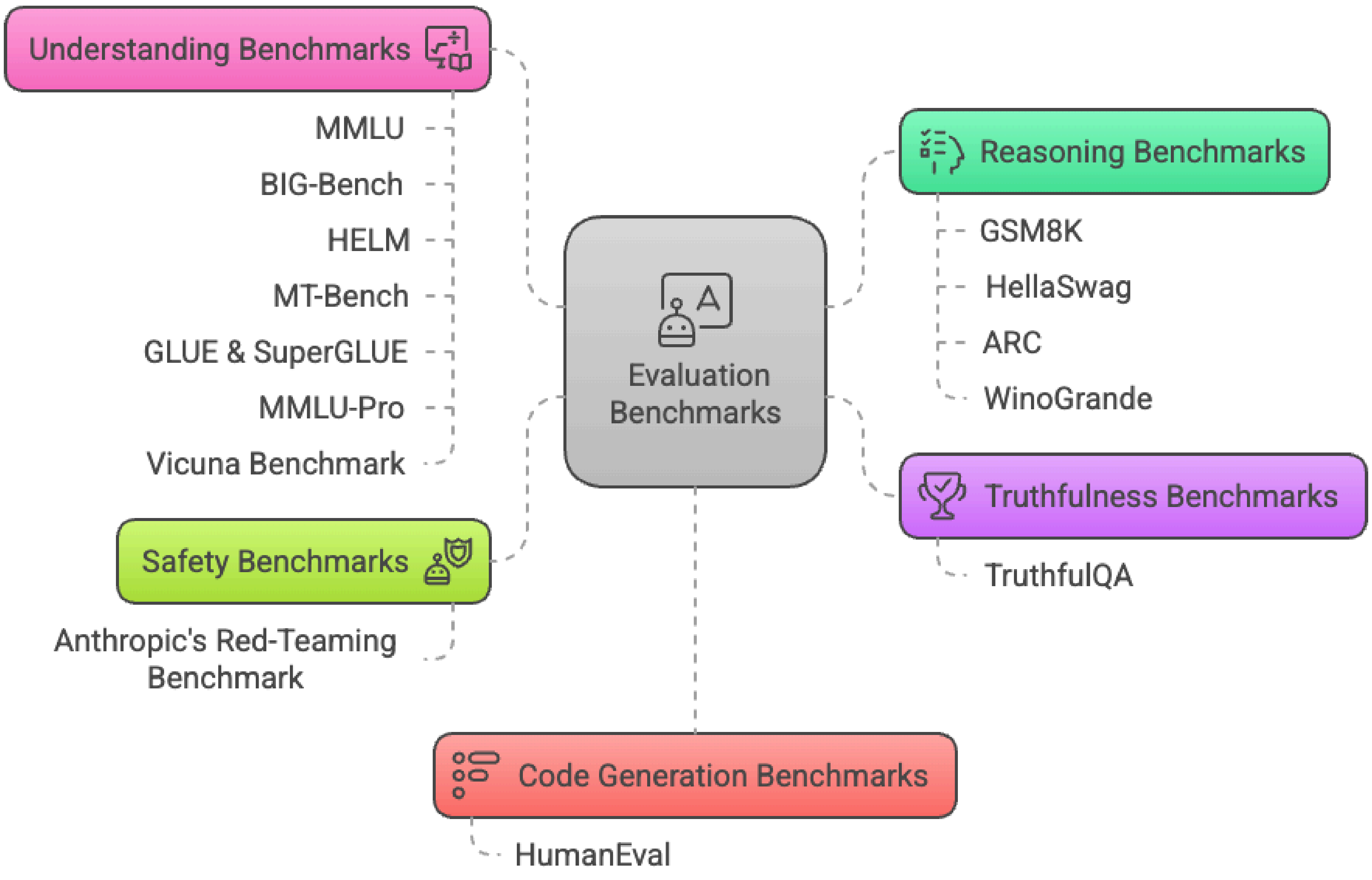
