Level 3**

**Question:** In NASA's Astronomy Picture of the Day on 2006 January 21, two astronauts are visible, with one appearing much smaller than the other. As of August 2023, out of the astronauts in the NASA Astronaut Group that the smaller astronaut was a member of, which one spent the least time in space, and how many minutes did he spend in space, rounded to the nearest minute? Exclude any astronauts who did not spend any time in space. Give the last name of the astronaut, separated from the number of minutes by a semicolon. Use commas as thousands separators in the number of minutes.
**Ground truth:** White; 5876

# Scope of tasks has increased



**task_id**

HumanEval/0

**prompt**

```
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
```

**canonical_solution**

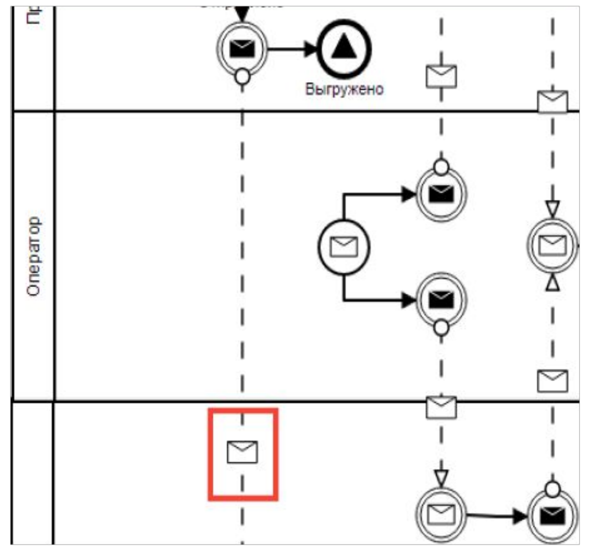
```
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True

    return False
```

**Diagramming**

**Show message element name**

Currently, names of message elements on message flows are not rendered

Given this example diagram [Image] ...



bpmn-js

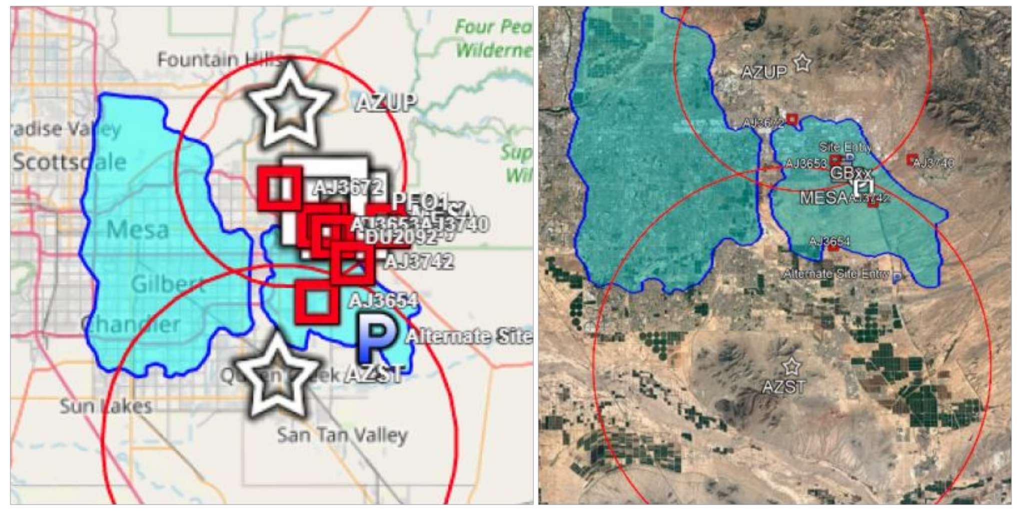
**Interactive Mapping**

**KML Symbol Align/Placement/Size**

There is a bug with the anchor point for some symbols

I've attached a screen clipping from Google Earth to show how it is supposed to look.

[Right Image] ...



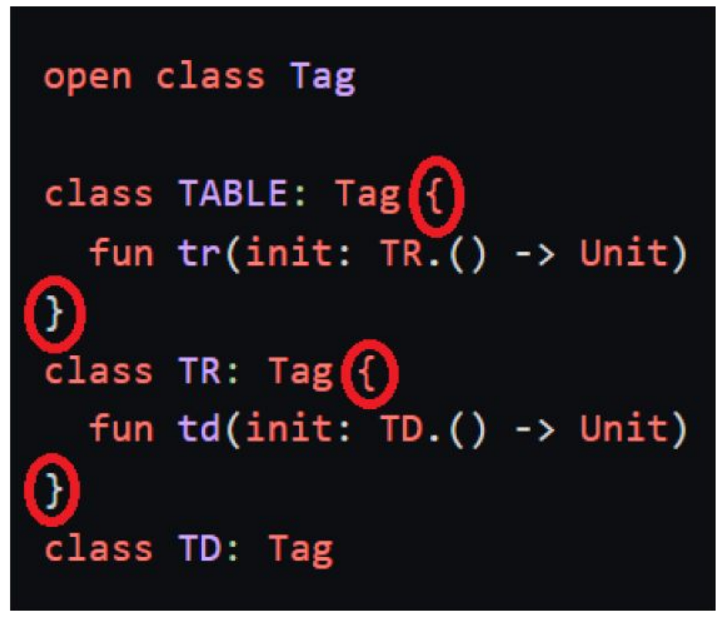
openlayers

**Syntax Highlighting**

**Bracket highlighted with different color in class inheritance context.**

- Reproduced in JSFiddle: https://jsfiddle.net/kkangmj/e7h48w36/7/

[Image] ...



```
open class Tag

class TABLE: Tag{
  fun tr(init: TR.() -> Unit)
}
class TR: Tag{
  fun td(init: TD.() -> Unit)
}
class TD: Tag
```

highlight.js

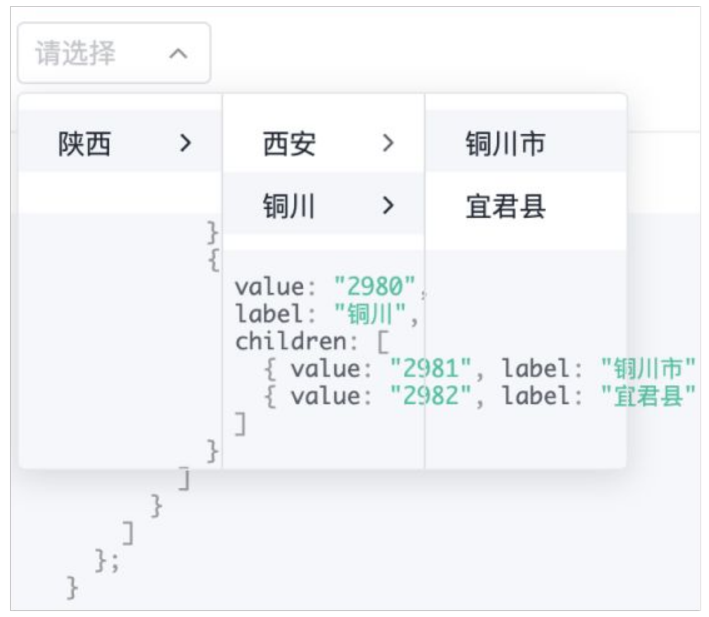
**Web Frameworks**

**[CascaderSelect]使用虚拟滚动时背景色异常**

### Component: CascaderSelect

### Steps to reproduce

[Image] ...



```
        }
        {
          value: "2980",
          label: "铜川",
          children: [
            { value: "2981", label: "铜川市"
            { value: "2982", label: "宜君县"
          ]
        }
      ]
    }
  ]
};
}
```

next

# Demo

[https://app.all-hands.dev/](https://app.all-hands.dev/)

