

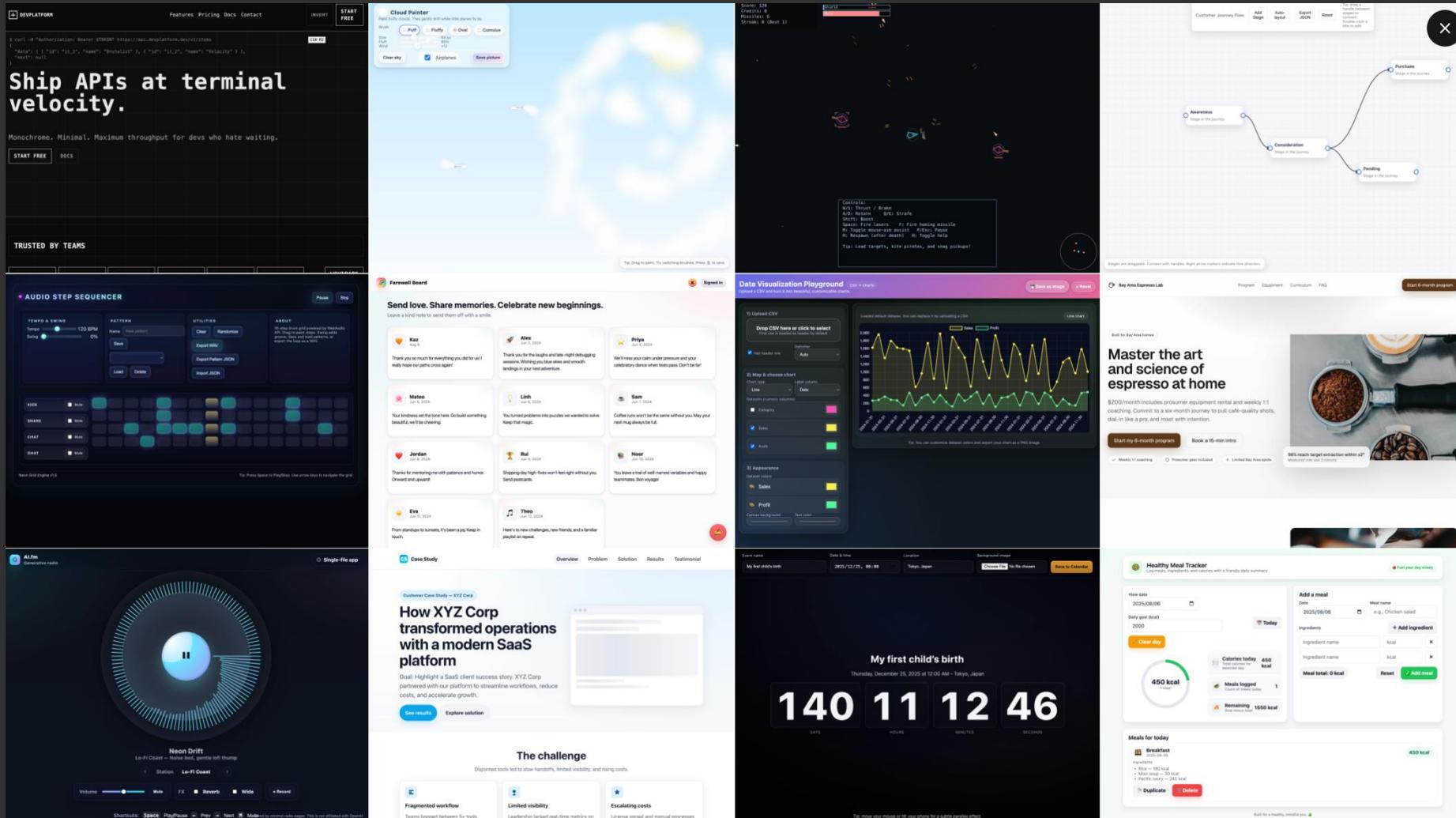
Vibe Coding w. Codex

박종현

목표. LLM 을 잘 사용 하는 방법 알기

- 남들은 어떻게 LLM 을 활용하고 있는가?
 - Research & Coding Agents!
- 좋은 예시 – OpenAI DeepResearch (+Operator = Agents)
 - OpenAI o3 를 fine-tuning 하여 만들어진 Agent
 - 딥리서치 팀 인터뷰 - <https://youtu.be/bNEvJYzoa8A?si=EWY7AiqZAF9TqOvc>
- Coding 이라는 문제를 풀어보면서 사용 방법 익히기
 - 다른 문제에도 적용이 가능할지...

Intro – GPT5 Coding Gallery



<https://platform.openai.com/docs/guides/latest-model?gallery=open>

Intro – 추석 동안 나온 업데이트

- OpenAI DevDay 2025
 - Codex SDK
 - Agent Builder
 - Codex with Slack
 - ChatGPT with App
- Claude Code 2.0
 - <https://www.youtube.com/watch?v=Yct0MvNtdfU>
 - Claude Sonnet 4.5
- Google Gemini
 - <https://www.youtube.com/watch?v=uLHF9T1SLrU>

VS Code Setup

- IDE (Integrated Development Environment)

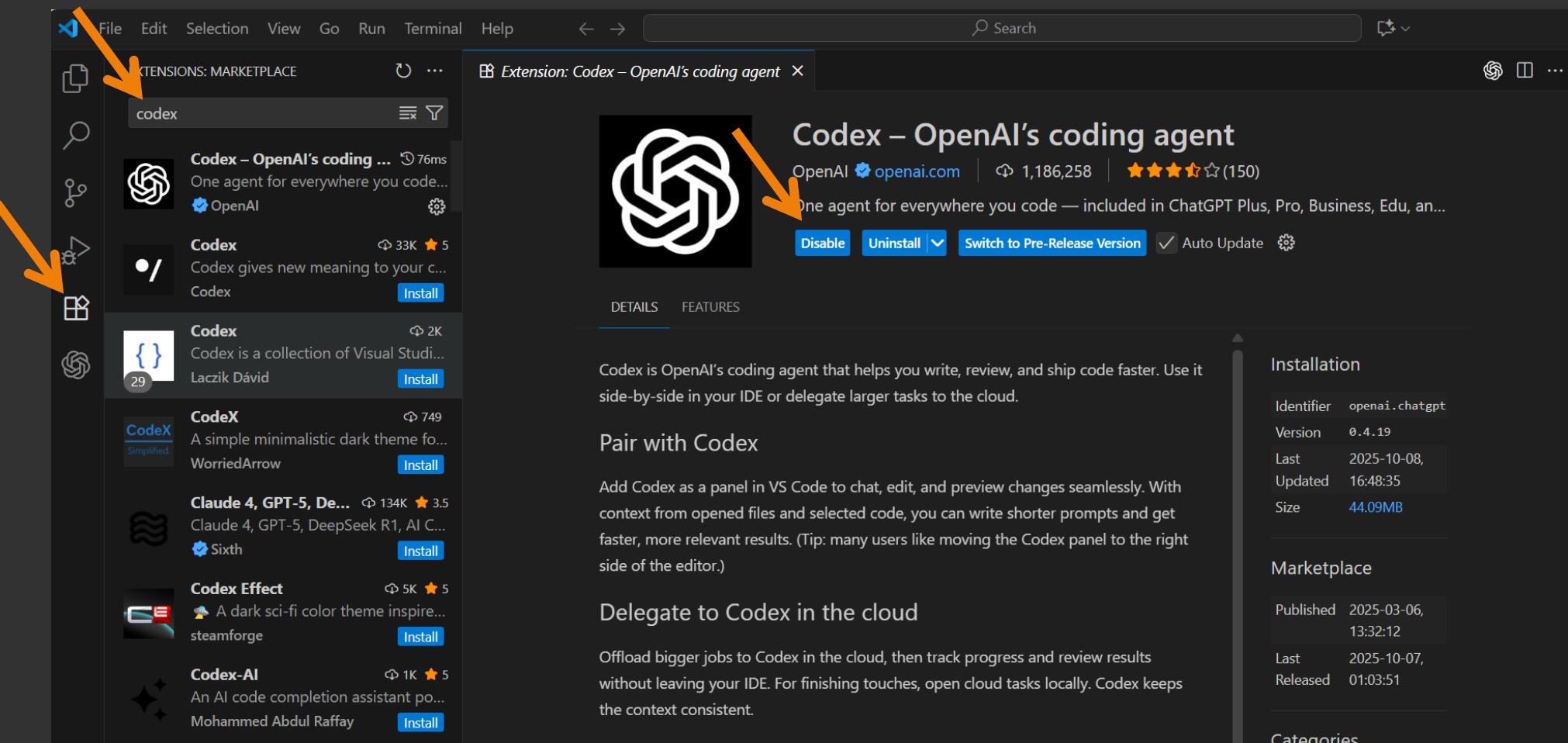
- 개발, 코딩 편하게 해주는 프로그램
 - VSCode
 - Cursor
 - Etc...