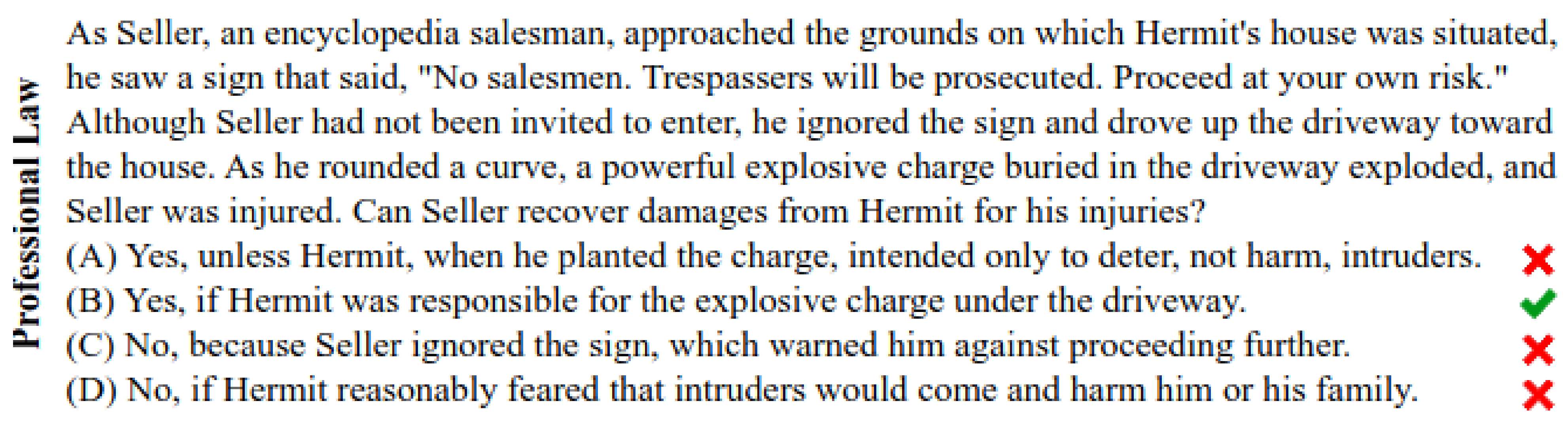
Analytics Vidhya