# Demo

OpenHands: Code Less, Make More

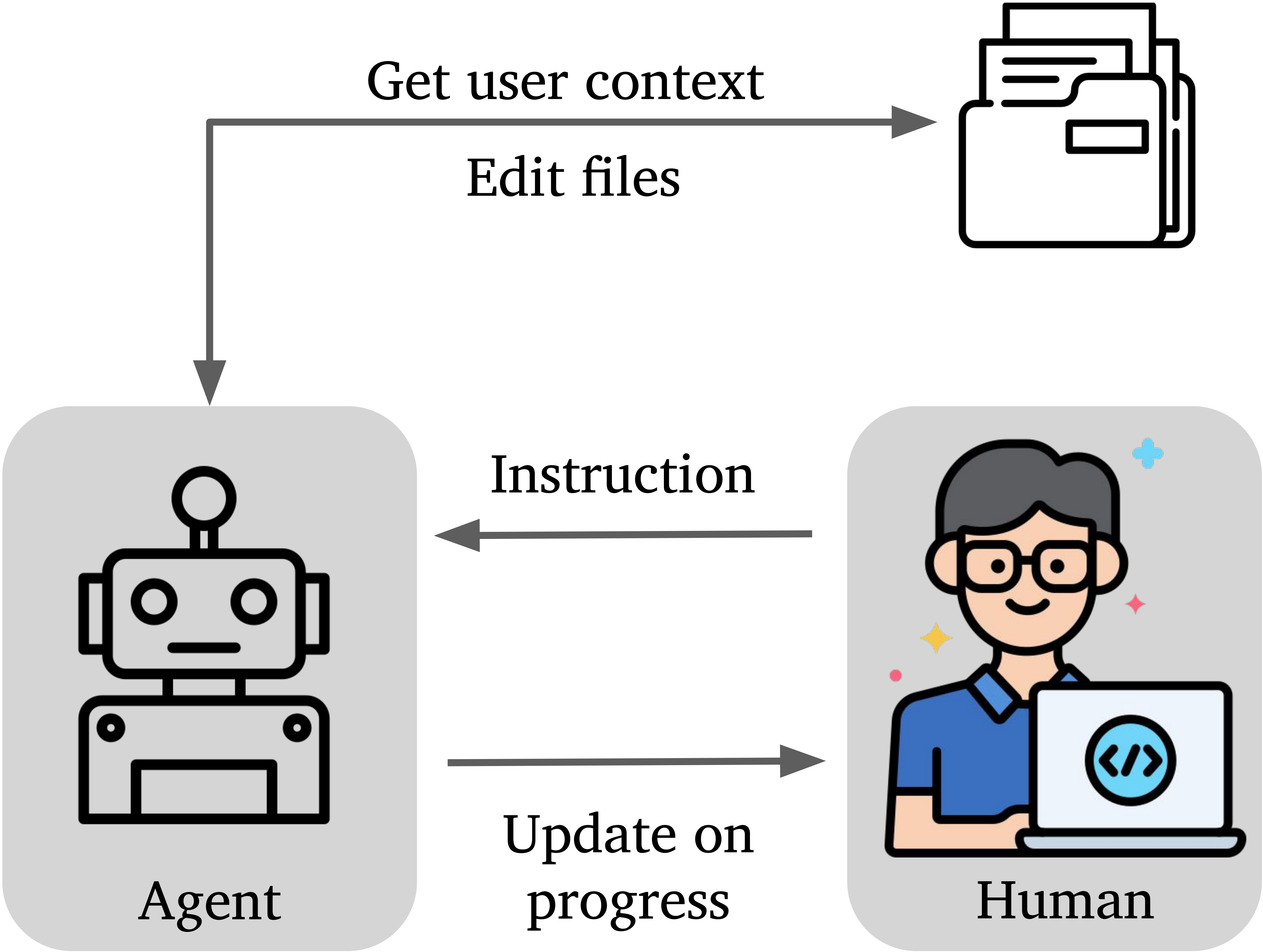
An open-source AI-powered software development platform that automates complex engineering tasks

What is OpenHands?

OpenHands is an open-source autonomous AI software engineer capable of executing complex engineering tasks and collaborating actively with users on software development projects.

Formerly known as OpenDevin, OpenHands is designed to help developers tackle the toil in their backlog, so they can focus on what matters: hard problems, creative challenges, and creative

# Agents still require humans in the loop



Get user context

Edit files

Instruction

Update on progress

Agent

Human

# Now, developers have *options*

GitHub Copilot

ChatGPT

Claude

BY ANTHROP\C

Devin

CURSOR

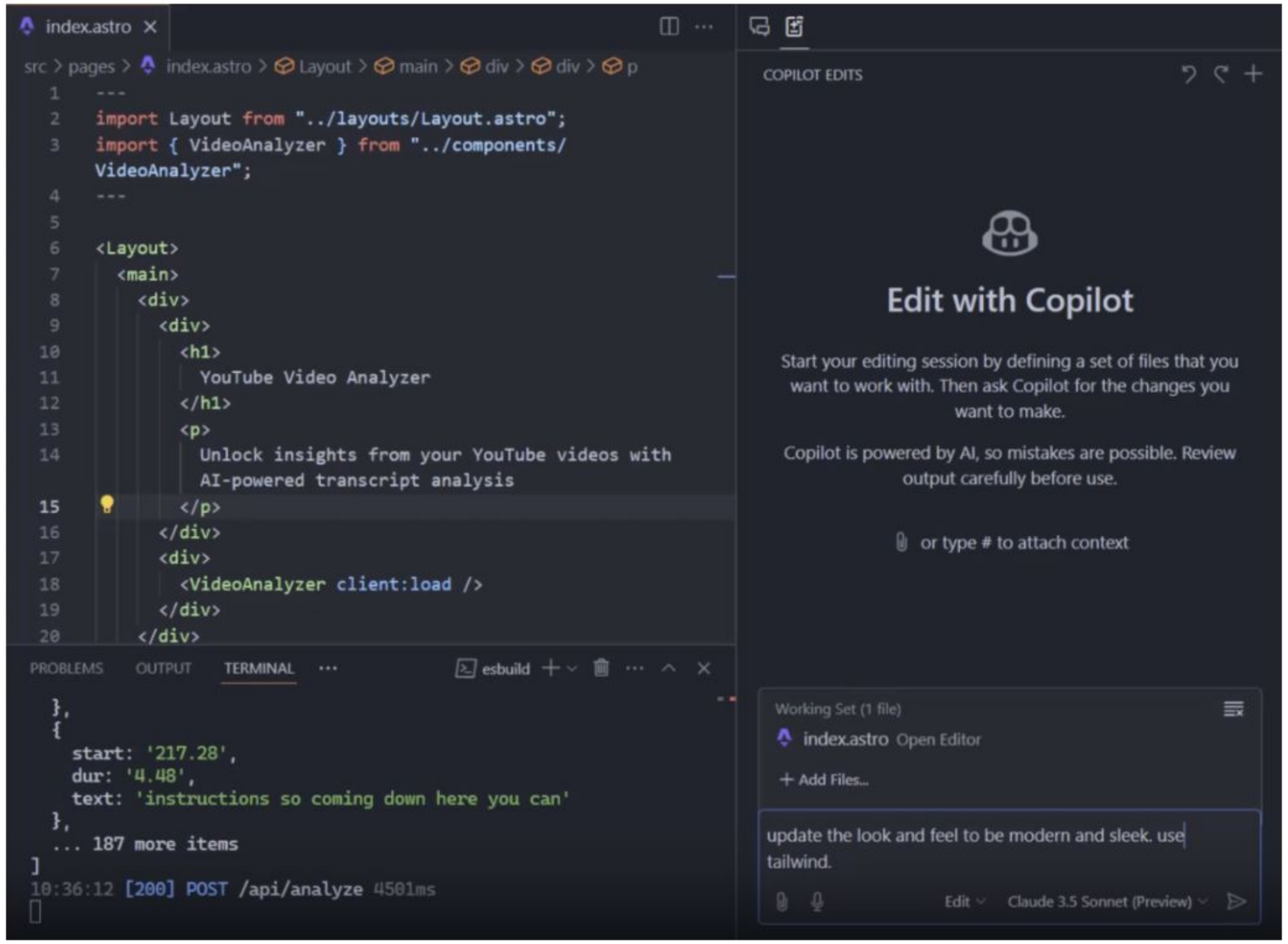
Continue

🙌 OpenHands

```
✻ Welcome to the Claude Code research preview!

CLAUDE
CODE

🎉 Login successful. Press Enter to continue
```

What if we put them head to head?

# Comparing copilots and agents



```
---
import Layout from "../layouts/Layout.astro";
import { VideoAnalyzer } from "../components/
VideoAnalyzer";
---

<Layout>
  <main>
    <div>
      <div>
        <h1>
          YouTube Video Analyzer
        </h1>
        <p>
          Unlock insights from your YouTube videos with
          AI-powered transcript analysis
        </p>
      </div>
      <div>
        <VideoAnalyzer client:load />
      </div>
    </div>
```

```
  },
  {
    start: '217.28',
    dur: '4.48',
    text: 'instructions so coming down here you can'
  },
  ... 187 more items
]
10:36:12 [200] POST /api/analyze 4501ms
```

Start your editing session by defining a set of files that you want to work with. Then ask Copilot for the changes you want to make.

