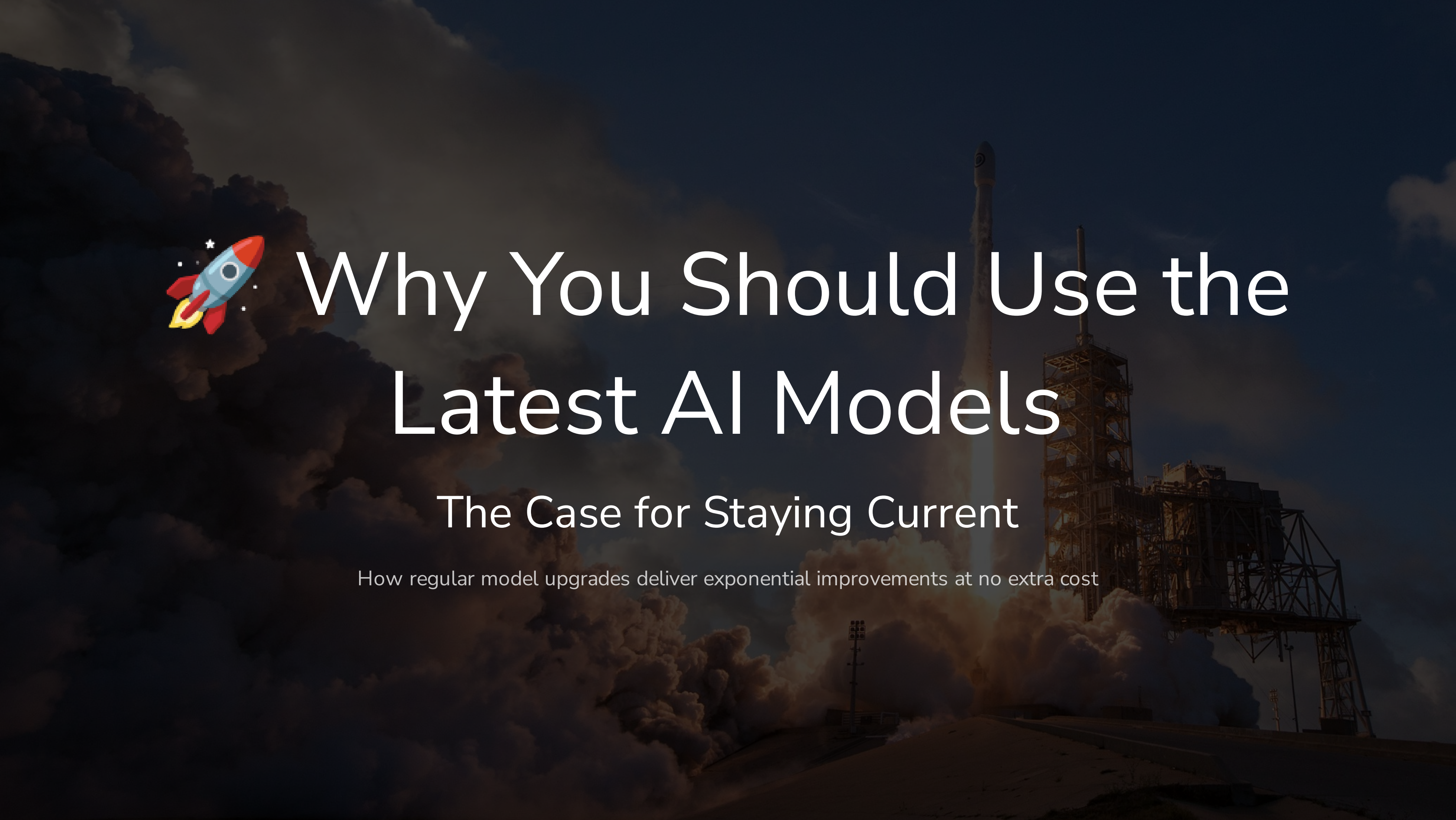


🚀 Why You Should Use the Latest AI Models

The Case for Staying Current

How regular model upgrades deliver exponential improvements at no extra cost

How do we know if a new model is better?

When a new model releases, how do we decide if we should upgrade?