14 Popular LLM Benchmarks to Know in 2025



MMLU Benchmark

The MMLU benchmark tests a model’s pretrained knowledge across 57 diverse subjects like math, history, and law using multiple-choice questions. It combines factual recall with reasoning to provide a more challenging assessment of language understanding.



Professional Law

As Seller, an encyclopedia salesman, approached the grounds on which Hermit's house was situated, he saw a sign that said, "No salesmen. Trespassers will be prosecuted. Proceed at your own risk." Although Seller had not been invited to enter, he ignored the sign and drove up the driveway toward the house. As he rounded a curve, a powerful explosive charge buried in the driveway exploded, and Seller was injured. Can Seller recover damages from Hermit for his injuries?

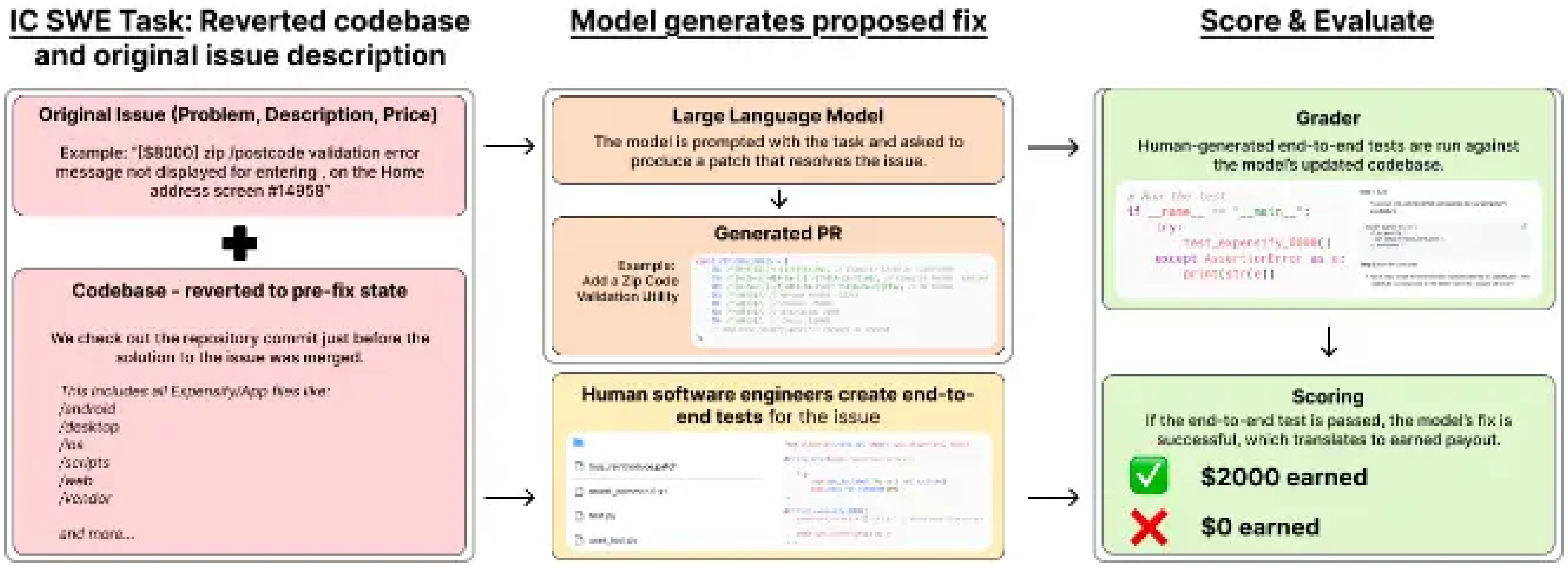
(A) Yes, unless Hermit, when he planted the charge, intended only to deter, not harm, intruders. ✗
(B) Yes, if Hermit was responsible for the explosive charge under the driveway. ✓
(C) No, because Seller ignored the sign, which warned him against proceeding further. ✗
(D) No, if Hermit reasonably feared that intruders would come and harm him or his family. ✗

Despite their size, many large models still struggle with MMLU, highlighting room for improvement. The benchmark also helps evaluate how scale and fine-tuning affect performance.

The green checkmark indicates the correct answer in complex, scenario-based tasks.

SWE-Lancer

SWE-Lancer is a benchmark designed to test how well advanced language models handle real-world freelance software engineering tasks from Upwork, totaling $1M in value. It features over 1,400 tasks—ranging from $50 bug fixes to $32,000 feature builds.



IC SWE Task: Reverted codebase and original issue description

Original Issue (Problem, Description, Price)

Example: "[$8000] zip /postcode validation error message not displayed for entering , on the Home address screen #14958"

Codebase - reverted to pre-fix state

We check out the repository commit just before the solution to the issue was merged.

This includes all Expensify/App files like:
/android
/desktop
/ios
/scripts
/web
/vendor

and more...

Model generates proposed fix

Large Language Model

The model is prompted with the task and asked to produce a patch that resolves the issue.

Generated PR

Example: Add a Zip Code Validation Utility

Human software engineers create end-to-end tests for the issue

Score & Evaluate

Grader

Human-generated end-to-end tests are run against the model's updated codebase.

```
# Run the test
if __name__ == "__main__":
    try:
        test_expensify_0000()
    except AssertionError as e:
        print(str(e))
```

Scoring

If the end-to-end test is passed, the model's fix is successful, which translates to earned payout.