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

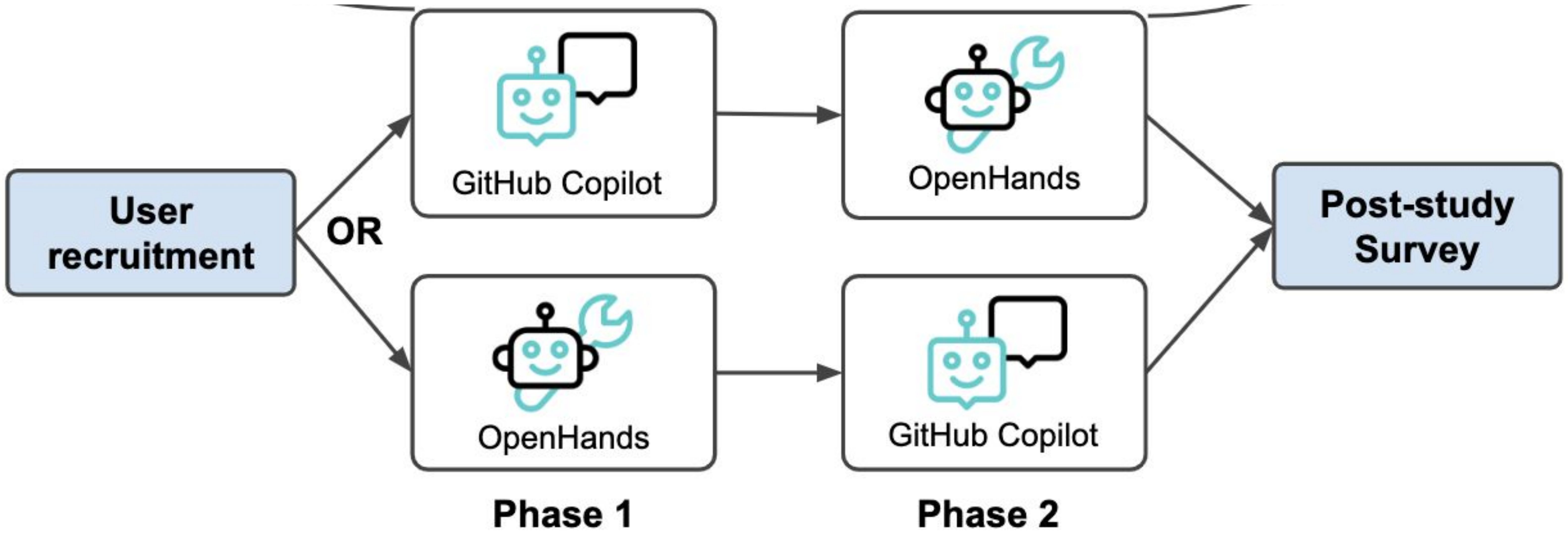
or type # to attach context

update the look and feel to be modern and sleek. use tailwind.

(a) GitHub Copilot

**We recruit participants who are regular users of GitHub Copilot**

# Study design



User recruitment

OR

GitHub Copilot

OpenHands

OpenHands

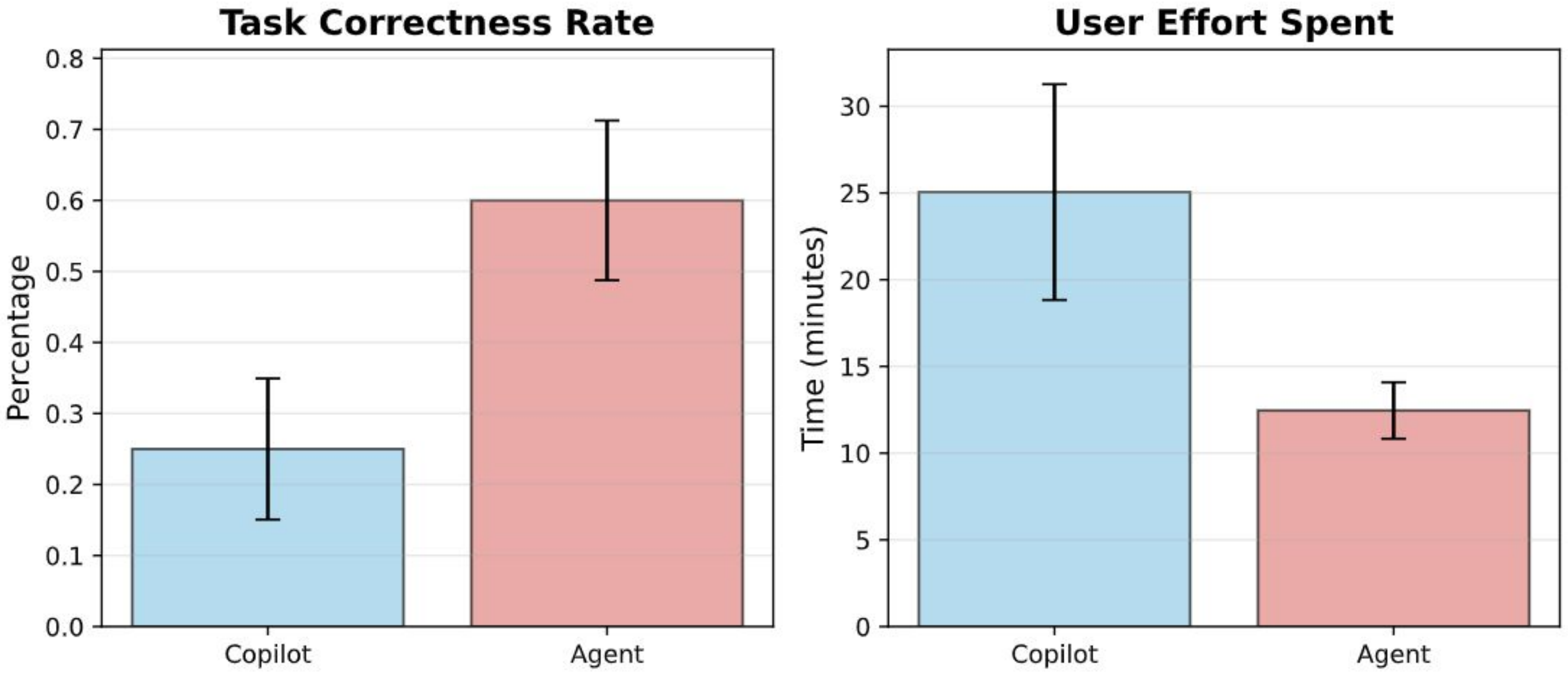
GitHub Copilot

Phase 1

Phase 2

Post-study Survey

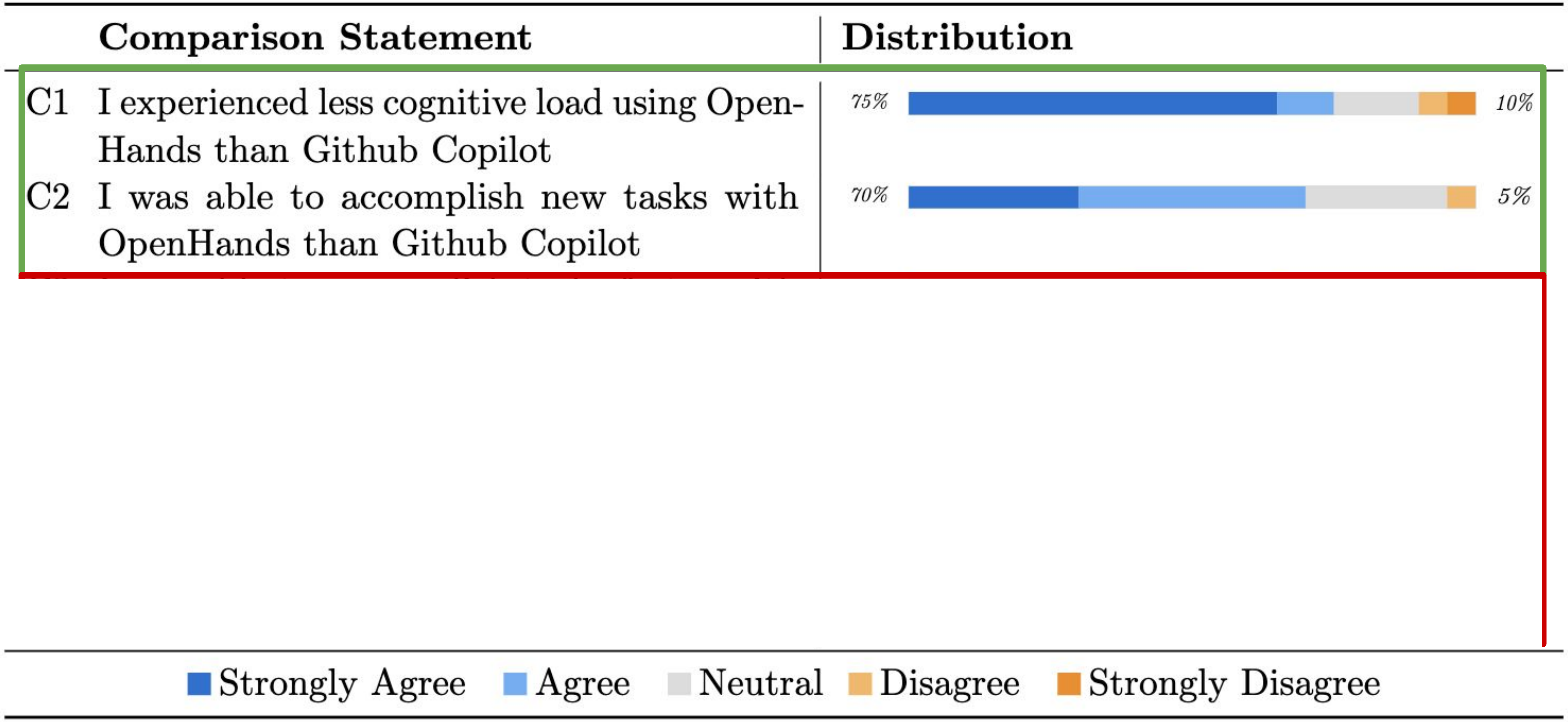
# Summary of Findings



- On average, participants with agents are more productive than with copilots (a 35% increase in task correctness).

- For correctly solved problems, we find a significant difference in user effort between the time spent using copilots and agents (25.1 vs 12.5 minutes).