You can’t just vibe check it 🤷

You can’t rely on gut feelings 🎲

You need a objective way to compare

That means 📈 Benchmarks and 📊 Evals

Coding Tasks are the Gold Standard

🎯 Objectively Measurable

- ✅ Works or ❌ doesn’t
- Tests pass or fail
- Zero ambiguity
- No tricks allowed

🤔 Other Tasks Fail

- "Product review" - Sounds good, means nothing
- "Summarize doc" - Hard to measure
- "Marketing copy" - Persuasive ≠ correct

SWE-bench

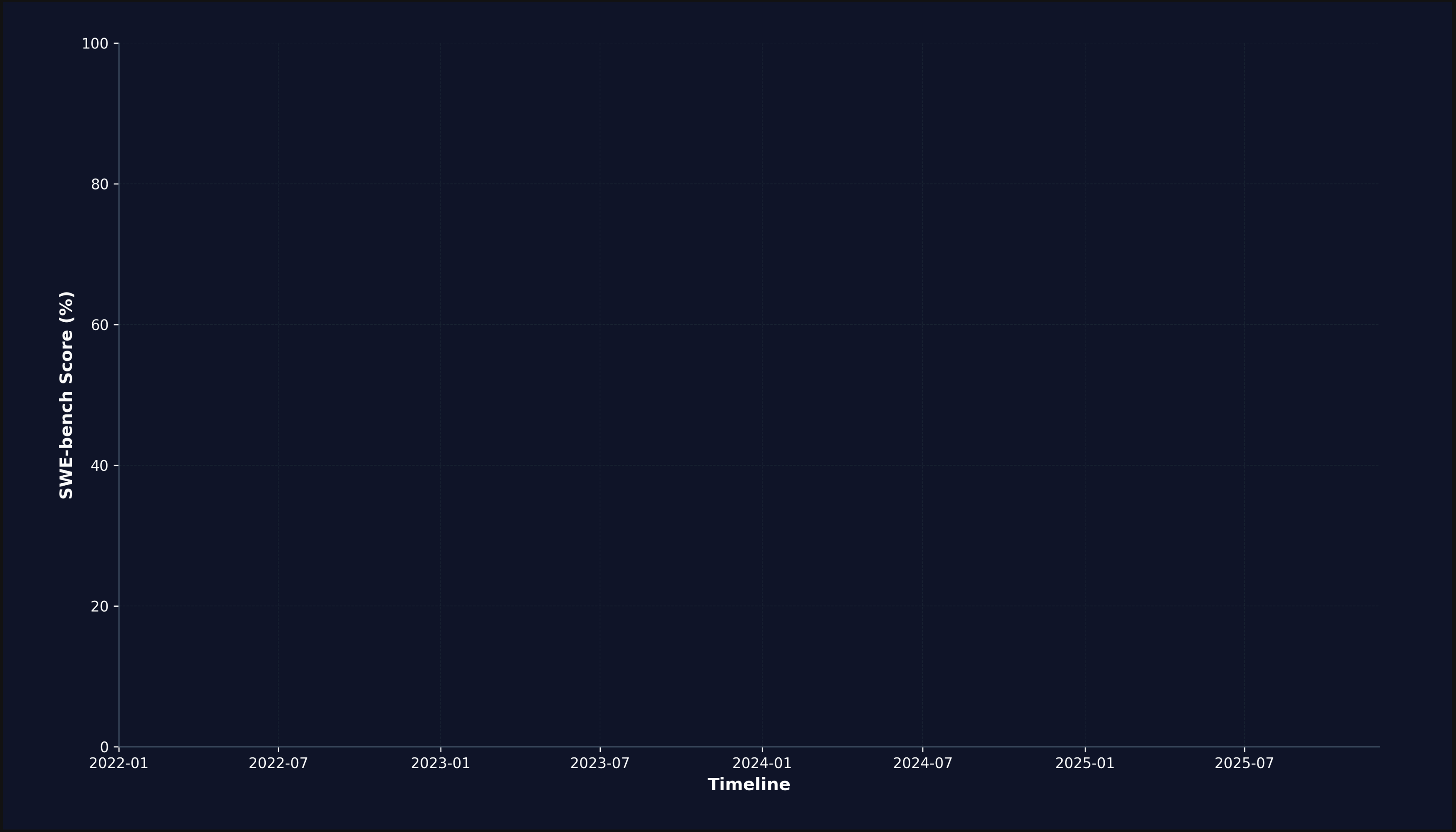
Real-World Coding Problems

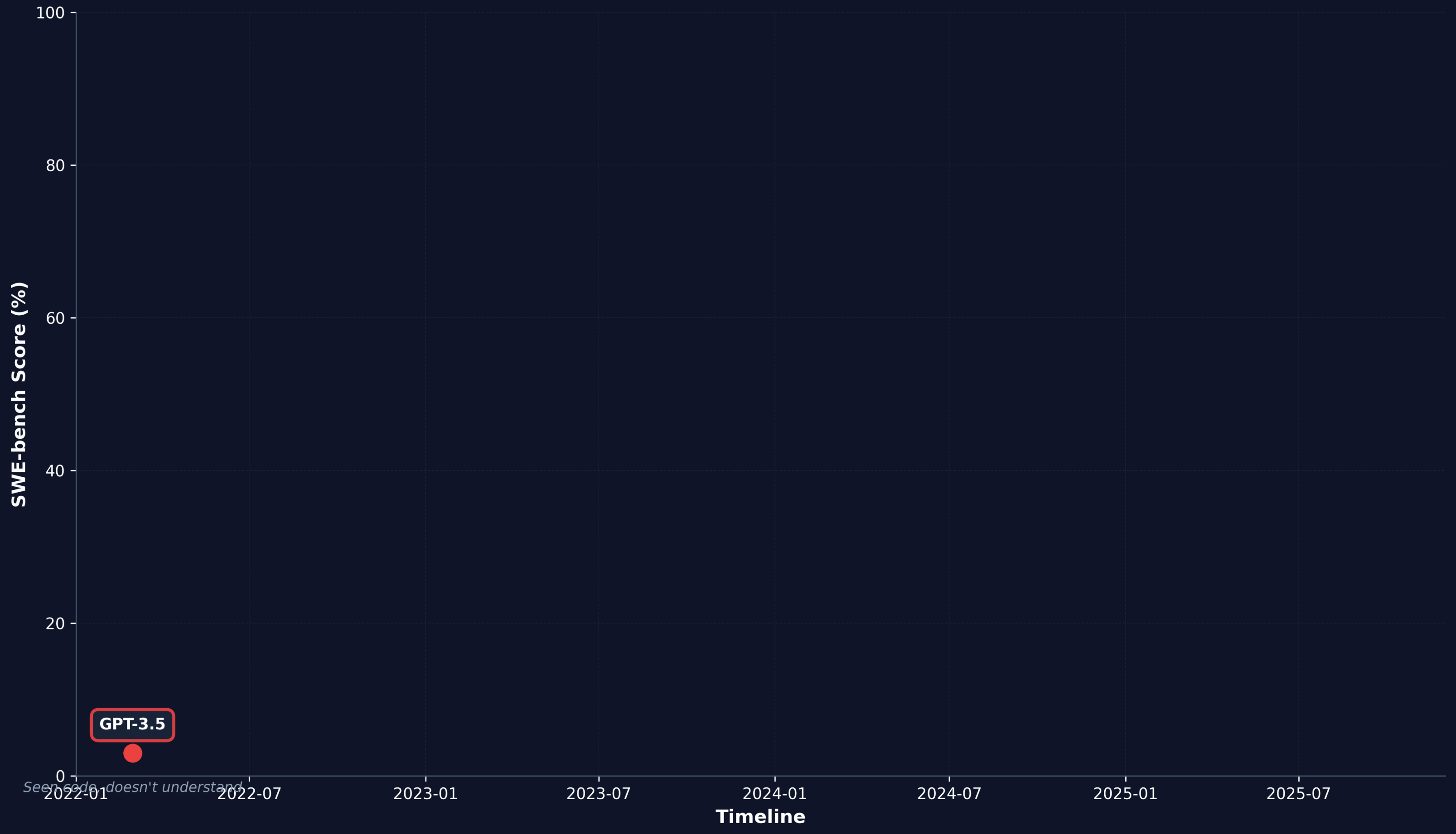
What it measures:

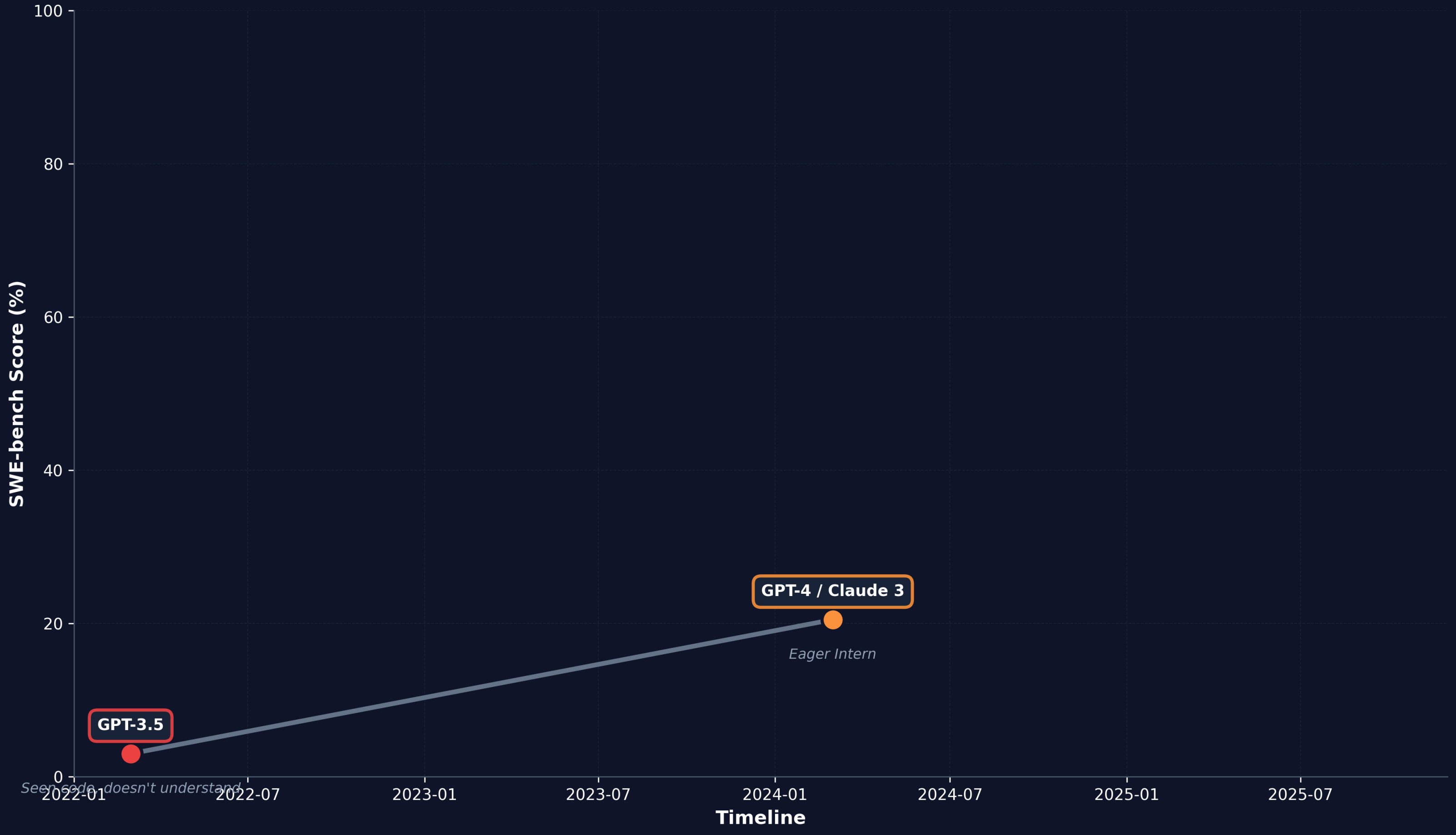
- Popular GitHub repositories
- Real GitHub issues
- Tests ability to understand, plan, and fix actual software bugs
- Industry-standard for measuring coding capability