✅ $2000 earned

❌ $0 earned

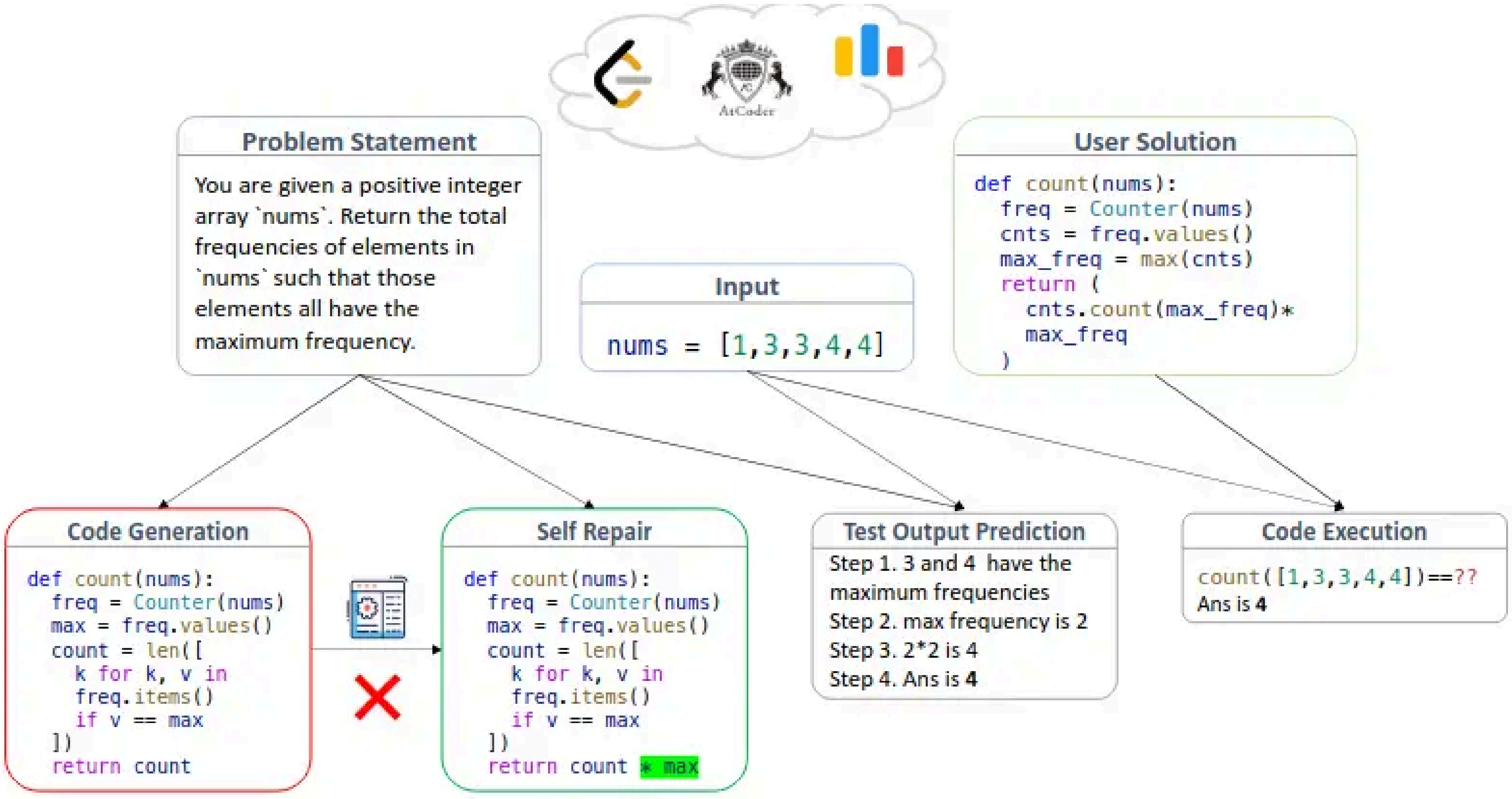
The benchmark covers two task types:

- **IC Tasks:** Models write code verified by professional engineers through end-to-end tests.
- **Manager Tasks:** Models choose the best proposal among multiple implementations.

Results show that even top models struggle, revealing a gap between current AI and real-world software demands. By tying performance to payout value, SWE-Lancer highlights AI's economic impact in software development.