# Summary of Findings

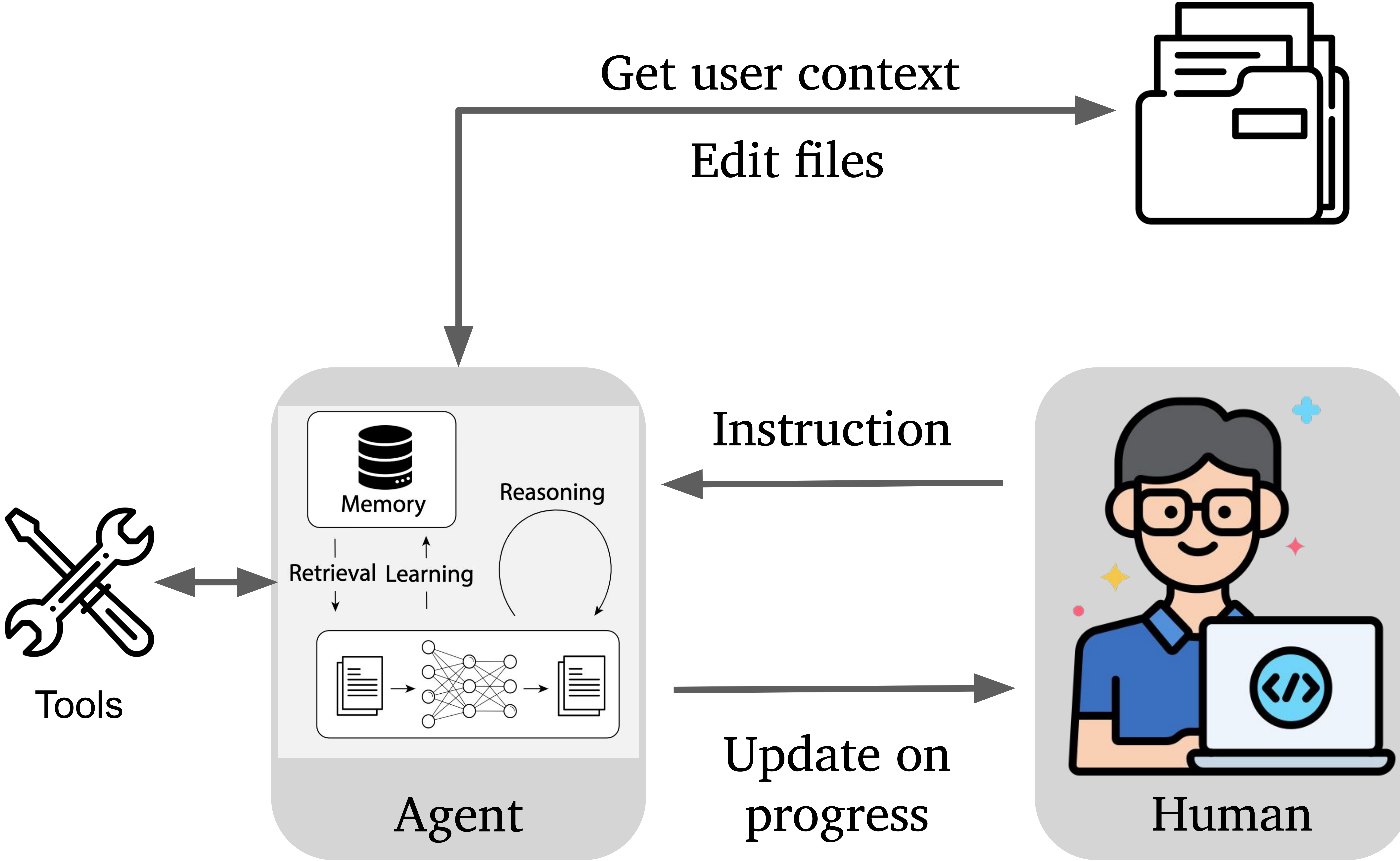


| | Comparison Statement | Distribution |
|---|---|---|
| C1 | I experienced less cognitive load using OpenHands than Github Copilot | 75% 10% |
| C2 | I was able to accomplish new tasks with OpenHands than Github Copilot | 70% 5% |

■ Strongly Agree ■ Agree ■ Neutral ■ Disagree ■ Strongly Disagree

There is room for improvement!

# What should we change about the agent?



Get user context

Edit files

Instruction

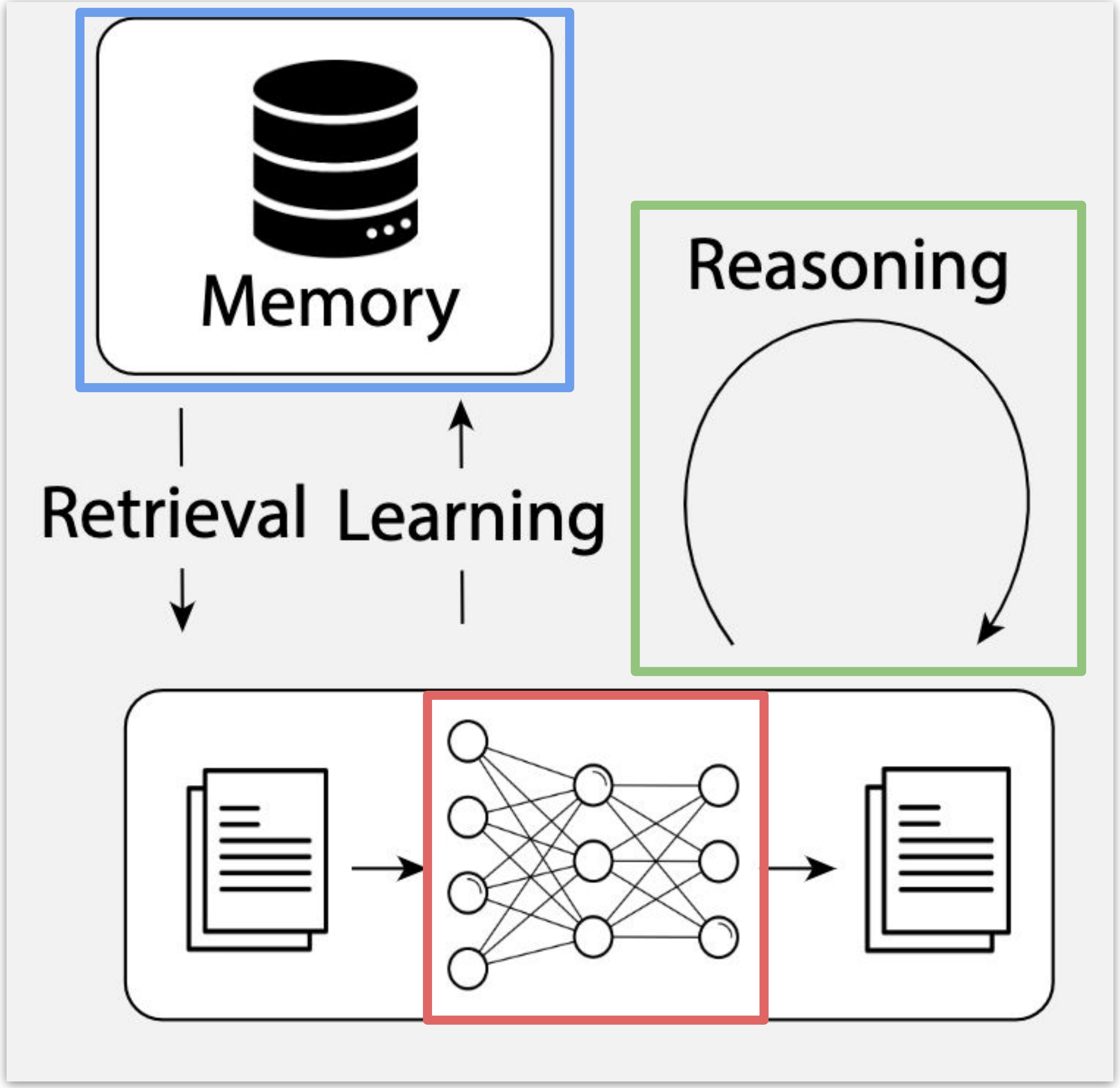
Update on progress

Tools

Agent

Human

# Case studies in agent design



## (1) LLM backbone

- `claude-3.7-sonnet`
- `claude-4-sonnet`
- `gpt-5 (high reasoning)`

## (2) Reasoning strategy

- break down complex tasks through planning
- no planning

## (3) Memory management

- truncate and summarize context after 120 steps
- truncate and summarize context after 80 steps