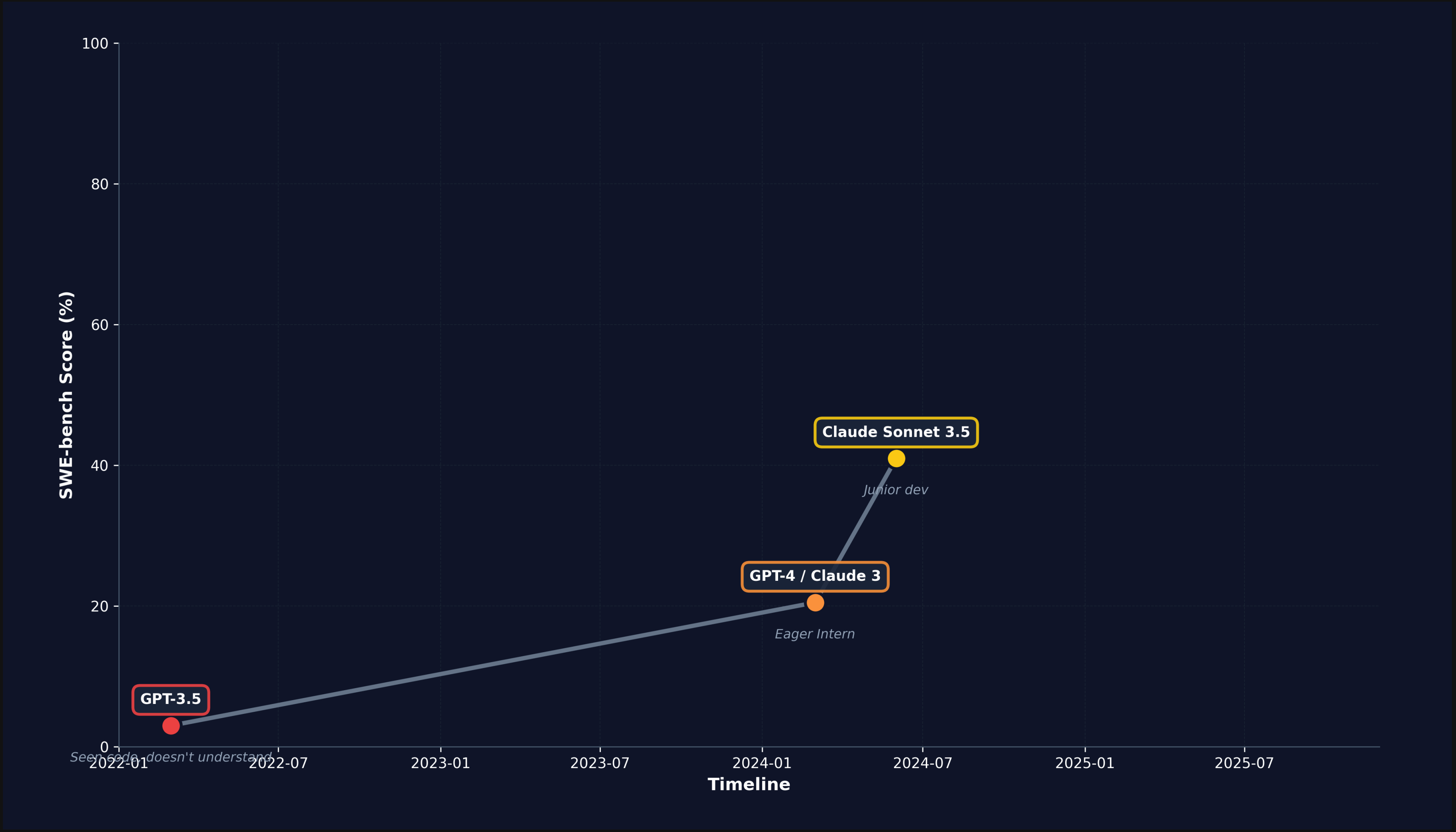
The Dataset

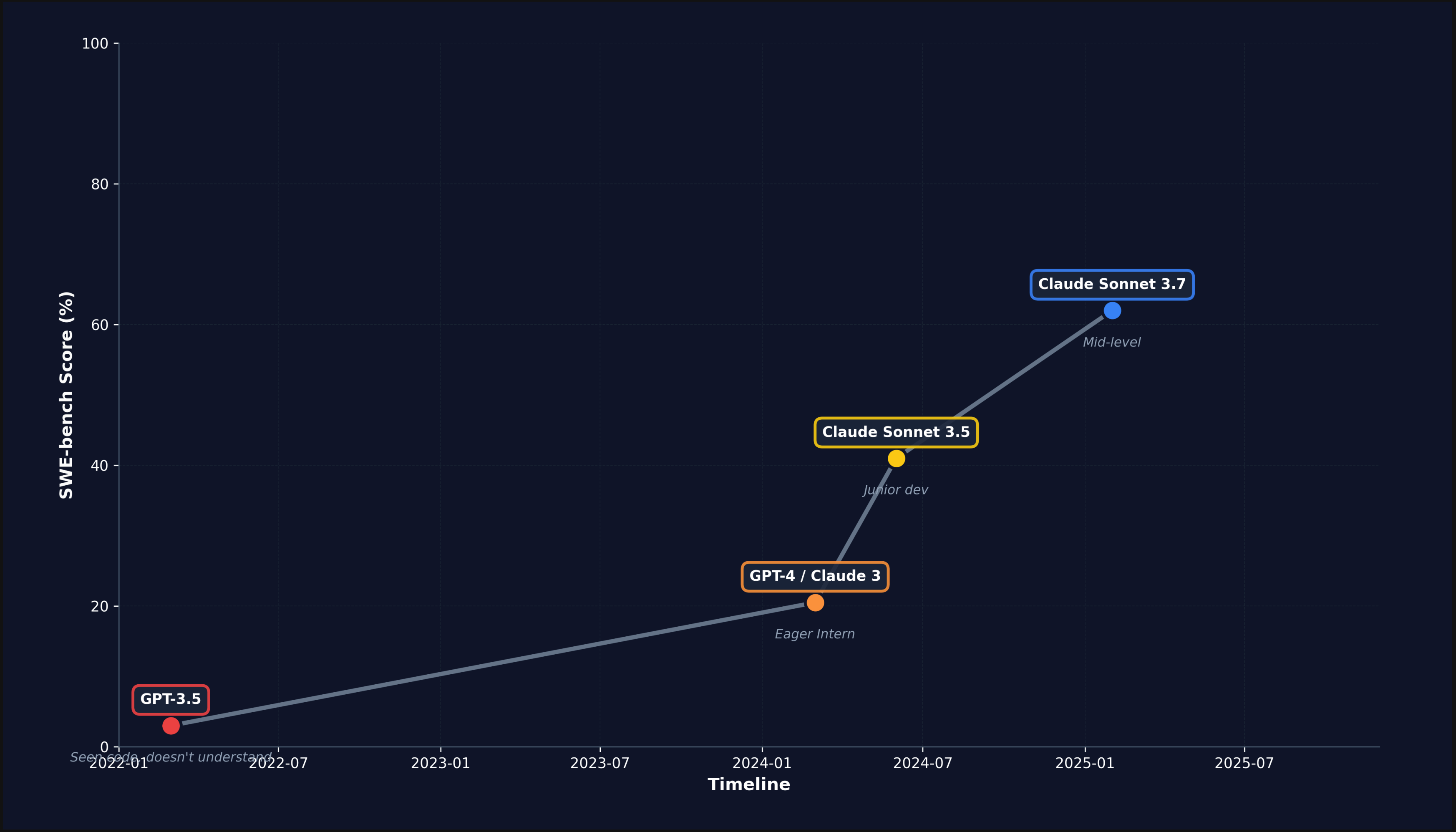
They collected 2,294 task instances by crawling Pull Requests and Issues from 12 popular Python repositories. Each instance is based on a pull request that (1) is associated with an issue, and (2) modified 1+ testing related files.