- VS Code 다운로드 & 설치

- <https://code.visualstudio.com/> 접속 후 다운로드
 - OS 에 맞게 설치

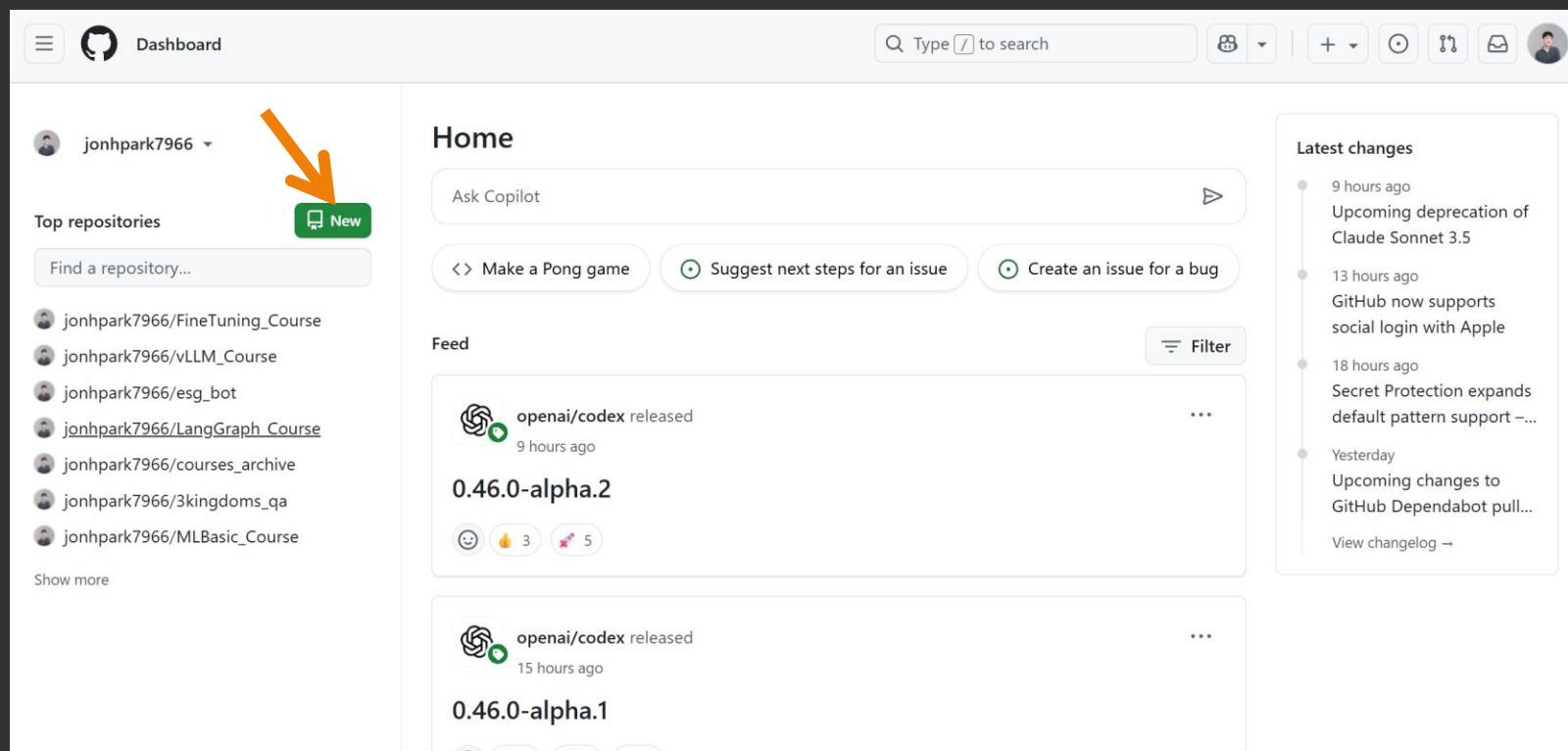
VS Code Setup

○ Codex Extensions on Marketplace



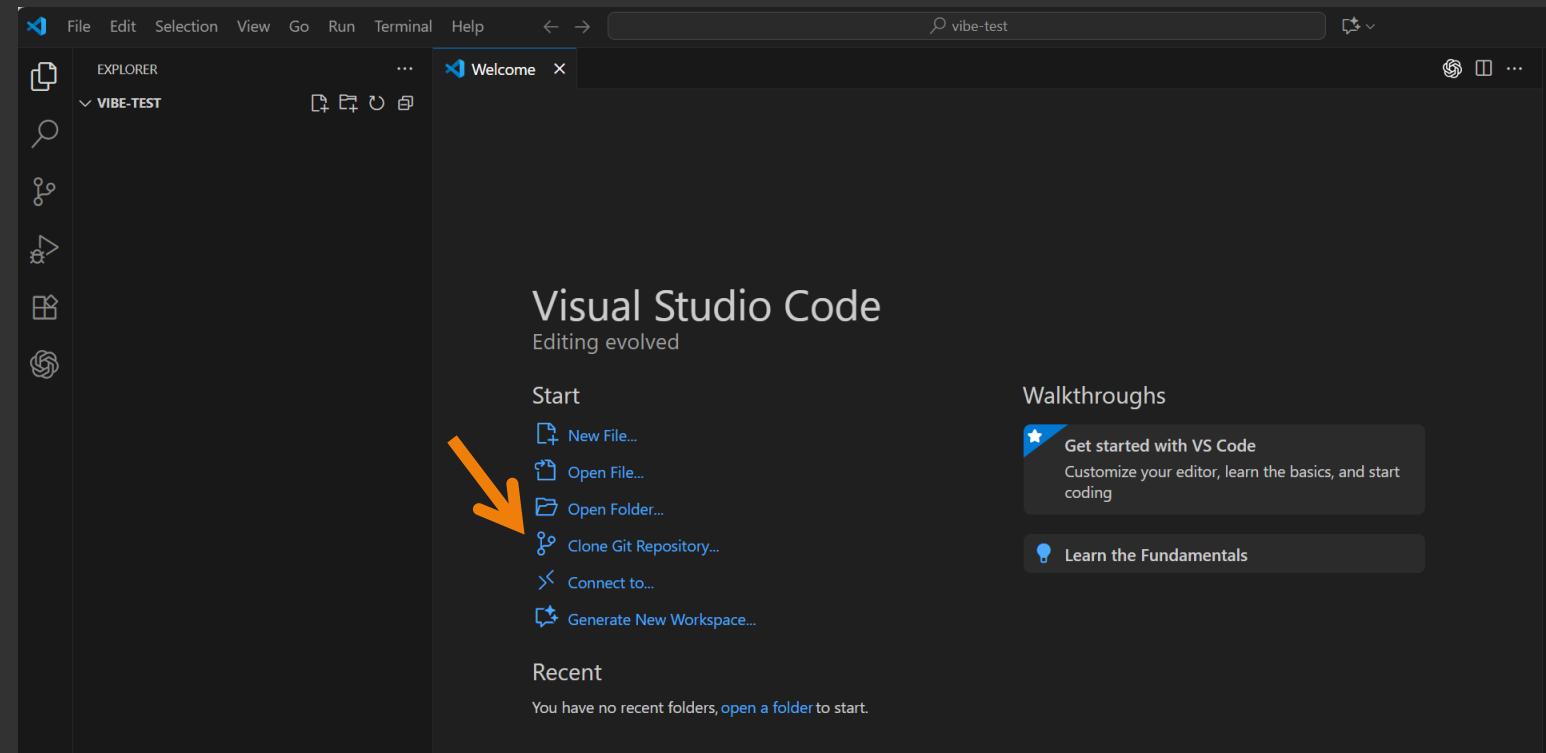
github Setup

- <https://www.github.com>
- Create git repository
- (cont) Clone git repo
- Github != git
- Git = Distributed Version Control System
- 협업에 매우 좋음 !!
→ AI 와의 협업에도 좋음