# Measuring quality of agent work

## Copilot Arena lends itself to natural feedback signal

If you accept suggestion:
suggestion good

If you continue typing:
suggestion bad

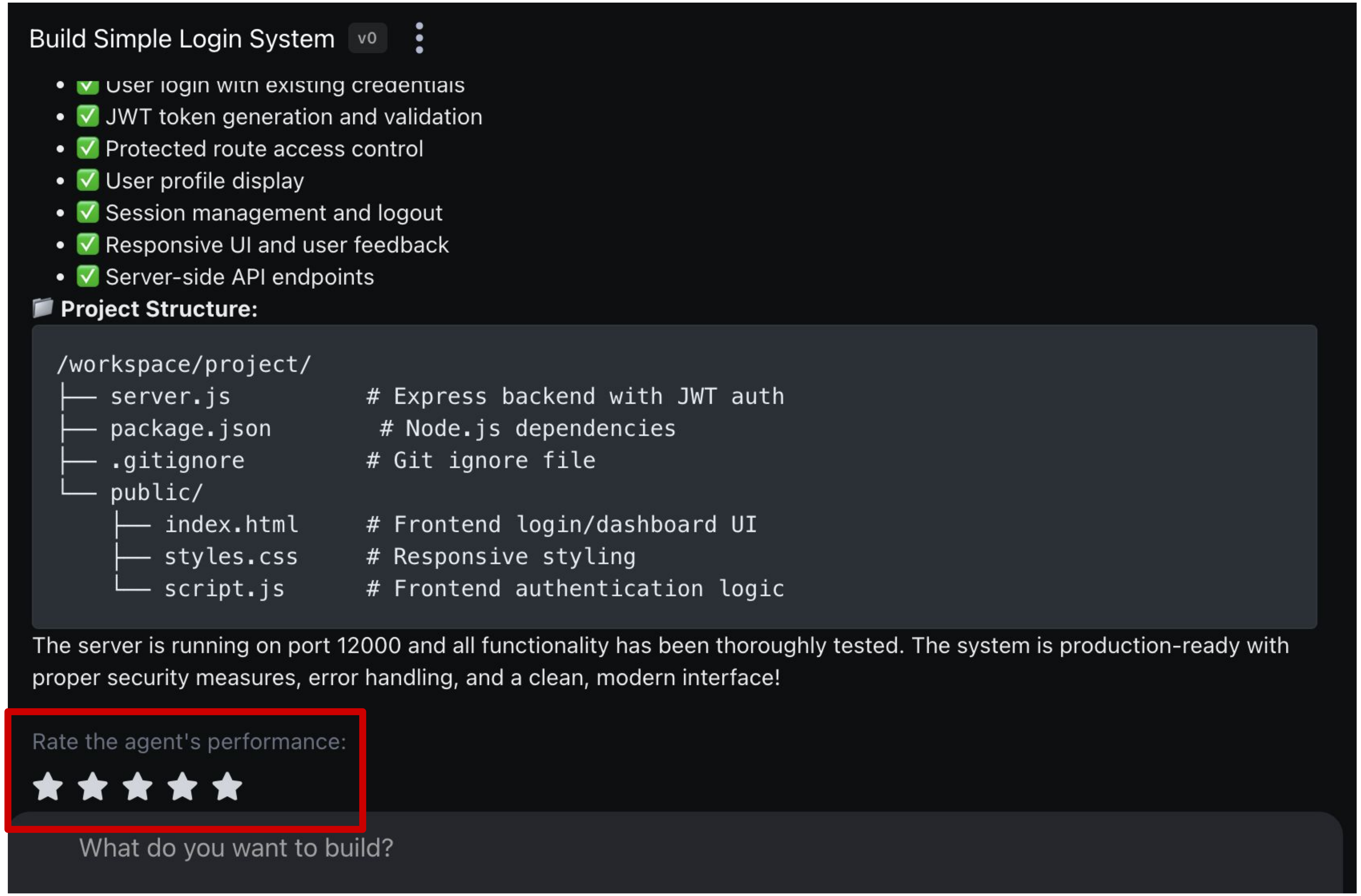
## There is no existing measure in agentic workflows

If you follow-up:
agent work is good??

If you stop a conversation:
agent work is bad??

# **P**rediction-powered **U**ser **L**abel **S**ynthesis & **E**valuation

# Step 1: Collect ratings



- ✅ User login with existing credentials
- ✅ JWT token generation and validation
- ✅ Protected route access control
- ✅ User profile display
- ✅ Session management and logout
- ✅ Responsive UI and user feedback
- ✅ Server-side API endpoints

📁 **Project Structure:**

```
/workspace/project/
├── server.js          # Express backend with JWT auth
├── package.json        # Node.js dependencies
├── .gitignore         # Git ignore file
└── public/
    ├── index.html     # Frontend login/dashboard UI
    ├── styles.css     # Responsive styling
    └── script.js      # Frontend authentication logic
```

The server is running on port 12000 and all functionality has been thoroughly tested. The system is production-ready with proper security measures, error handling, and a clean, modern interface!

Rate the agent's performance:

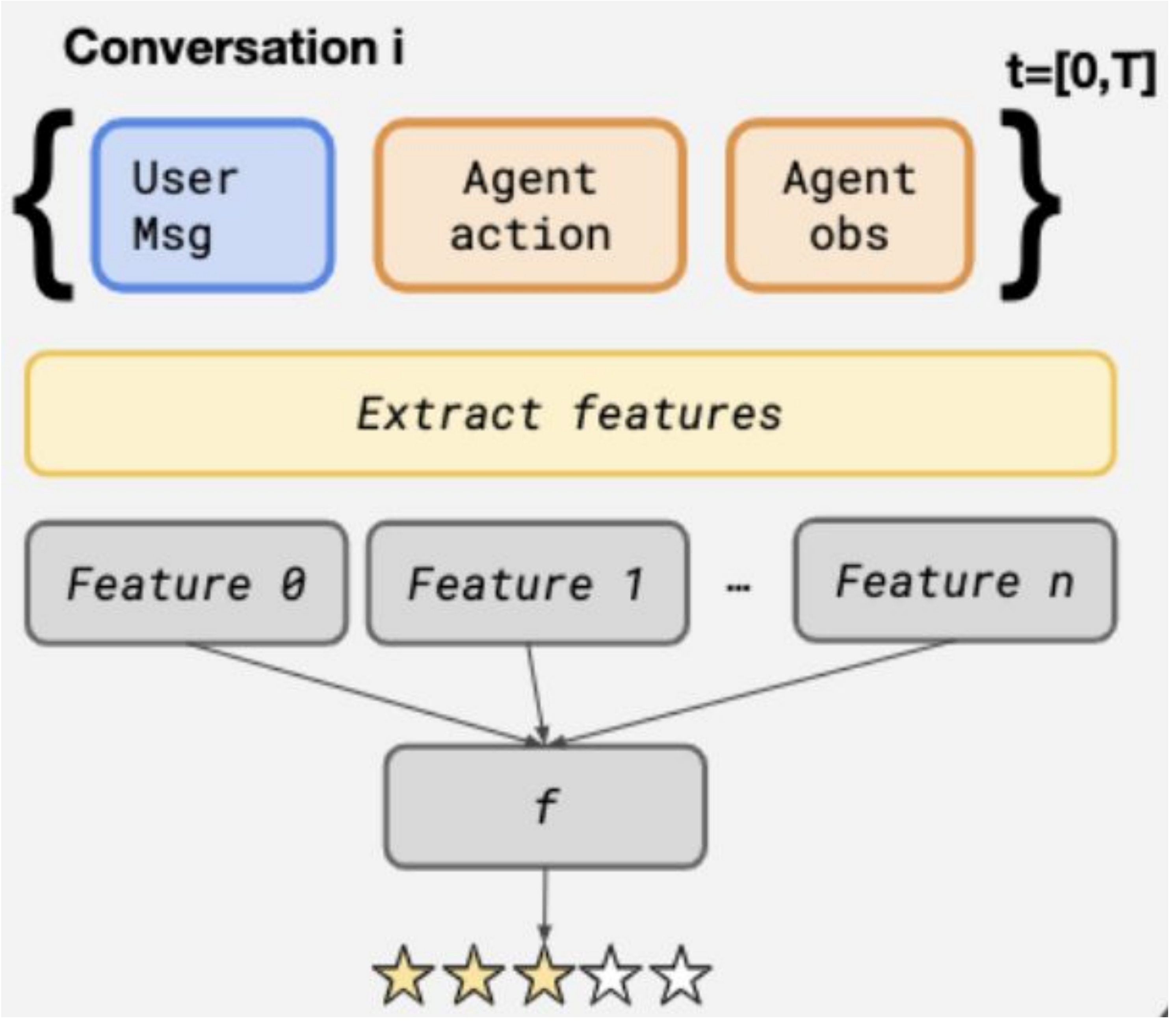
★ ★ ★ ★ ★

Users are prompted to provide feedback after each work segment

A work segment = actions between user command and the agent returning to “stopped” state

We collect a dataset of $N = 1747$ labeled user trajectories where the average rating is 4.07

# Step 2: Train rating predictor



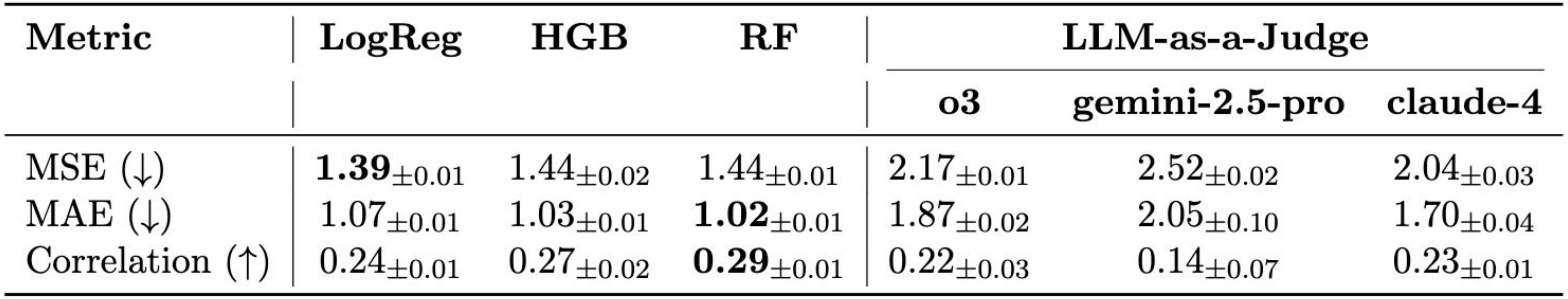
**Features based on the user:**
- Sentiment of user messages
- Number of user messages