100
80
60
40
20
0
SWE-bench Score (%)
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
Timeline

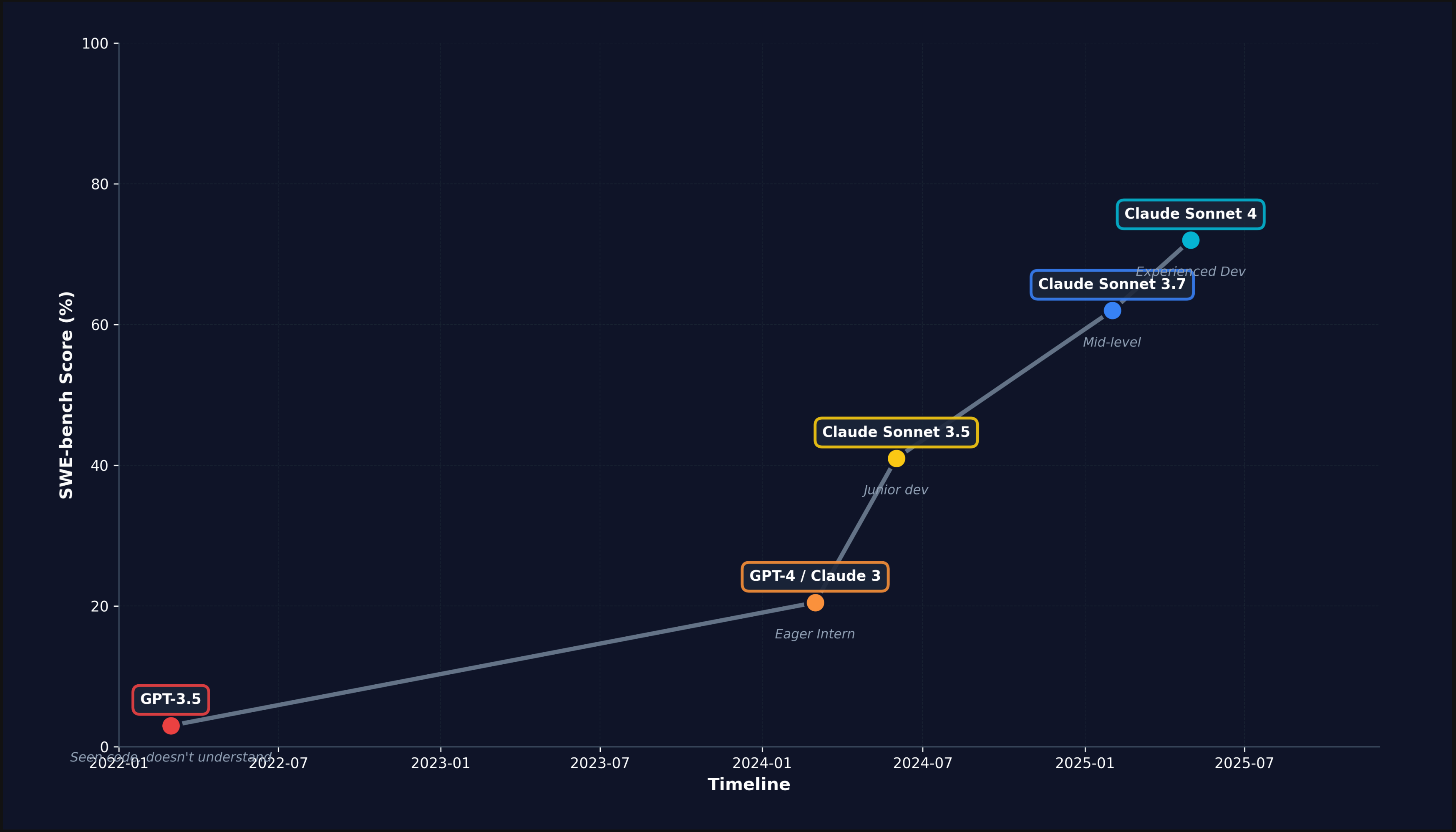

SWE-bench Score (%)
100
80
60
40
20
0
GPT-3.5
Seen code, doesn't understand
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
Timeline