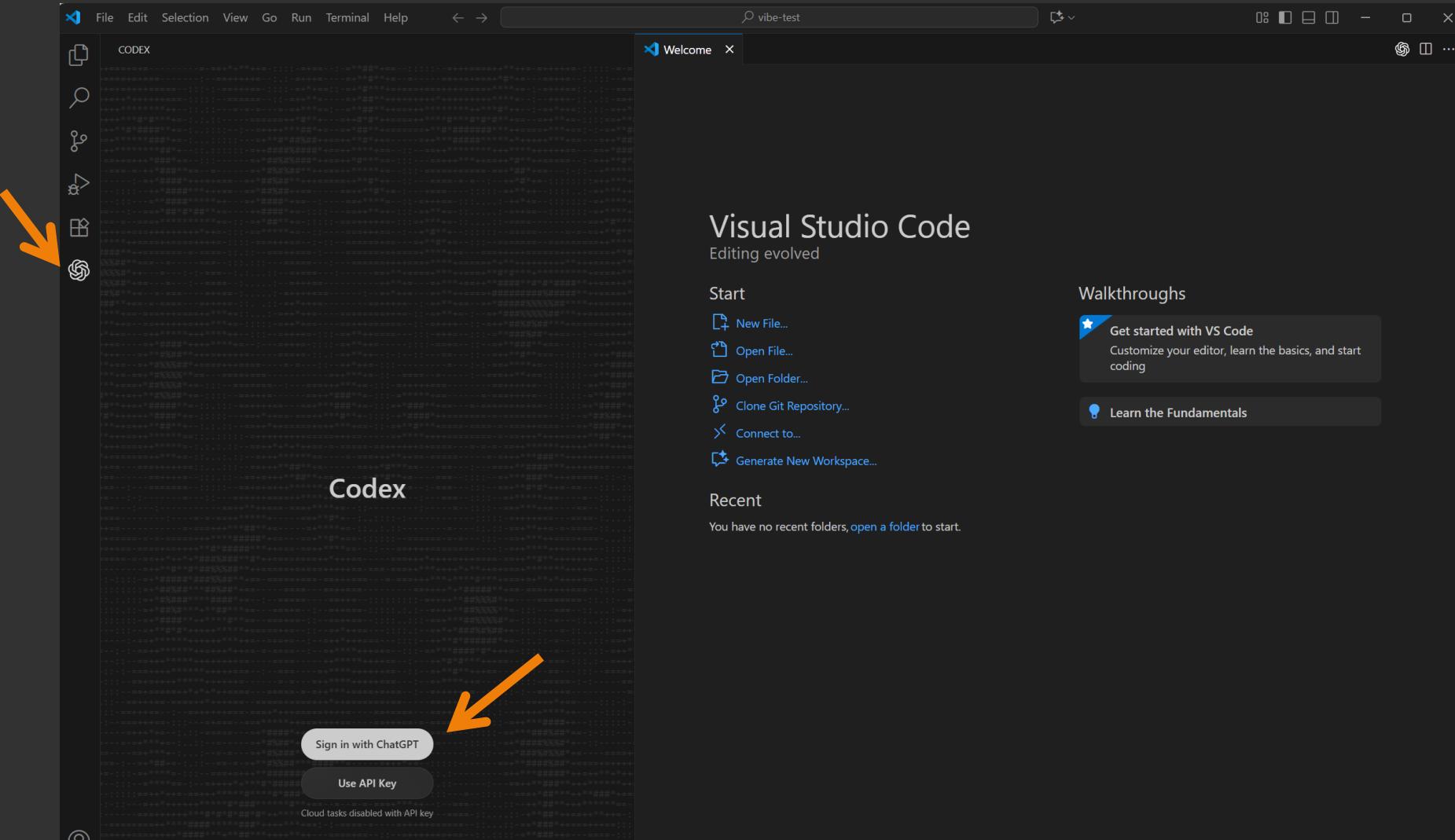
VS Code & github clone

- Clone Git Repository
- Github extension install
- Login with github
- clone



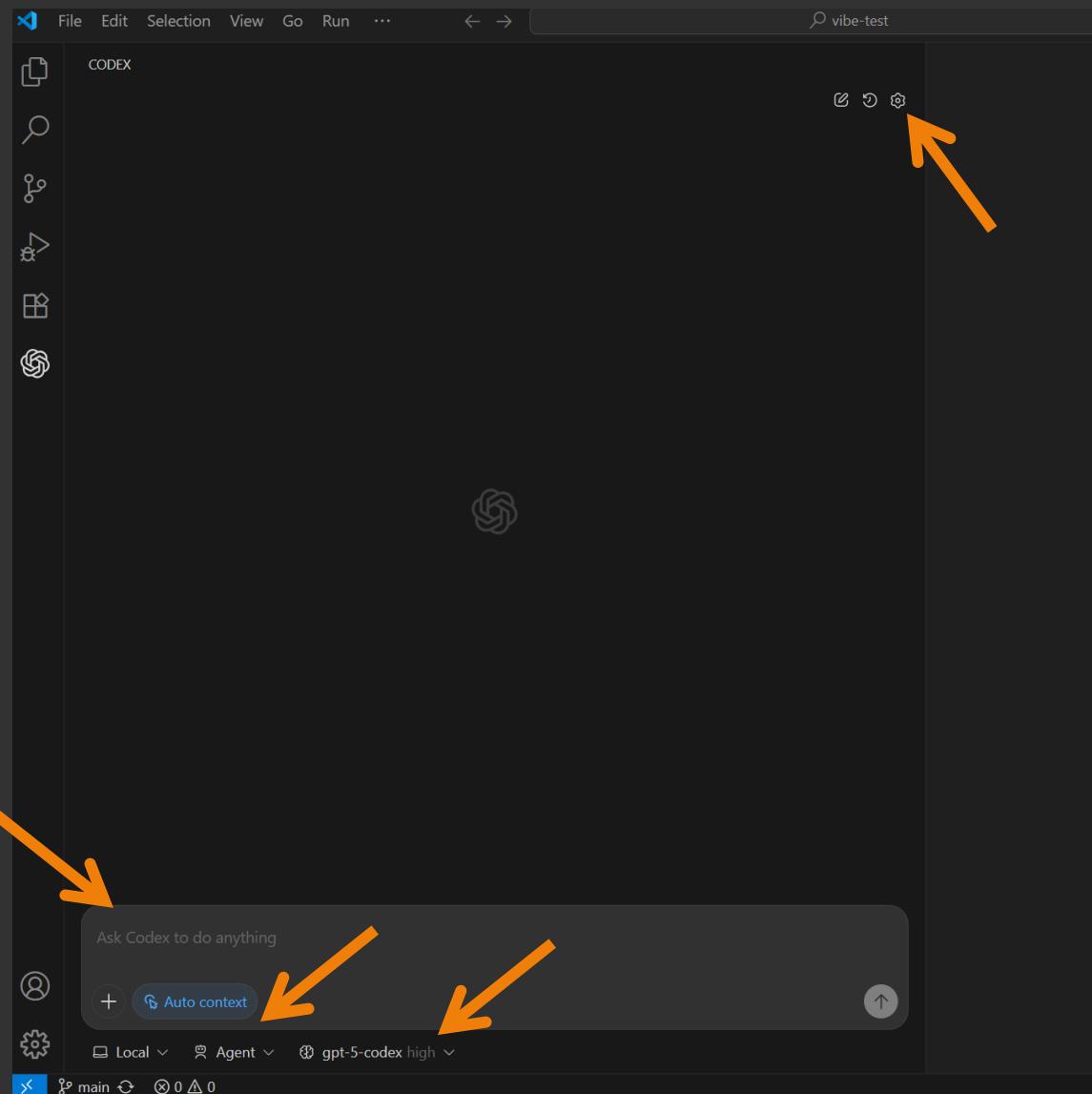
Sign in with ChatGPT

- 유료 계정으로 로그인
- 5시간 / 주간 limit
- 다른 도구들도 비슷



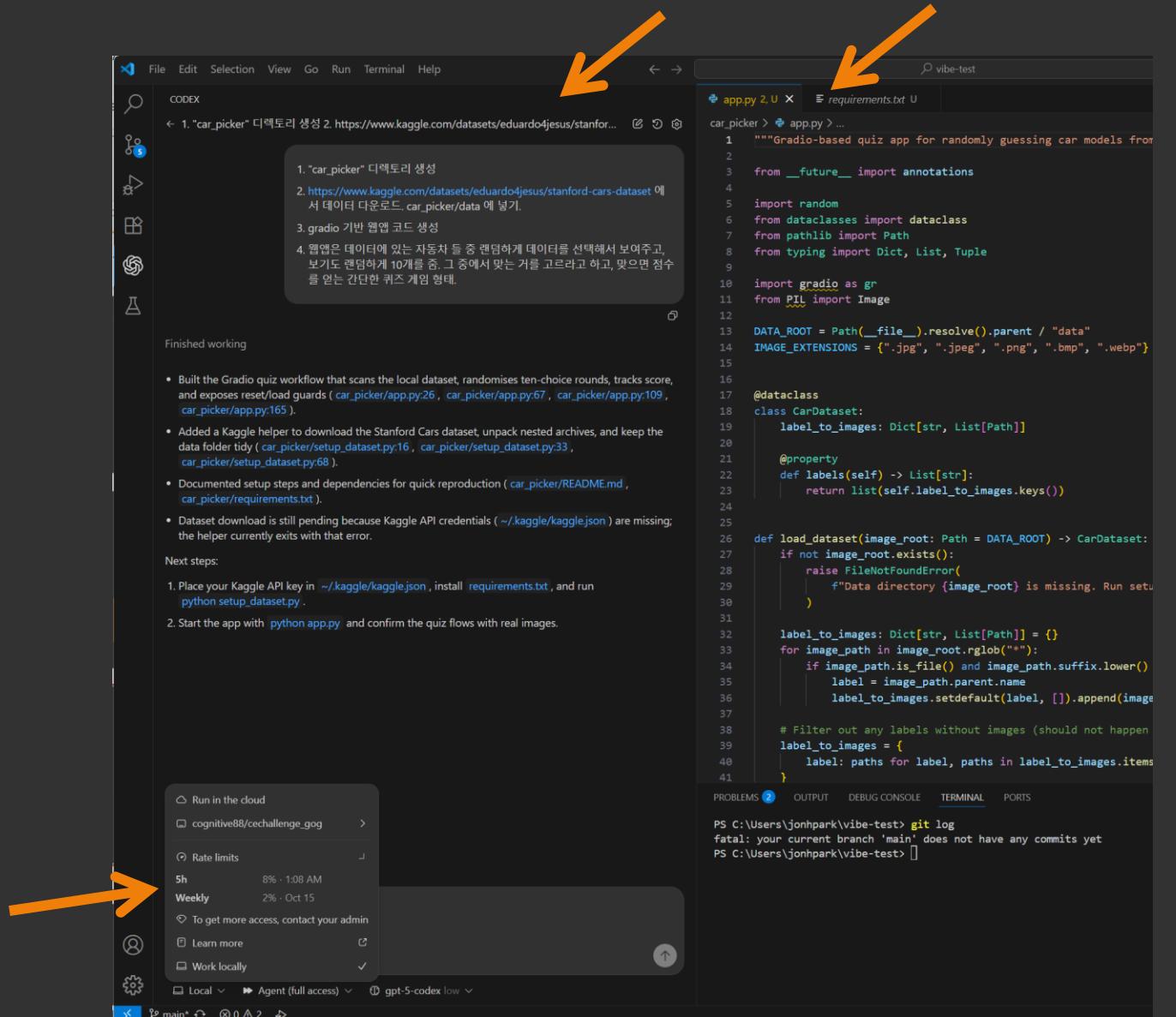
Codex IDE Extension

- Limit 확인
- 로컬 수행