**Features based on the agent:**
- Type of task

**Features that show task progression:**
- Git actions

**User sentiment and git push were the most predictive features!**

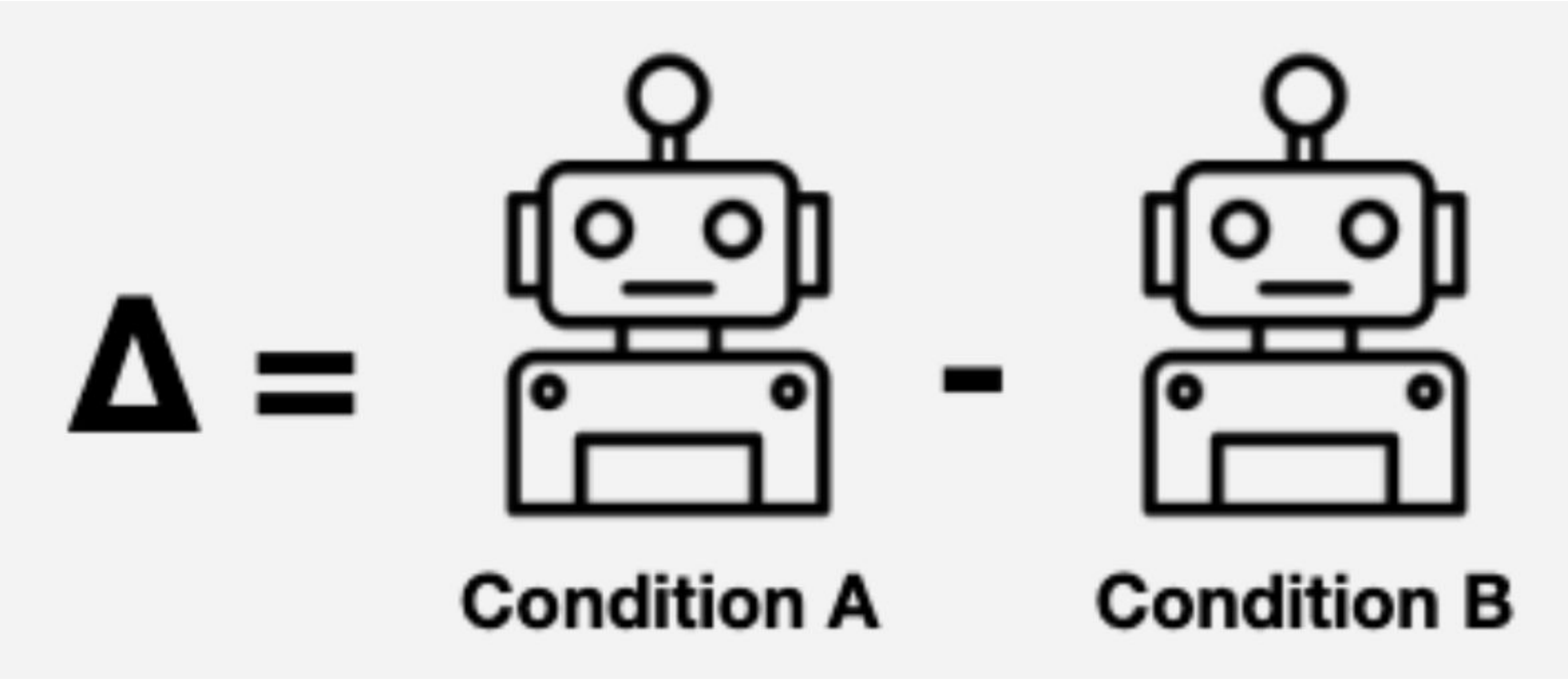
# Step 2: Train rating predictor



| Metric | LogReg | HGB | RF | LLM-as-a-Judge | | |
|---|---|---|---|---|---|---|
| | | | | o3 | gemini-2.5-pro | claude-4 |
| MSE (↓) | **1.39**$_{\pm 0.01}$ | 1.44$_{\pm 0.02}$ | 1.44$_{\pm 0.01}$ | 2.17$_{\pm 0.01}$ | 2.52$_{\pm 0.02}$ | 2.04$_{\pm 0.03}$ |
| MAE (↓) | 1.07$_{\pm 0.01}$ | 1.03$_{\pm 0.01}$ | **1.02**$_{\pm 0.01}$ | 1.87$_{\pm 0.02}$ | 2.05$_{\pm 0.10}$ | 1.70$_{\pm 0.04}$ |
| Correlation (↑) | 0.24$_{\pm 0.01}$ | 0.27$_{\pm 0.02}$ | **0.29**$_{\pm 0.01}$ | 0.22$_{\pm 0.03}$ | 0.14$_{\pm 0.07}$ | 0.23$_{\pm 0.01}$ |

Supervised learning approaches using our features outperform naive LLM-as-a-Judge baseline

# Step 3: Compute effect size



## Naive approach

$$\widehat{\Delta}_{\text{naive}} = \frac{1}{n_{c_1}}\sum_{i\in c_1} Y_i \; - \; \frac{1}{n_{c_2}}\sum_{i\in c_2} Y_i$$

## Augment with Infilled Labels

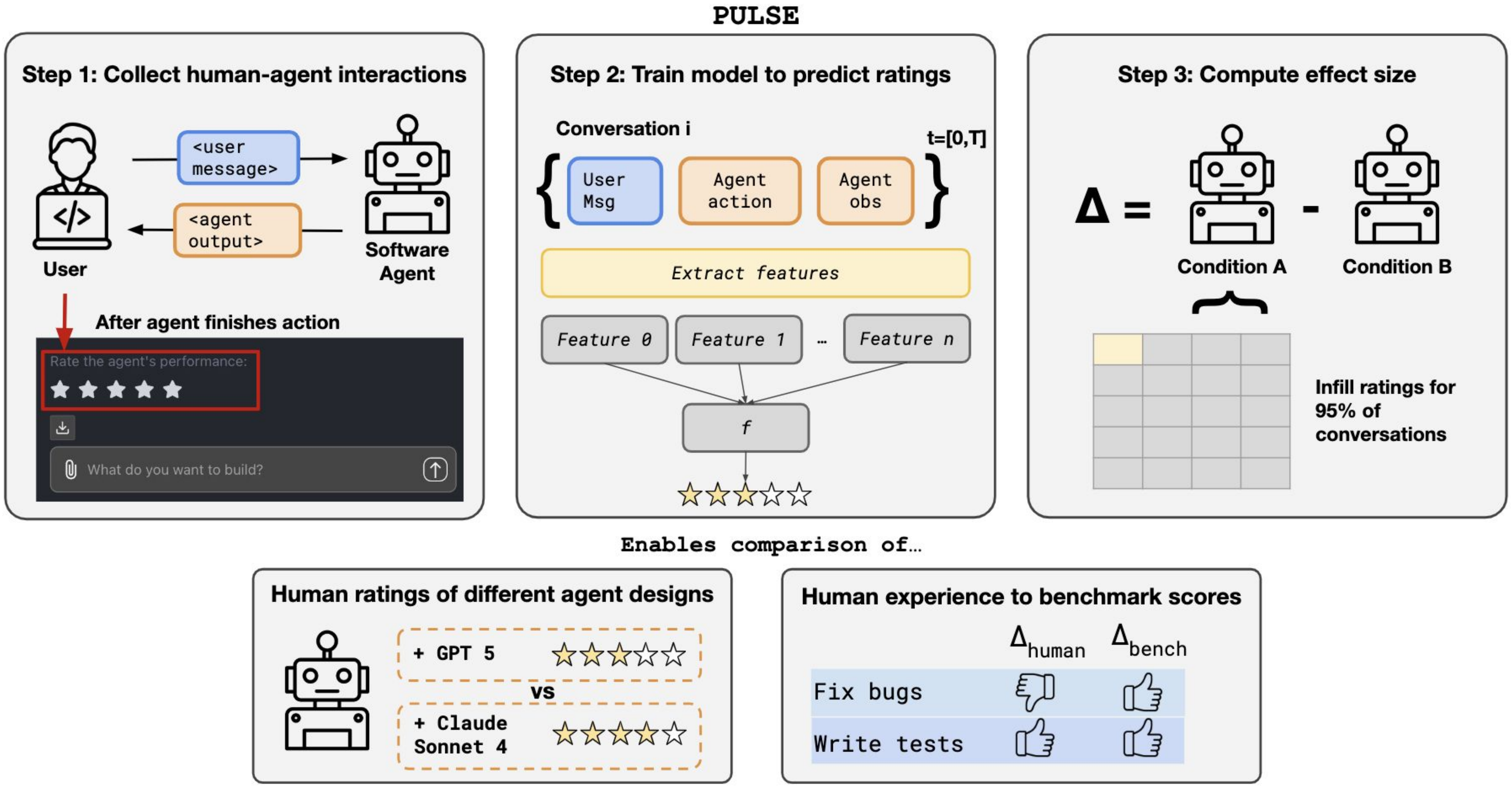
$$\widehat{\mu}_c(\lambda_c) = \underbrace{\frac{1}{n_c}\sum_i Y_i}_{\text{sample mean of labels}} + \lambda_c\left(\frac{1}{N_c}\sum_j \underbrace{f(\tilde{X}_j)}_{\text{unlabeled traj.}} - \frac{1}{n_c}\sum_i \underbrace{f(X_i)}_{\text{labeled traj.}}\right)$$

$$\widehat{\Delta}_{\text{augment}} = \widehat{\mu}_{c_1}(\lambda_{c_1}) - \widehat{\mu}_{c_2}(\lambda_{c_2}).$$

# Overview of Users

Over the course of multiple months, we ran our 3 case studies on over 15k users of the OpenHands SaaS platform.