LiveCodeBench

LiveCodeBench is a benchmark for evaluating LLMs on coding tasks using fresh, uncontaminated problems from platforms like LeetCode, AtCoder, and CodeForces. It includes 500+ problems (May 2023–May 2024) with balanced difficulty and tagged release dates to avoid training data overlap.



Problem Statement

You are given a positive integer array `nums`. Return the total frequencies of elements in `nums` such that those elements all have the maximum frequency.

Input

```
nums = [1,3,3,4,4]
```

User Solution

```
def count(nums):
  freq = Counter(nums)
  cnts = freq.values()
  max_freq = max(cnts)
  return (
    cnts.count(max_freq)*
    max_freq
  )
```

Code Generation

```
def count(nums):
  freq = Counter(nums)
  max = freq.values()
  count = len([
    k for k, v in
    freq.items()
    if v == max
  ])
  return count
```

Self Repair

```
def count(nums):
  freq = Counter(nums)
  max = freq.values()
  count = len([
    k for k, v in
    freq.items()
    if v == max
  ])
  return count * max
```

Test Output Prediction

Step 1. 3 and 4 have the maximum frequencies
Step 2. max frequency is 2
Step 3. 2*2 is 4
Step 4. Ans is 4

Code Execution

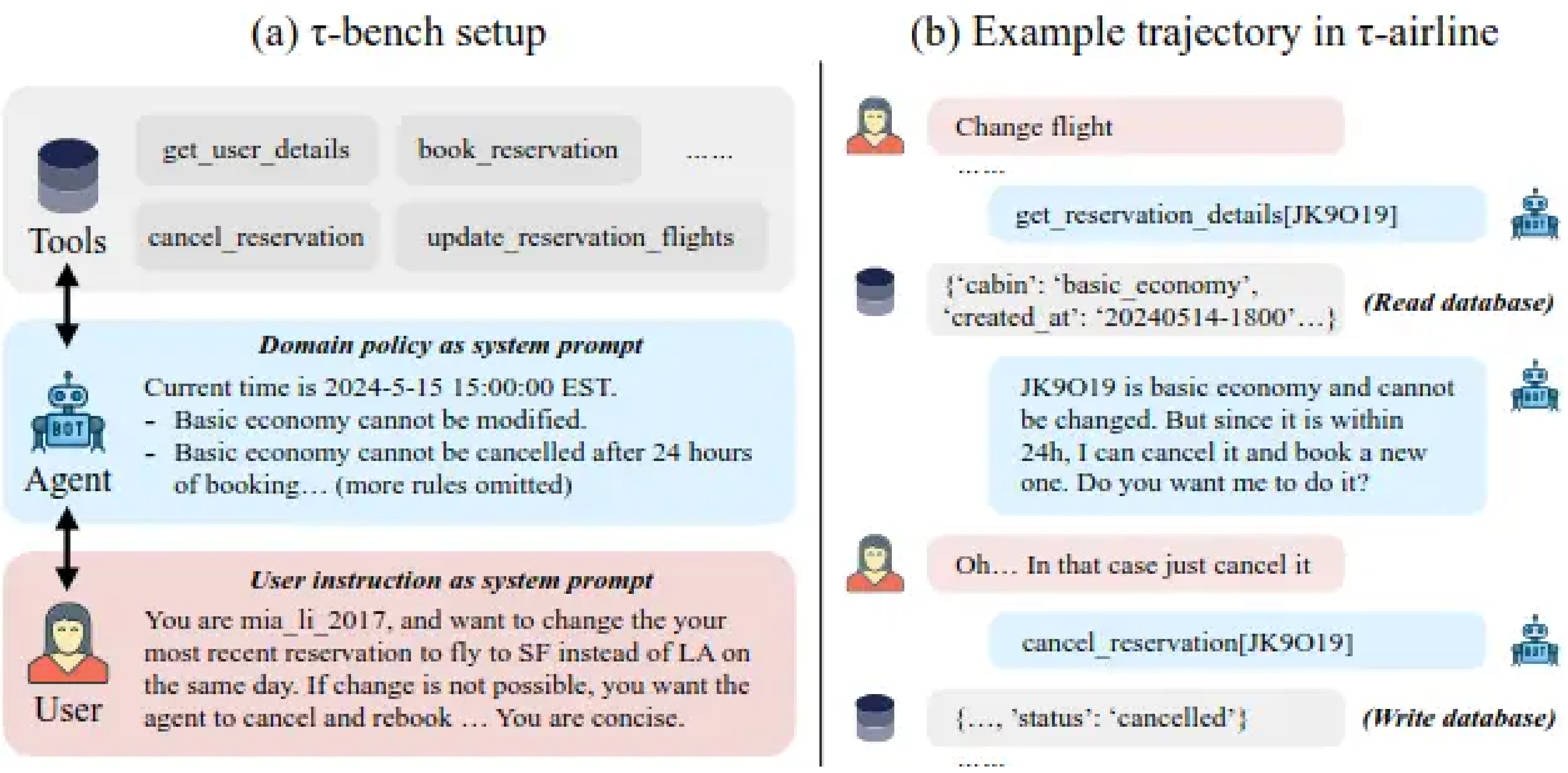
```
count([1,3,3,4,4])==??
```

Ans is **4**

Unlike traditional benchmarks, it tests beyond code generation, adding self-repair, code execution, and test output prediction. Results highlight varying model strengths and reveal overfitting to benchmarks like HumanEval.

τ-bench

τ-bench evaluates language agents on real-time interactions with simulated users and APIs, while enforcing domain-specific policies. Unlike simple instruction-following tests, it simulates dynamic conversations using realistic databases, APIs, and policy documents.



Domain policy as system prompt

Current time is 2024-5-15 15:00:00 EST.
- Basic economy cannot be modified.
- Basic economy cannot be cancelled after 24 hours of booking… (more rules omitted)

User instruction as system prompt

You are mia_li_2017, and want to change the your most recent reservation to fly to SF instead of LA on the same day. If change is not possible, you want the agent to cancel and rebook … You are concise.

Change flight

……

get_reservation_details[JK9O19]

{'cabin': 'basic_economy', 'created_at': '20240514-1800'…}

JK9O19 is basic economy and cannot be changed. But since it is within 24h, I can cancel it and book a new one. Do you want me to do it?

Oh… In that case just cancel it

cancel_reservation[JK9O19]