SWE-bench Score (%)
100
80
60
40
20
0
GPT-4 / Claude 3
Eager Intern
GPT-3.5
Seen code, doesn't understand
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
Timeline

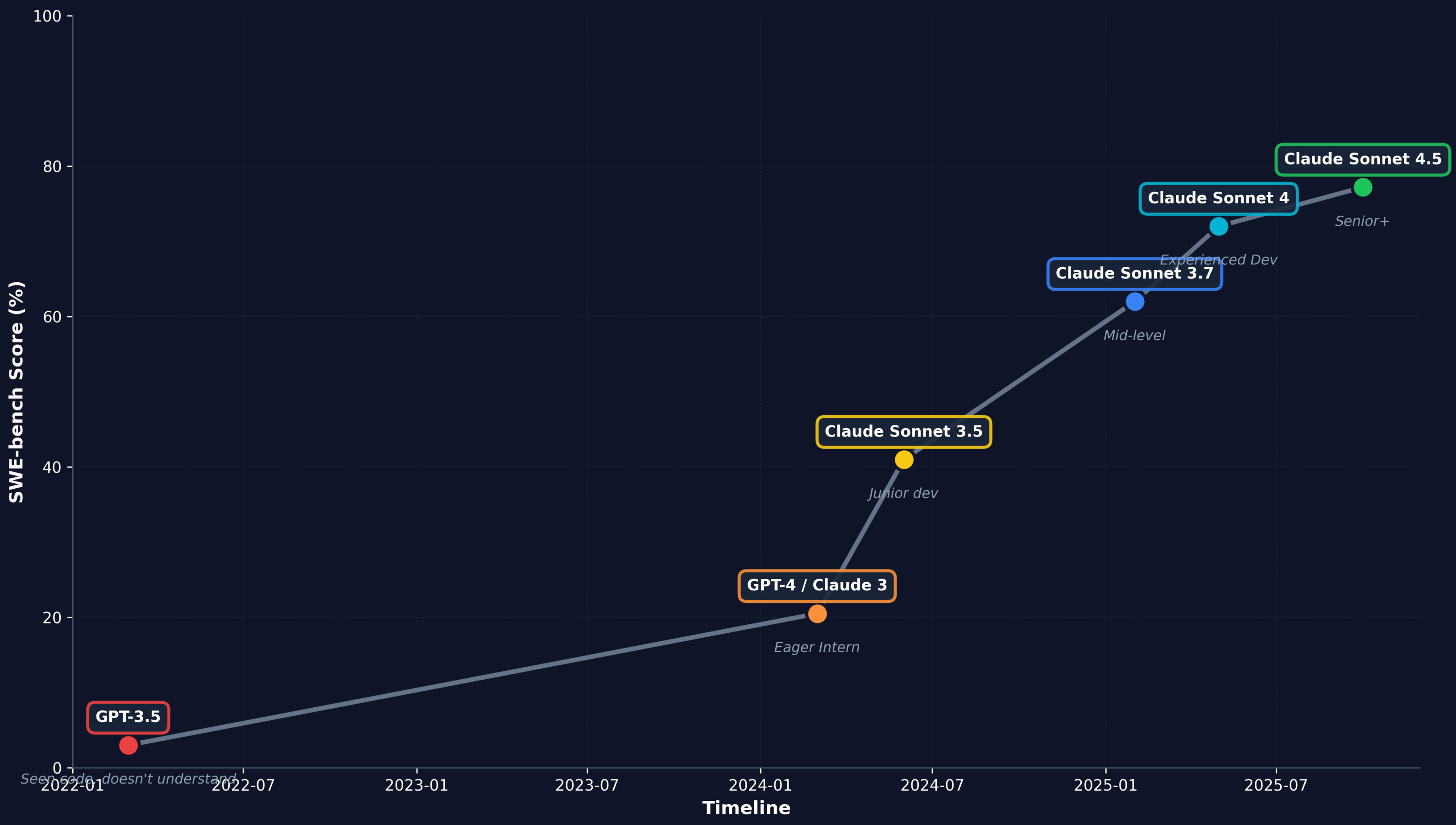

SWE-bench Score (%)
100
80
60
40
20
0
Claude Sonnet 3.5
Junior dev
GPT-4 / Claude 3
Eager Intern
GPT-3.5
Seen code, doesn't understand
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
Timeline