- Agent 모드 설정
- 모델 변경
- 전체 codex 설정 (TBD)



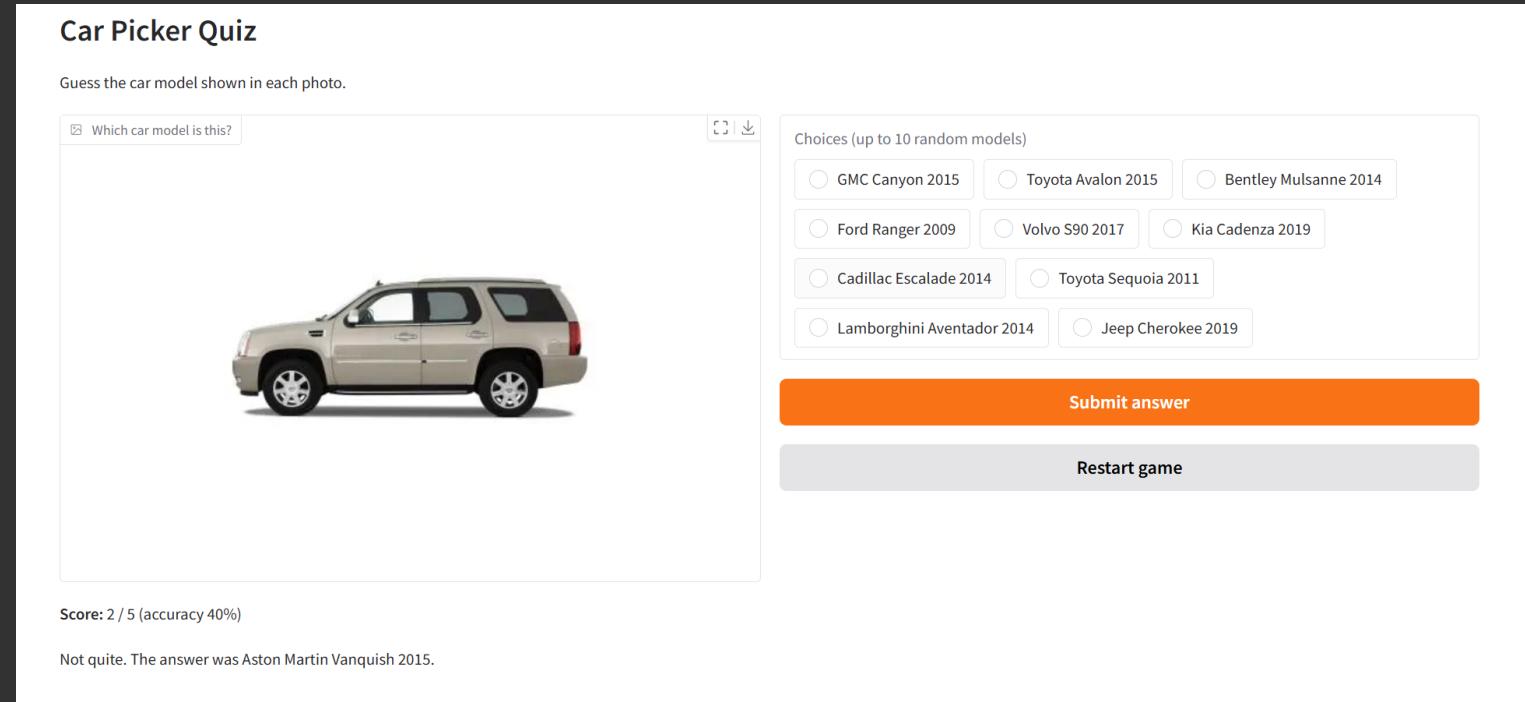
Zero shot Prompting

- 로컬에 필요한 도구 설치
 - 알아서 프로그래밍
 - 결과 확인 & Limit 확인



추가로 해야 할 셋업

- Python
 - vscode 다시 켜보기
 - 안되면 추가 셋업 필요
 - pip –r ./requirements.txt
 - 필요한 도구들 설치하기
- Data 연결하기
- Furthermore
 - Python virtual environment
 - Sandboxing