# Results



**PULSE**

**Step 1: Collect human-agent interactions**

<user message>

<agent output>

User

Software Agent

After agent finishes action

Rate the agent's performance:

★★★★★

What do you want to build?

**Step 2: Train model to predict ratings**

Conversation i

t=[0,T]

{ User Msg | Agent action | Agent obs }

*Extract features*

*Feature 0* | *Feature 1* | … | *Feature n*

$f$

★★★☆☆

**Step 3: Compute effect size**

Δ = Condition A - Condition B

Infill ratings for 95% of conversations

**Enables comparison of…**

**Human ratings of different agent designs**

+ GPT 5 ★★★☆☆

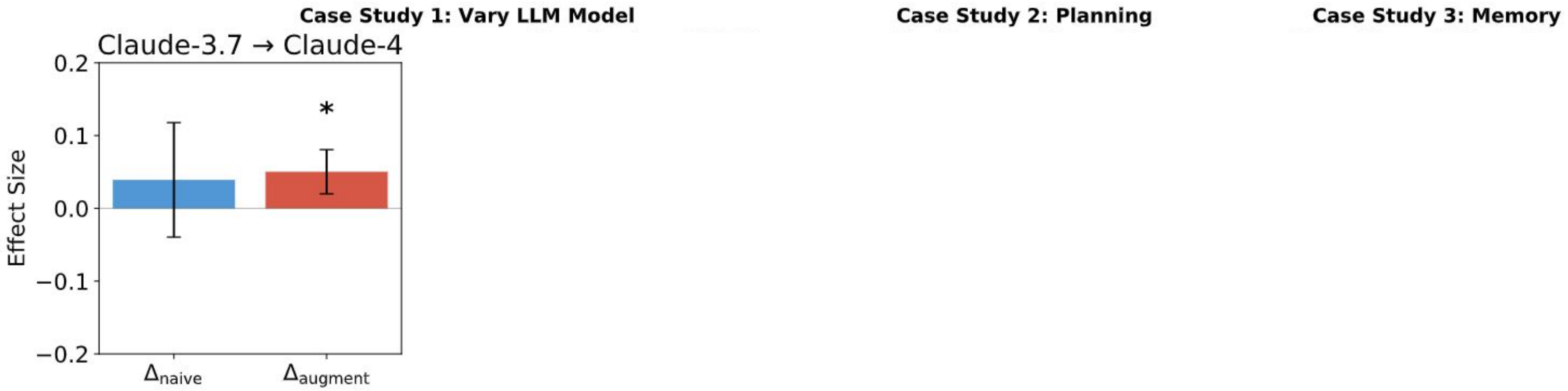
vs

+ Claude Sonnet 4 ★★★★☆

**Human experience to benchmark scores**

| | $\Delta_{human}$ | $\Delta_{bench}$ |
|---|---|---|
| Fix bugs | 👎 | 👍 |
| Write tests | 👍 | 👍 |

# Results



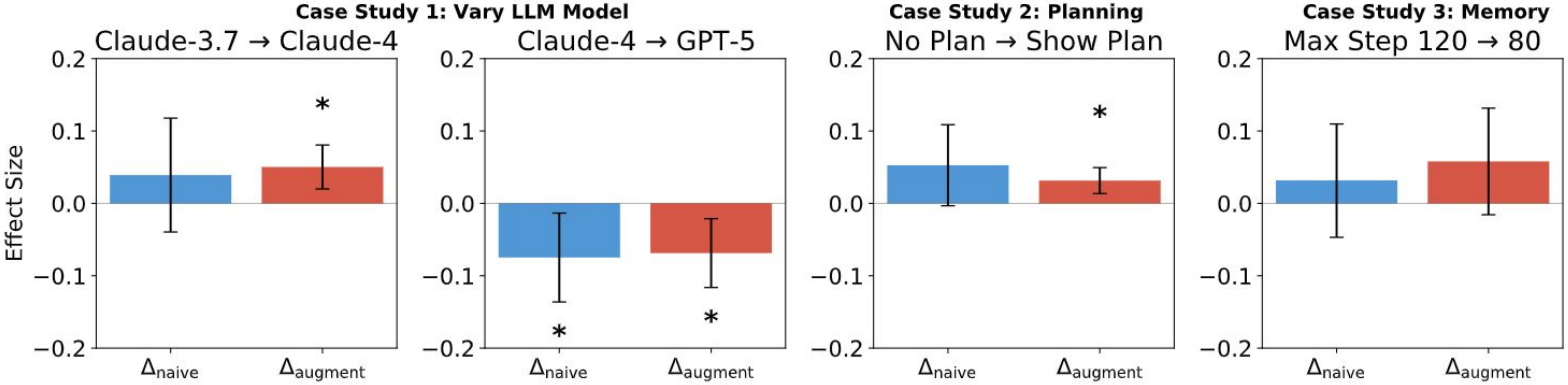
- Changes to the **LLM model** has the largest differences on user satisfaction

+5.9% difference between `claude-3.7-sonnet` and `claude-4-sonnet`
-7.8% difference between `claude-4-sonnet` and `gpt-5`

vs.

+3.1% difference between `no plan` and `planning`

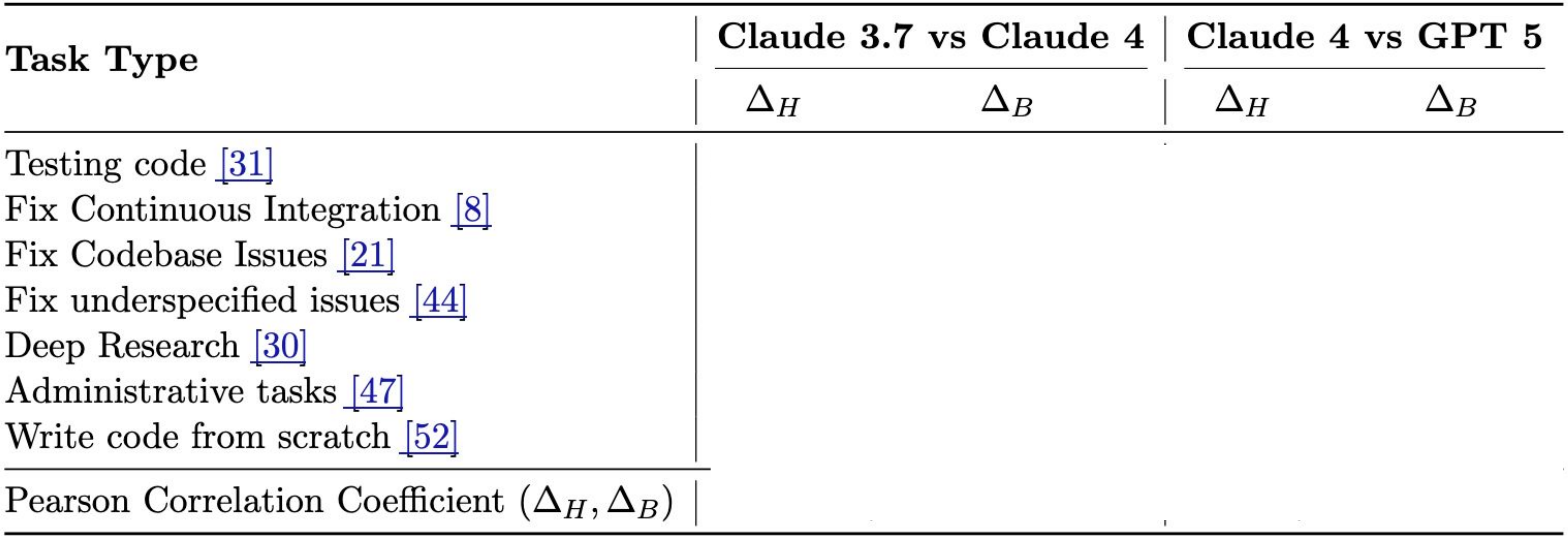
# Results



- No statistically significant in memory case study actually shows how we can reduce cost while preserving user satisfaction

