LLM Models

A list of models

- Gemini 2.5 Series
- Anthropic
- Claude 4
- Anthropic Claude 3 Opus
- Evaluating AGENTS.md: Are Repository-Level Context Files Helpful for Coding Agents?

Gemini 2.5 Series

Developer/Trainer	Google
Homepage	https://deepmind.google/models/gemini/pro/

Anthropic

Anthropic is a silicon valley startup and AI company that develops and sells LLM models and LLM-based products and services. They also developed the Model Context Protocol for passing information between models.

Key Information

- Founded as a competitor to OpenAI
- Developed the Claude model series, including Claude 3 Opus, Sonnet, and Haiku
- Known for safety-focused AI development approaches
- Created the Model Context Protocol (MCP) standard

Relationship to Other Models

Anthropic are one of the earliest competitors to OpenAI and their Claude Sonnet and Opus series models were early competitors to GPT-4.

Claude Models

The Claude model series includes several versions:

Claude 3 Series

- Anthropic Claude 3 Opus
- Anthropic Claude 3 Sonnet
- Anthropic Claude 3 Haiku

Other Models

- Anthropic Claude 2
- Anthropic Claude 1

Related Content

- Model Context Protocol

Claude 4

Claude 4 is a large language model developed by Anthropic.

Key Features

- Advanced reasoning capabilities
- Improved safety measures
- Better performance on complex tasks

Anthropic Claude 3 Opus

Overview

Claude 3 Opus is Anthropic's most powerful model, designed for highly complex tasks that require the highest levels of reasoning and understanding.

Capabilities

- Advanced reasoning and problem-solving
- Complex content creation and editing
- Comprehensive understanding of nuanced topics
- High-quality code generation and debugging

Performance

- More capable than Claude 3 Sonnet and Haiku
- Excellent at handling complex reasoning tasks
- Strong performance in benchmarks like MMLU and GSM8K

Use Cases

- Research and analysis
- Complex content creation
- Code generation with extensive reasoning

```
flowchart LR


A[Hard] -->|Text| B(Round)
B --> C{Decision}
C -->|One| D[Result 1]
C -->|Two| E[Result 2]
```

Evaluating AGENTS.md: Are Repository-Level Context Files Helpful for Coding Agents?

Title: Evaluating AGENTS.md: Are Repository-Level Context Files Helpful for Coding Agents?

Authors: Thibaud Gloaguen, Niels Mündler (ETH Zurich), Veselin Raychev, Mark Müller, Martin Vechev (LogicStar.ai / ETH Zurich)

Published: June 2026 — arXiv:2602.11988

Link: https://arxiv.org/abs/2602.11988

Summary

A widespread practice in software development is to tailor coding agents to repositories using context files (AGENTS.md, CLAUDE.md), strongly encouraged by agent developers. This paper presents the first rigorous evaluation of whether such context files actually improve task completion performance.

Key findings:

- **LLM-generated context files do NOT improve success rates** — they slightly reduce them (−0.5% on SWE-Bench, −2% on CTX-Bench), though not statistically significantly
- **They increase inference cost by over 20%** on average, as agents follow instructions and run more tests/exploration
- **Developer-written context files outperform LLM-generated ones** by ~7%, but the improvement over having no file is marginal (p=21%)
- **Repository overviews are not helpful** — agents don't discover relevant files faster with an overview section
- **Instructions ARE followed** — if a tool is mentioned in the context file, usage increases dramatically. The cost increase comes from more thorough testing, not ignoring

instructions

The authors created **CTX BENCH**, a new benchmark of 138 real-world Python tasks from 12 niche open-source repos with developer-committed context files, complementing SWE-Bench Lite evaluation.

Conclusion: Context files should only contain non-standard coding practices not already in the README. LLM-generated ones are not worth it yet. Any attempts to improve performance should be rigorously evaluated before deployment.