Github commit and push

- Git 설정

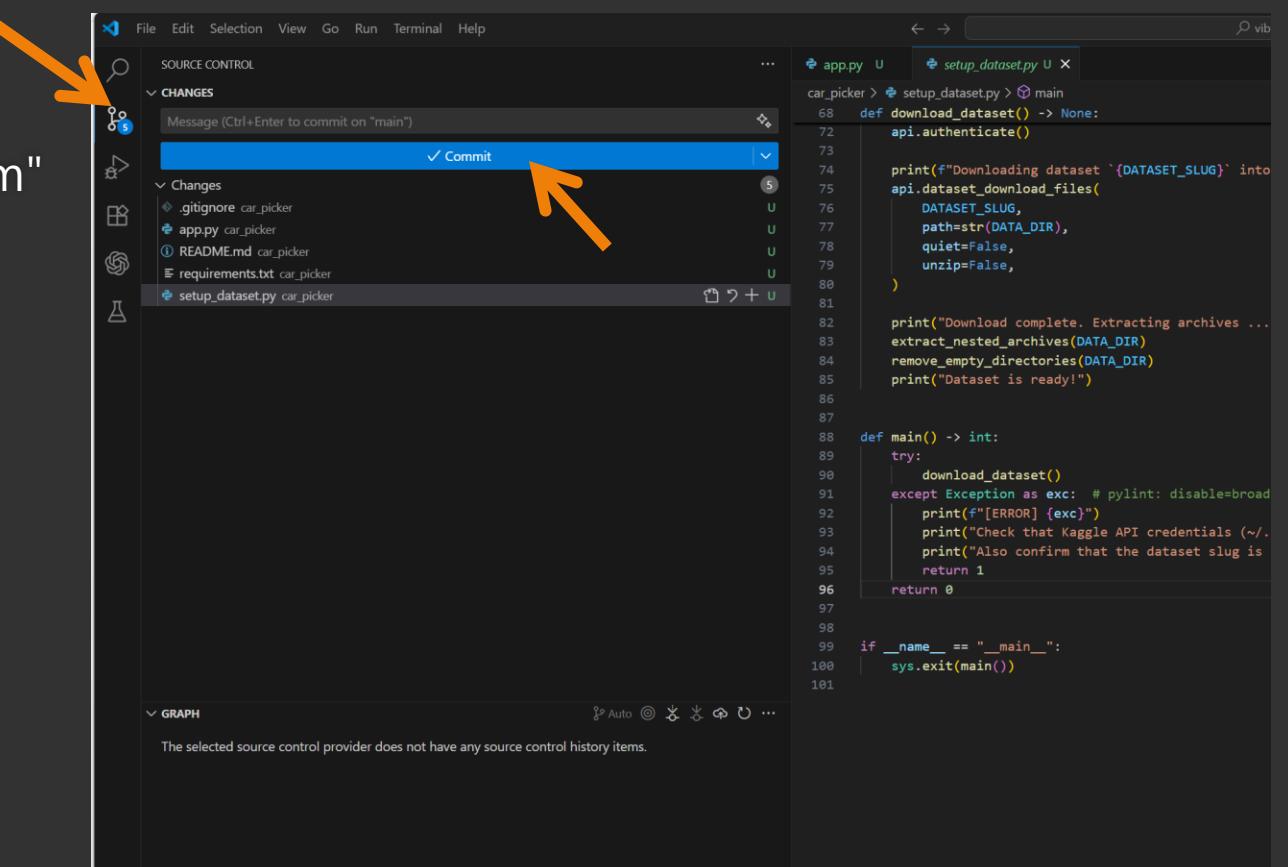
```
git config --global user.name "Your Name"
```

```
git config --global user.email "you@example.com"
```

- Git commit

- Git push

- Github에서 확인



The screenshot shows the Visual Studio Code interface with the Source Control extension open. The 'CHANGES' tab is selected, displaying a list of files with status indicators (S, U). A message input field at the top says 'Message (Ctrl+Enter to commit on "main")'. Below it is a large blue button labeled '✓ Commit'. Two orange arrows point from the left towards the 'Commit' button. On the right side of the interface, there is a code editor window showing Python code related to dataset download and extraction.

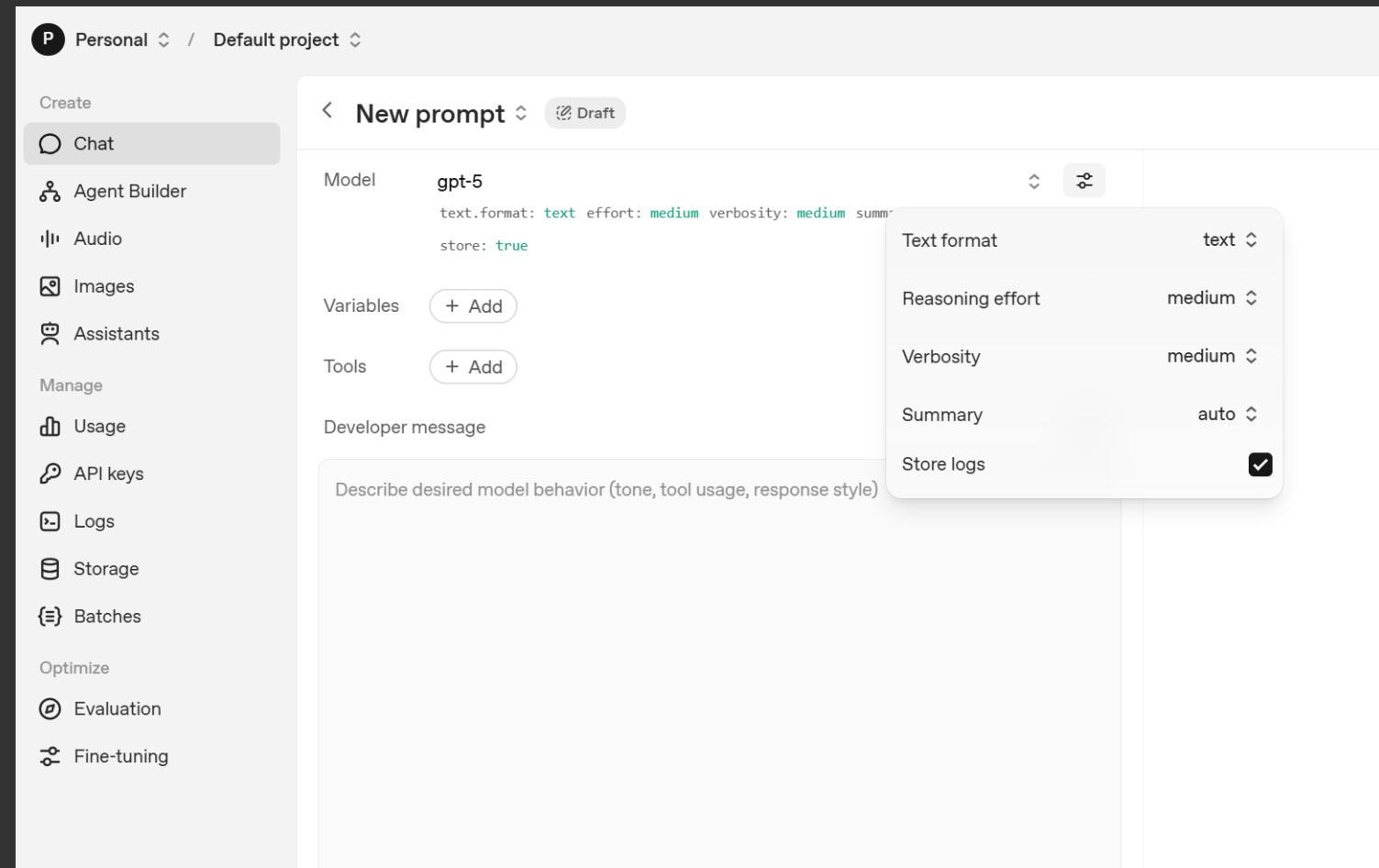
```
File Edit Selection View Go Run Terminal Help
SOURCE CONTROL ...
CHANGES
Message (Ctrl+Enter to commit on "main")
✓ Commit
Changes
.gitignore car_picker
app.py car_picker
README.md car_picker
requirements.txt car_picker
setup_dataset.py car_picker
GRAPH
The selected source control provider does not have any source control history items.
68 def download_dataset() -> None:
72     api.authenticate()
73
74     print(f"Downloading dataset '{DATASET_SLUG}' into
75         api.dataset_download_files(
76             DATASET_SLUG,
77             path=str(DATA_DIR),
78             quiet=False,
79             unzip=False,
80         )
81
82     print("Download complete. Extracting archives ...
83     extract_nested_archives(DATA_DIR)
84     remove_empty_directories(DATA_DIR)
85     print("Dataset is ready!")
86
87
88 def main() -> int:
89     try:
90         download_dataset()
91     except Exception as exc: # pylint: disable=broad
92         print(f"[ERROR] {exc}")
93         print("Check that Kaggle API credentials (~/.kaggle/kaggle.json) are
94         print("Also confirm that the dataset slug is correct")
95         return 1
96
97
98
99 if __name__ == "__main__":
100     sys.exit(main())
101
```

