Frontier Models: GPT, Claude, Gemini and Their Strengths and Pitfalls

Frontier and Foundation Models Overview

- Frontier models also called foundation models; terms often used interchangeably
- Foundation models serve as core building blocks for many companies
- No strict definitions; overlap in usage of both terms

Major Labs and Their Models

- OpenAI: GPT series (GPT-5 is hybrid chat/reasoning, replaces earlier models), O models (pure reasoning), GPT-4.1 (preferred for speed and chat)
- Anthropic: Claude models in Haiku, Sonnet, Opus sizes; Sonnet 4.5 is latest, Cluely 5 possibly newer
- Google: Gemini (2.5 current, 3 expected soon), chat interface called Gemini Advanced
- X AI: Grok model and chat platform (spelled with a K), not to be confused with other similarly named products
- Deepseek AI: Chinese company, open-sourced all models including largest, chat product also called Deepseek
- OpenAI also released an open-source model (OSS), possibly in response to Deepseek

Strengths of Frontier Models

- Exceptional at synthesizing and summarizing information (e.g., web pages)
- Provide detailed, structured, well-researched answers
- Strong at weighing pros and cons, generating content (emails, presentations, project plans)
- Useful for brainstorming and fleshing out new initiatives
- Outstanding coding abilities; can write, test, and iterate on code
- Have overtaken Stack Overflow as go-to resource for developers
- Claude and ChatGPT can solve complex coding problems quickly and clearly

Limitations and Risks

- Gaps in knowledge; strong in some scientific areas, weak in others
- Knowledge cutoff limits awareness of recent developments
- Can make outdated or incorrect assertions about models or tools
- Web search capability in chat products is external add-on, not part of model itself
- LLMs can make confident but dangerous mistakes (hallucinations)
- Plausibility prioritized over accuracy; can sound convincing even when wrong
- Particularly risky for junior developers who may not recognize errors
- Best used by experienced users who can supervise and validate outputs

Concrete Example of LLM Pitfall

- Student used wrong model name (base instead of chat variant) with Hugging Face
- LLM generated pages of unnecessary code to "fix" the issue, missing the simple root cause

- Junior user didn't recognize the mistake, followed misleading output
- Example highlights need for human oversight and critical thinking

Key Takeaways

- LLMs excel at content and code generation but require supervision
- Should be treated like tireless junior analysts—productive but need checking
- Users must challenge and verify outputs to avoid being misled
- Best results when LLMs are guided and their work is reviewed