SWE-bench Score (%)
100
80
60
40
20
0
Timeline
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
GPT-3.5
Seen code, doesn't understand
GPT-4 / Claude 3
Eager Intern
Claude Sonnet 3.5
Junior dev
Claude Sonnet 3.7
Mid-level



SWE-bench Score (%)
100
80
60
40
20
0
Claude Sonnet 4
Experienced Dev
Claude Sonnet 3.7
Mid-level
Claude Sonnet 3.5
Junior dev
GPT-4 / Claude 3
Eager Intern
GPT-3.5
Seen code, doesn't understand
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
Timeline


SWE-bench Score (%)
100
80
60
40
20
0
Timeline
2022-01
2022-07
2023-01
2023-07
2024-01
2024-07
2025-01
2025-07
GPT-3.5
Seen code, doesn't understand
GPT-4 / Claude 3
Eager Intern
Claude Sonnet 3.5
Junior dev
Claude Sonnet 3.7
Mid-level
Claude Sonnet 4
Experienced Dev
Claude Sonnet 4.5
Senior+

What we pay per Model

💰 $3 / $15

Claude 3.5 Sonnet

💰 $3 / $15

Claude 4.5 Sonnet

~50+% better performance

The Easiest Performance Win

With new models releasing regularly, the highest-leverage improvement isn’t:

- ❌ Rewriting your prompts
- ❌ Fine-tuning a custom model
- ❌ Adding more RAG context
- ❌ Implementing complex workflows

✅ Just use the latest model

One parameter change. Huge performance boost. Same cost.

Switching Models: One Line of Code

```
import boto3
bedrock = boto3.client('bedrock-runtime', region_name='us-east-1')

response = bedrock.invoke_model(
    modelId='us.anthropic.claude-3-5-sonnet-20241022-v2:0',  # Old model
    body=json.dumps({
        "anthropic_version": "bedrock-2023-05-31",
        "max_tokens": 1024,
        "messages": [{"role": "user", "content": "Hello!"}]
    })
)
```

Switching Models: One Line of Code

```python
import boto3
bedrock = boto3.client('bedrock-runtime', region_name='us-east-1')

response = bedrock.invoke_model(
    modelId='us.anthropic.claude-sonnet-4-5-20250929-v1:0',  # New model - that's it!
    body=json.dumps({
        "anthropic_version": "bedrock-2023-05-31",
        "max_tokens": 1024,
        "messages": [{"role": "user", "content": "Hello!"}]
    })
)
```

Change one string. Get better performance. Same price.

It can work for 30+ hours straight

Without losing focus or context

Autonomous Work Duration

Here’s the game-changer: how long can it stay focused without human intervention?

Performance:

- Claude 4.0 Opus: 7 hours of focused work
- Claude 4.5 Sonnet: **30+ hours** of focused work
- Improvement: **4x longer** autonomous operation

What this means:

- Start it Friday evening, review Monday morning
- Handles complex refactors while you sleep
- Fewer "I need to ask the human" interruptions

*Requires proper feedback loops (tests, linting, etc.)

Questions & Answers

What’s the catch?

We hit throttle limits on AWS shared accounts.

- Default: 200 req/min. My team raised requests on 10/16 to raise it to 1000/min for Sonnet 4.5. Non-prod done 10/24, prod pending.
- This is actually one of the issues we brought up with AWS as being a real problem for us and trying to get them to be more proactive in helping us solve it.

Do we need to retrain our team?

No. Drop-in replacement. Change one parameter. but you made evals to verify performance. Right? 😊

"What about hallucinations/accuracy?"

25% accuracy improvement (HackerOne).

Additional Benchmarks for Sonnet 4.5

AIME 2025

Advanced Math

- With Python tools: **100%**
- Without Python tools: 87%

GPQA Diamond

Science Reasoning

- Score: **83.4%**

Response Quality

- Harmless response rate: **99.29%**
- Over-refusal rate: **0.02%** (down from 0.15%)

Official Sources

- Launch: September 29, 2025
- API: `claude-sonnet-4-5`
- Available: Amazon Bedrock, Claude.ai, Claude Code

Follow-Up Resources

To Share After Presentation:

1. Anthropic official announcement
2. API documentation
3. Case studies PDF (HackerOne, Palo Alto, IG Group)
4. Internal pilot team signup sheet
5. Baseline metrics template

For Technical Deep-Dive:

- SWE-bench methodology and results
- API migration guide (3.5 → 4.5)
- Context window optimization strategies
- Prompt engineering best practices

Thank You

Questions?

Created: 2025-10-29 | Framework: "Why You Should Use the Latest AI Models" | Version 2.0