ChatGPT 사용해보기

- GPTs
 - 나만의 GPT 를 사용해서 어떤 기능들이 있는지 살펴보기
 - Web Browsing
 - RAG
 - Prompting
- App 연동하기
- Deep Research
- Reasoning Efforts

GPT 사용해보기

- Openai Platform → <https://platform.openai.com>
- GPT != ChatGPT
- Reasoning Effort
- Chat Completion
- Responses
- Files
- Fine Tuning
- Prompting 에 대한 고찰
 - OpenAI's Prompting Guide



Codex Workflow (by OpenAI)

- Start with the Codex CLI to generate code for a new project
 - Open your preferred IDE to make edits, assisted by the Codex IDE extension
 - If you want to build new features that are pretty much independent from the current codebase, you can delegate this to the Codex cloud agent (e.g. adding auth, connecting to a database, adding new pages, etc.)
 - Review the changes in the Codex interface and create PRs on GitHub
 - Check out the PR locally and test the changes
 - If changes are needed, you can iterate on the PR on GitHub by tagging @codex in a comment
 - While this is happening, you can continue working on other tasks in your IDE
 - Once you are satisfied with the changes, you can merge the PR
 - Repeat the process with other tasks
-
- 참조 - <https://developers.openai.com/codex/quickstart>

Concept

○ Local Task

- Codex will directly interact with your local file system to modify files and run commands.
- To limit the risk of Codex modifying files outside of your workspace, or perform other undesired actions, local tasks are run in a sandbox environment by default.

○ Sandbox

- For Windows, we recommend you to run Codex locally in Windows Subsystem for Linux (WSL)
- 가능하다면, 맥이나 리눅스로 갈아타시는 것을 추천 드립니다

○ Cloud Task

- 클라우드 컴퓨터에서 작업 시키기. OpenAI 서버에서 알아서 코딩하고 실행하고 버그잡고 수행함 (!!)

Codex CLI / Codex IDE extension / Codex Cloud

- **Codex CLI**

- <https://developers.openai.com/codex/cli>
- The Codex CLI officially supports macOS and Linux. Windows support is still experimental — we recommend running in WSL.
- 개발자들이 가장 선호하는 형태의 도구, 25.10.08 기준 windows 미지원