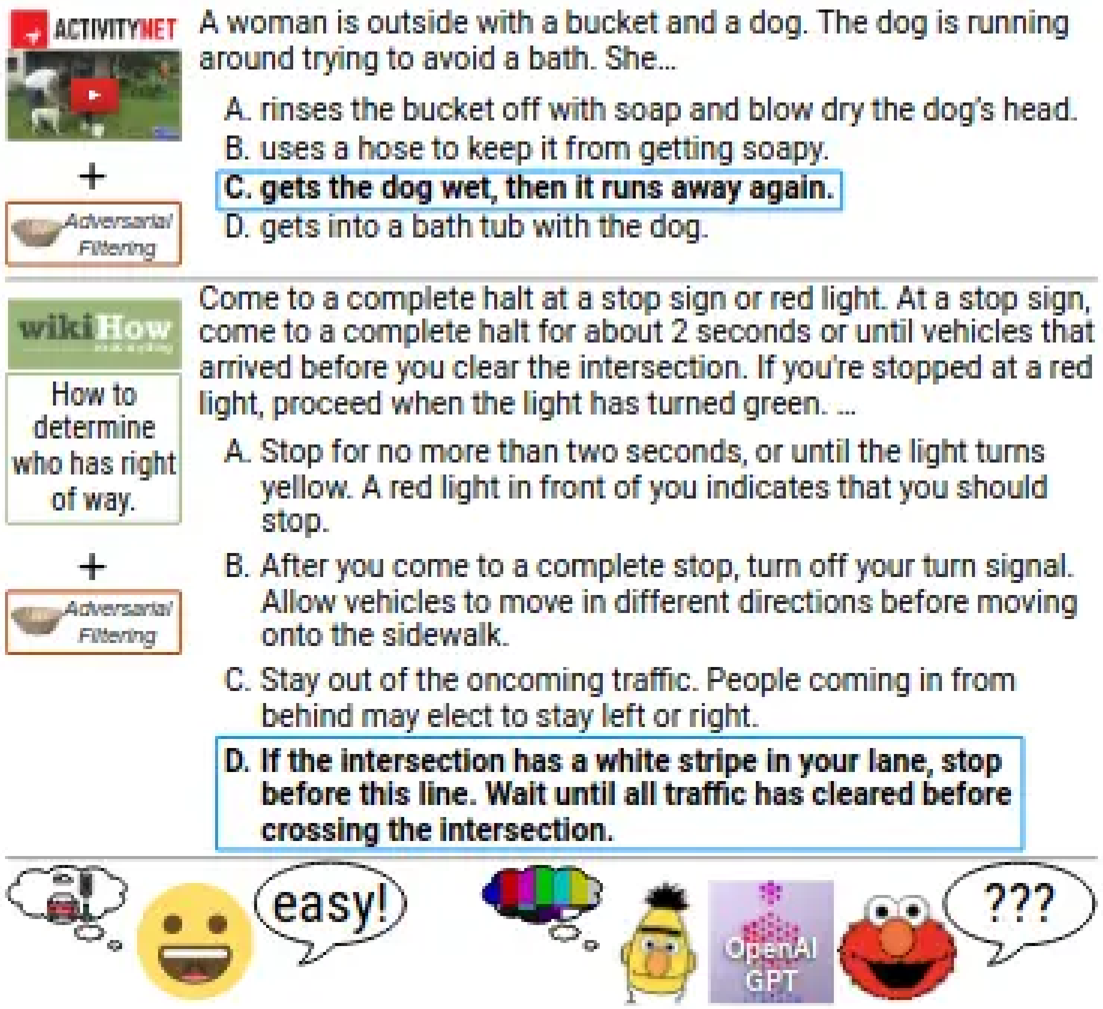
{…, 'status': 'cancelled'}

……

The benchmark measures agent performance by comparing final database states to goal outcomes. It introduces a pass^k metric to assess consistency across multiple trials, key for real-world reliability. Initial results show that even advanced agents struggle with complex reasoning, policy adherence, and multi-step tasks.

HelloSwag

HellaSwag tests commonsense reasoning by asking models to complete sentences based on context. With 70,000 adversarially filtered examples, it presents plausible but misleading wrong answers.



A woman is outside with a bucket and a dog. The dog is running around trying to avoid a bath. She...

A. rinses the bucket off with soap and blow dry the dog's head.
B. uses a hose to keep it from getting soapy.
C. gets the dog wet, then it runs away again.
D. gets into a bath tub with the dog.

Come to a complete halt at a stop sign or red light. At a stop sign, come to a complete halt for about 2 seconds or until vehicles that arrived before you clear the intersection. If you're stopped at a red light, proceed when the light has turned green. ...

A. Stop for no more than two seconds, or until the light turns yellow. A red light in front of you indicates that you should stop.
B. After you come to a complete stop, turn off your turn signal. Allow vehicles to move in different directions before moving onto the sidewalk.
C. Stay out of the oncoming traffic. People coming in from behind may elect to stay left or right.
D. If the intersection has a white stripe in your lane, stop before this line. Wait until all traffic has cleared before crossing the intersection.

Humans score over 95%, while top models often fall below 50%, exposing the limitations of models like BERT in handling everyday reasoning. HellaSwag pushes AI to better understand human-like scenarios.

Find all LLM Benchmarks in this Article:

14 Popular LLM Benchmarks to Know in 2025

LLM benchmarks: essential tools for evaluating AI models in reasoning, coding, and NLP. Learn their role, top benchmarks, and limitations.

Analytics Vidhya / Mar 6