- Our results also show how PULSE can lead to more conclusive results (reducing confidence intervals by up to 40%)

# Comparison to benchmarks



| **Task Type** | **Claude 3.7 vs Claude 4** | | **Claude 4 vs GPT 5** | |
|---|---|---|---|---|
| | $\Delta_H$ | $\Delta_B$ | $\Delta_H$ | $\Delta_B$ |
| Testing code [31] | | | | |
| Fix Continuous Integration [8] | | | | |
| Fix Codebase Issues [21] | | | | |
| Fix underspecified issues [44] | | | | |
| Deep Research [30] | | | | |
| Administrative tasks [47] | | | | |
| Write code from scratch [52] | | | | |
| Pearson Correlation Coefficient $(\Delta_H, \Delta_B)$ | | | | |

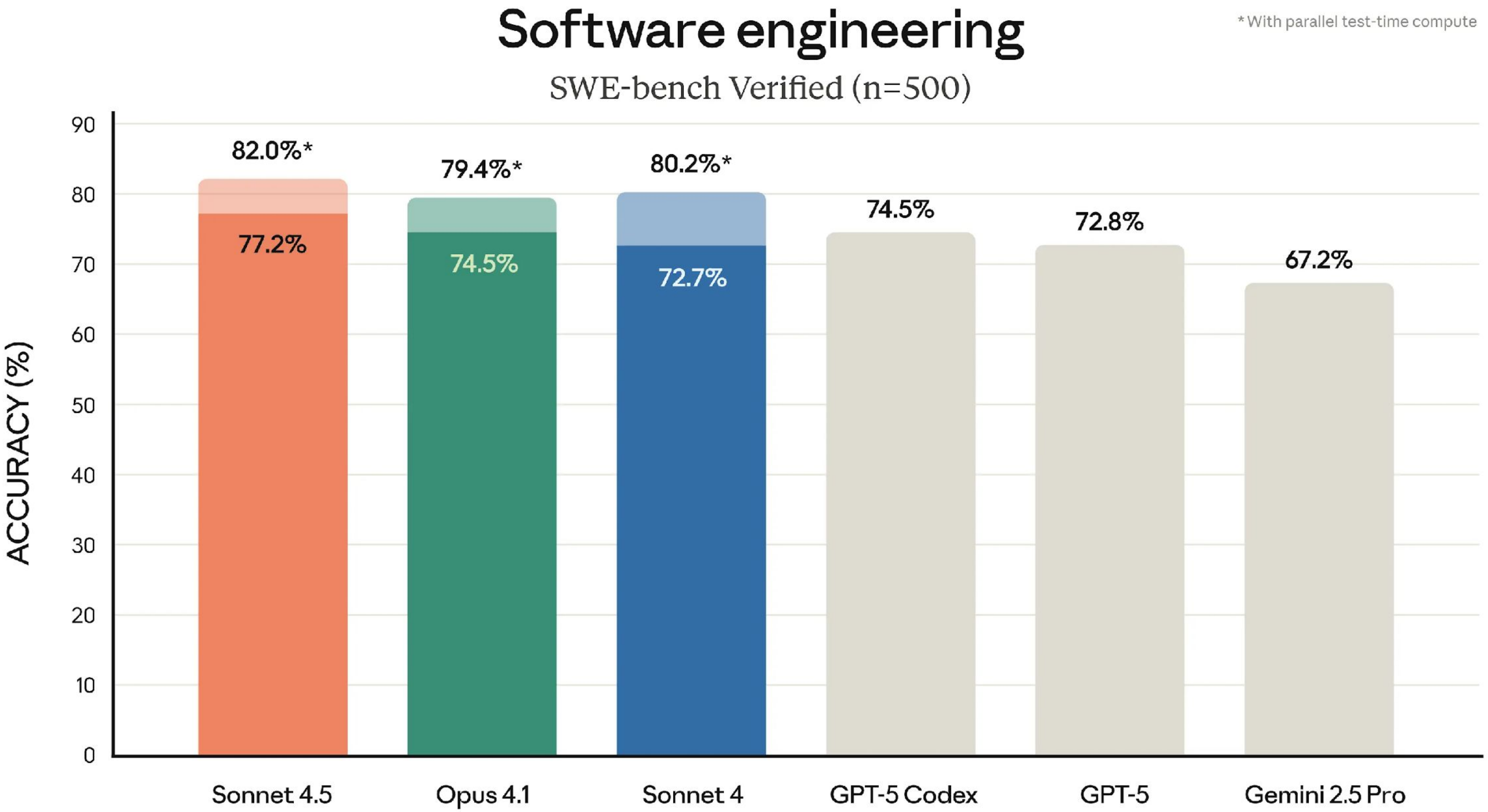
Static benchmarks don’t tell the full story!

# TLDR

- We are seeing a shift towards more autonomous workflows in AI coding assistants
- Evaluations in these multi-turn settings pose unique challenges compared to the copilot setting
- However, benchmarks do not always correlate with user satisfaction, requiring the use of efficient human-in-the-loop approaches

# Future Directions

# Current benchmarks



# Alternatively, focus on collaboration

Desiderata 1:
Agent behaviors should be transparent to users.

Desiderata 2:
Agents should have balanced proactivity.

Desiderata 3:
Agent should effectively leverage human effort.

# Desiderata 1:
# Agent behaviors should be transparent to users.

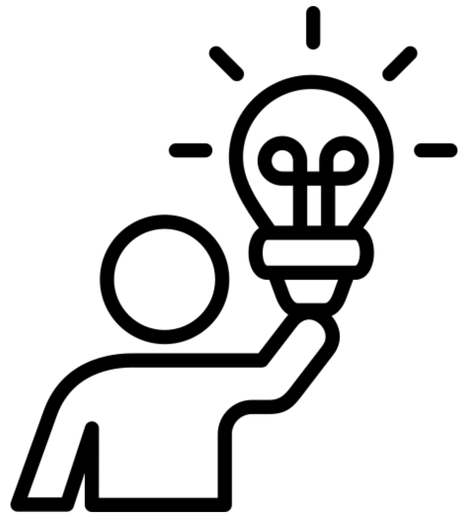
## Current Usage

- Since agents may be editing multiple files and making many changes, it can become difficult for users to understand why the agent made certain changes.
- In the post-study feedback, we found that participants wanted ways to understand quickly what the agent did and why changes were necessary.
- We also see this in the user messages, where one participant asked *"Did you delete most of the functions in [filename]? If so, explain why did you do this."*

## Future Usage

- Prior literature has studied how users consume model explanations has largely focused on ML models and more recently LLMs.
- However, there is a need to propose explanations of agent actions. Recent work introduced a way for agent developers to view counterfactual roll-outs, but this is not necessarily user-friendly for end users (e.g., developers).
- Future work should consider how to improve the transparency of agent behaviors

# Desiderata 2:
# Agents should have balanced proactivity.



### Current Usage

- Many participants observed, or even complained, about how OpenHands would take more actions than necessary.
- One participant wrote in a message to the agent, *“could the code have been simplified, I did not expect 10 files to be created with more than 1000 lines each.”*

### Future Usage

- Agents should be better calibrated in terms of their confidence about whether it has completed the user’s request.
- Recent work on UI agents has explored proactively pausing agents at task boundaries, identifying such boundaries in the software engineering settings may be a fruitful direction to improve user perception of agent actions.

# Desiderata 3:
# Agent should effectively leverage human effort.

## Current Usage

- Human effort can be measured in many ways, including the amount of time spent interacting with the agent.
- On this front, many participants noted that the *“the generation time is slower”* for OpenHands than GitHub Copilot and sometimes *“im just kind of sitting there”*.

## Future Usage

- User experience can be improved by explicitly optimizing for latency when engaging in back-and-forth with the user and providing more direct ways for users to steer agent behaviors.
- Additionally, developers will increasingly need to multi-task to be most productive in agentic workflows, though prior work has characterized the cognitive cost of doing so.

# Acknowledgements and paper links

Papers mentioned in this talk

- The RealHumanEval: Evaluating Large Language Models' Abilities to Support Programmers. TMLR, 2025.
- Copilot Arena: A Platform for Code LLM Evaluation in the Wild. ICML, 2025
- Code with Me or for Me? How Increasing AI Automation Transforms Developer Workflows. Under Review.
- How can we assess human-agent interactions? Case studies in software agent design. Under Review.

The projects presented in this talk are in collaboration with

Aditya Bharat Soni, Aditya Mittal, Ameet Talwalkar, Anastasios Angelopoulos, Calvin Smith, Chris Donahue, David Sontag, Dennis Wei, Graham Neubig, Hoang H Tran, Hussein Mozannar, Ion Stoica, Juan Michelini, Manish Nagireddy, Mohammed Alsobay, Naman Jain, Prasanna Sattigeri, Robert Brennan, Rohit Malhotra, Sebastian Zhao, Subhro Das, Tianjun Zhang, Wayne Chi, Wei-Lin Chiang, Xingyao Wang, Xuhui Zhou

From

CMU, MIT, UC Berkeley, IBM, OpenHands, LM Arena

# Thank you! Questions?