- **Codex Ide Extension**

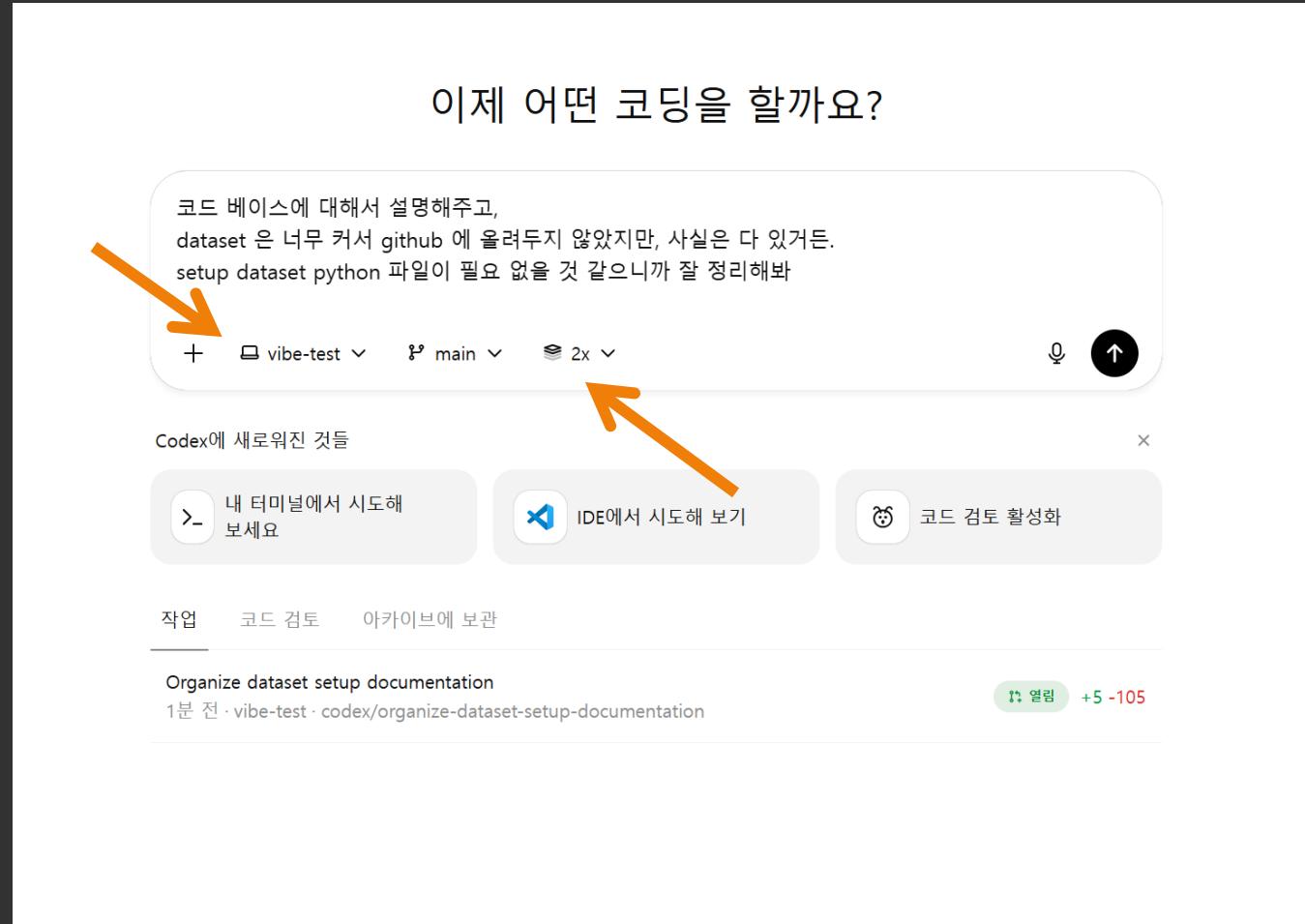
- 사용 중

- **Codex Cloud**

- Chatgpt Codex 에서 대화하며 물어보고 일 시키기

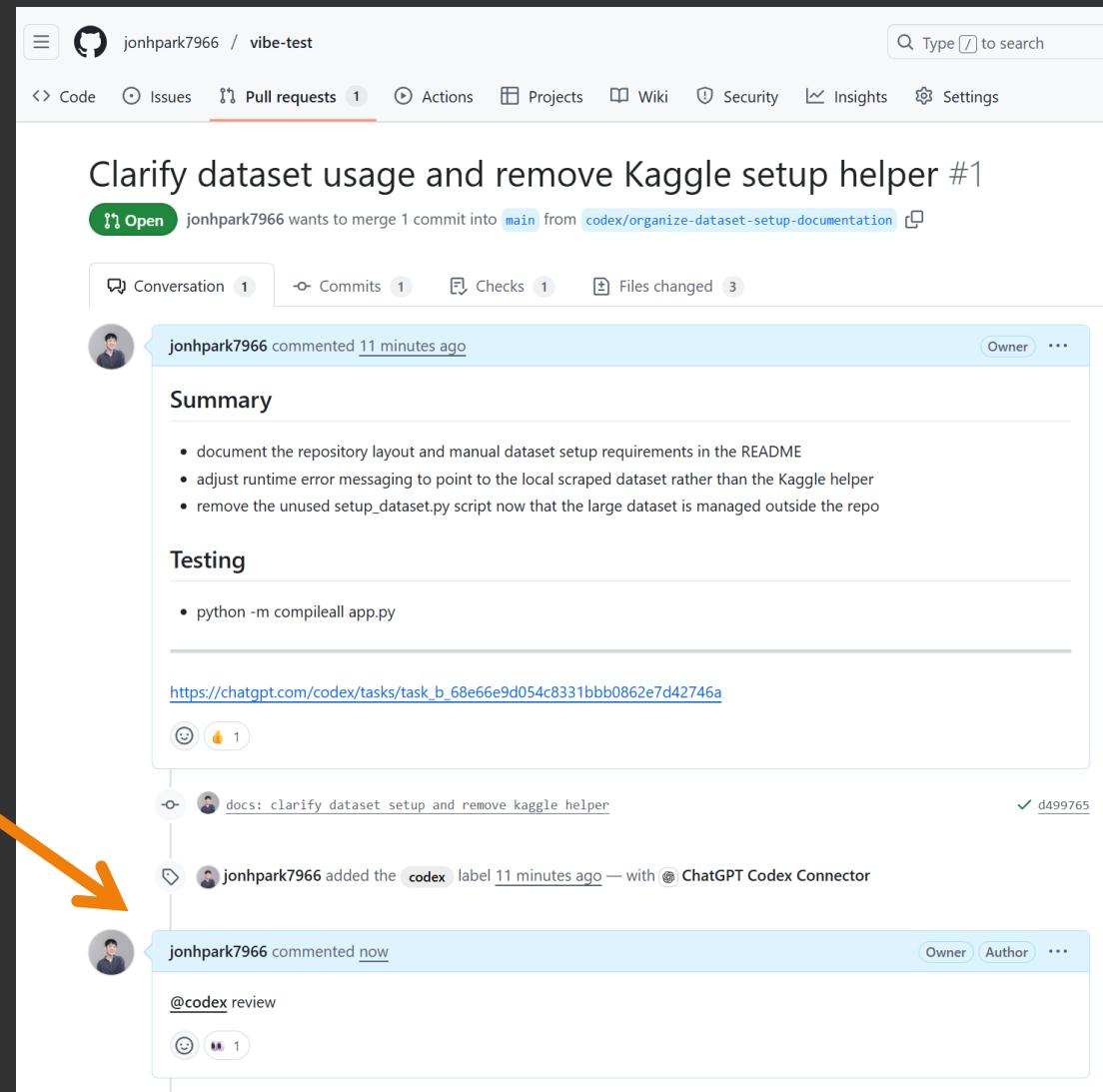
Codex Cloud 사용해보기

- <https://chatgpt.com/codex>
- 프롬프트로 일 시키기
- Github repo 설정
- 복수 버전 생성 가능
- 코드 검토 활성화
 - 자동 코드 리뷰 - github 연동 설정
- Planning 만 따로 가능
- 일 위임하기



Codex Cloud 사용해보기

- @codex review
- 커멘트를 남겨서 리뷰 시키기
- 괜찮다면 accept & merge 해서 반영

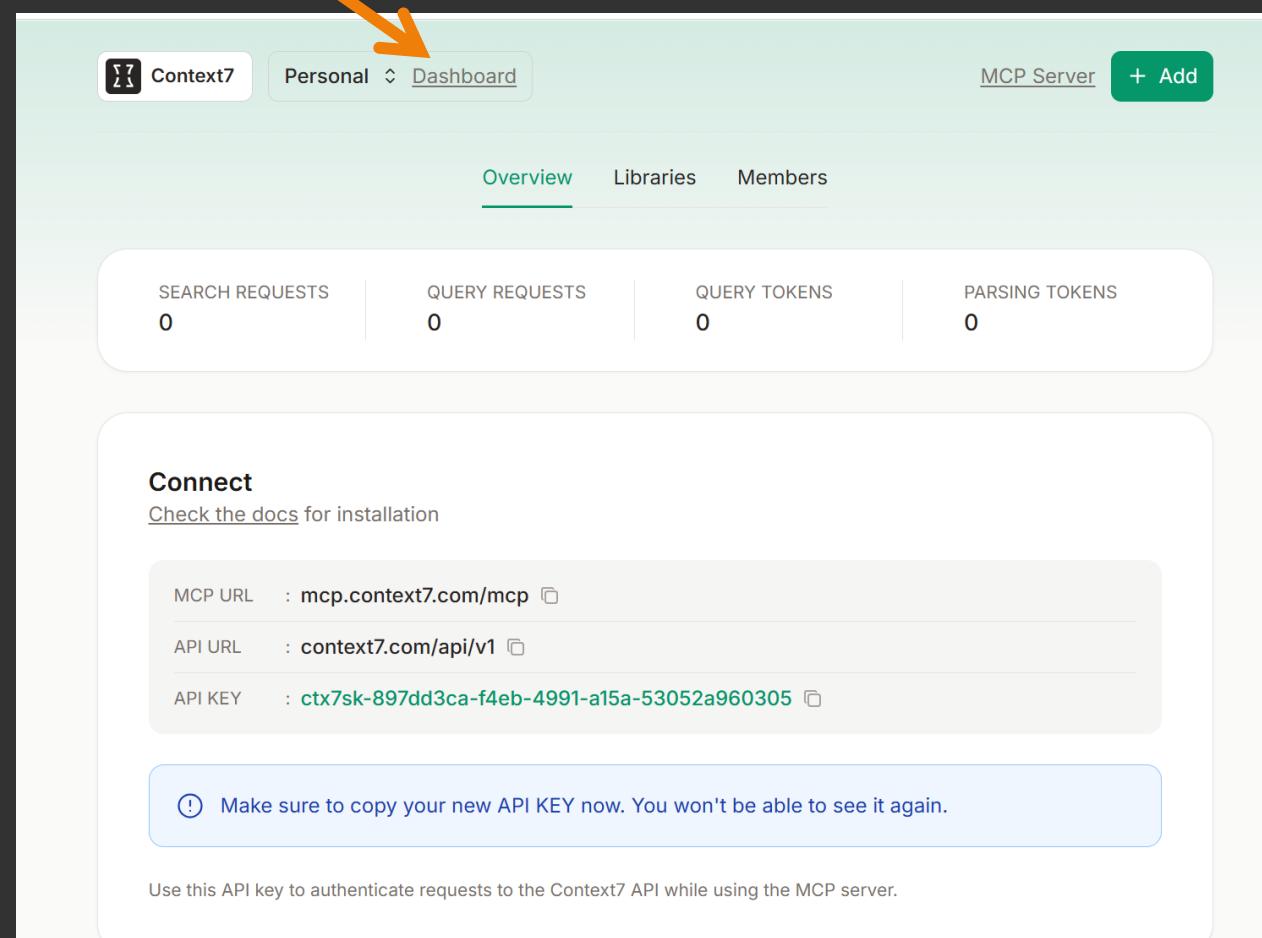


Claude Code vs Cursor vs Codex ...

- Claude Code
 - 개발자들이 가장 선호하는 도구
 - 이유는 단순, 가장 똑똑해서.
 - CLI 기반 == Codex CLI
 - Opus & Sonnet 이 코딩을 잘하고 tool 도 잘 사용함
- Cursor
 - 센세이셔널한 등장
 - VS Code 를 이용해서 개량한 IDE
 - 코딩 에이전트 시대를 열었음
 - Model 은 대부분 sonnet 이나 gpt5 를 사용
 - 모델이 아닌 잘 만든 harness
- 그 외 많은 도구들
 - 구글의 Gemini CLI, Jules
 - Lovable, Replit, V0, ...
 - 특정 framework 이나 디자인에 특화
- 결론
 - 모델 제조사 (OpenAI, Google, Anthropic, Alibaba)를 제치고 Harness 제조사가 이길 수 있을까?
 - 편리한 UI/UX 가 중요한가?

MCP

- Model Context Protocol
- LLM 이 보조도구를 잘 사용할 수 있게함
- (현재는) 도구가 너무 많으면 LLM 이 정신을 못 차림
- 꼭 필요한 도구들을 잘 연결하면 좋음
- 예시 - Context7
(<https://github.com/upstash/context7>)
 - 최신 문서를 참조할 수 있도록 도와줌
 - <https://context7.com/>



MCP – context7 설정하기

- Config.toml 파일에 MCP 추가 설정
- Context7 은 api-key 발급받아서 설정)

참조 - <https://github.com/upstash/context7>

[mcp_servers.context7]

```
args = ["-y", "@upstash/context7-mcp", "--api-key", "YOUR_API_KEY"]
```

```
command = "npx"
```

- Node js 다운 필요
(<https://nodejs.org/en/download>)
- 필요시 windows 전용 우회 셋팅 필요

The screenshot shows a terminal window with two tabs: 'app.py' and 'config.toml'. The 'config.toml' tab is active, displaying the following configuration:

```
C: > Users > jonhpark > .codex > config.toml
1 model = "gpt-5-codex"
2 model_reasoning_effort = "high"
3
4 [mcp_servers.context7]
5 args = ["-y", "@upstash/context7-mcp", "--api-key", "ctx7sk-468a4l"]
6 command = "npx"
```

Orange arrows point from the text above to the 'config.toml' tab and the 'args' section of the configuration. A callout box is overlaid on the terminal window, pointing to the 'args' section, containing the following explanatory text:

← 1. "car_picker" 디렉토리 생성 2. https://www.kaggle.... 🔍 ⓘ ⓘ

/car_picker/data 에 넣었음. 데이터에 대한 설명은 <https://github.com/nicolas-gervais/predicting-car-price-from-scraped-data/tree/master/picture-scraper> 참조.

메이커 모델 년식 정도 만 추출해서 자동차 퀴즈 웹페이지로 만들어줘.

The terminal also displays a message about finishing work and a list of updates.

MCP – context7 설정하기

- 셋팅이 완료되면, codex 가 최신 문서를 MCP 를 이용해서 열람할 수 있음

The screenshot shows a terminal window with the following content:

```
File Edit Selection View Go Run Terminal Help
← →
app.py config.toml X
C: > Users > jonhpark > .codex > config.toml
1 model = "gpt-5-codex"
2 model_reasoning_effort = "low"
3
4 [mcp_servers.context7]
5 command = "cmd"
6 args =
7   "/c",
8   "npx",
9   "-y",
10  "@upstash/context7-mcp",
11  "--api-key",
12  "ctx7sk-468a4b84-6b60-4ab1-b60e"
13 ]
14 env = { SystemRoot="C:\\Windows" }
15 startup_timeout_ms = 20_000
```

On the left, there is a sidebar with icons for search, file, folder, and a neural network. Two orange arrows point from these icons to the corresponding sections in the terminal output.

The terminal output shows the following steps:

- read gradio documentation through context7
- Finished working
 - Calling resolve function
 - MCP: context7.resolve-library-id("gradio")
 - Selecting gradio directory
 - MCP: context7.get-library-docs({"context7CompatibleLibrary": ...})
 - Preparing summary response
- Pulled the latest Gradio docs from Context7 (/gradio-app/gradio). Let me know if you want me to dig into any specific area—components, Blocks, client APIs, etc.

Agents.md

- Codex (또는 다른 Coding Agent) 에게 알려주는 매뉴얼
- <https://agents.md/> 에 수많은 프로젝트들의 agents.md 파일 예시가 있음
- Codex 의 agents.md 를 한번 봅시다!
<https://github.com/openai/codex/blob/main/AGENTS.md>
- 보통은 이 외에도 수 많은 md 파일 (ex. TODO.md, Plans.md) 로 문서를 만들고, 이를 보고 코딩을 하도록 가이드하는 것이 일반적!

Designs

- **Figma**
 - Adobe 가 인수한 디자인 도구 (디자이너 → 프론트엔드 개발자 협업에 많이 사용됨)
 - Chat interface 로 사용 가능
 - 만들어진
 - 25.10.10, Figma MCP 발표
 - Figma MCP server now supports Gemini CLI and OpenAI Codex (<https://www.youtube.com/watch?v=ubEZ4fX2kUI>)
- **Lovable**
 - <https://lovable.dev/>
 - 갤러리에 가면 수 많은 예시들을 볼 수 있음
- 그냥 간단히 캡쳐한 후 전달하면 비슷하게 나옴

Workflow from the beginning

- 생각 & 설계

- Plan 모드를 사용하거나, ChatGPT (Gpt-5 PRO, deep research, figma app) 을 사용
- 만들어진 설계도를 md 파일로 정의

- Plans

- task 들을 잘게 쪼개서 TODO list 로 관리
- 각 작업은 중간중간 git commit 으로 짧게 찍혀서 언제든 되돌릴 수 있어야 함
(Coding agent 는 실수를 많이 함... 사실 사람도 많이 함)

- Implementation

- 작은 Plan 들을 하나씩 수행
- 가능하다면
 - 매 TASK 는 Test 를 통과해야 하고,
 - Codex 가 아닌 다른 AI 들이 검수자로 검수하면 좋음

고민해볼 거리

- 어떤 언어?
 - Python – 학습 데이터 많음, 자유도 높음
 - Rust – 시스템에서 정보를 많이 제공
 - ...
- 어떤 모델? (너무 빨리 변함)
 - Claude – 코딩 잘함, *agentic* 지능 높음, 범용성 조금 부족
 - GPT5 – 설계 잘함, 지식 많음, 대신 codex 모델, deep research 모델이 있음
 - Gemini – 범용성 높음, 가벼운 일 잘함, 문서 잘 씀
 - ...
- 어떤 제품?
 - Cursor, Claude Code, Codex, Gemini, Lovable, Replit, ...
 - 너무 많고 특색이 있음. 근본에 가까운 도구 하나와 나에게 필요한 도구 하나 정도 조합 추천

고민해볼 거리

- AI에게 일을 잘 시킬 수 있는 방법
 - 최대한 많은 정보를 빠짐 없이 전달 하기
 - Agents.md, todo.md, architecture.md, ...
 - Python 문법 검사하기, 에러 정보 제공하기, 다른 AI에게 물어보기
 - = Context Engineering
 - 일을 최대한 잘 쪼개서 주기
 - 삭제/재작업이 사람에 비해 저렴하므로 언제든 되돌릴 수 있게
- 부하직원에게 일을 잘 시키는 것과 다르지 않다!
 - 정치/이해관계/감정이 없으니 더 쉬운가?

Case study

- AI에게 일을 많이 시키면 얼마나?
 - <https://www.youtube.com/watch?v=Og0mtAbDrDY>
- 개인이 하루에 3억 어치도 사용 가능함
- 클로드 코드에 주간 사용량이 도입된 이유
- 최대한 잘, 많이 사용하는 것이 중요하다

Claude Code Token Usage Report - Daily				
Date	Models	Input	Output	Cost (USD)
2025-08-20	- opus-4 - sonnet-4	992,266	13,248,656	\$90679.97
2025-08-30	- opus-4	2,602,358	3,364,540	\$71535.53
2025-09-01	- opus-4	3,674	50,790	\$222.51
2025-09-02	- opus-4	748,414	10,287,446	\$52810.97
2025-09-03	- opus-4	2,928,816	41,023,548	\$209182.69
2025-09-13	- sonnet-4	354	58,741	\$5.29
2025-09-15	- sonnet-4	29,357	1,454,879	\$581.28
2025-09-16	- sonnet-4	4,708	333,256	\$83.01
2025-09-17	- sonnet-4	1,838	139,140	\$36.35
Total		7,311,785	69,960,996	\$425137.59

Local LLM Serving with Ollama

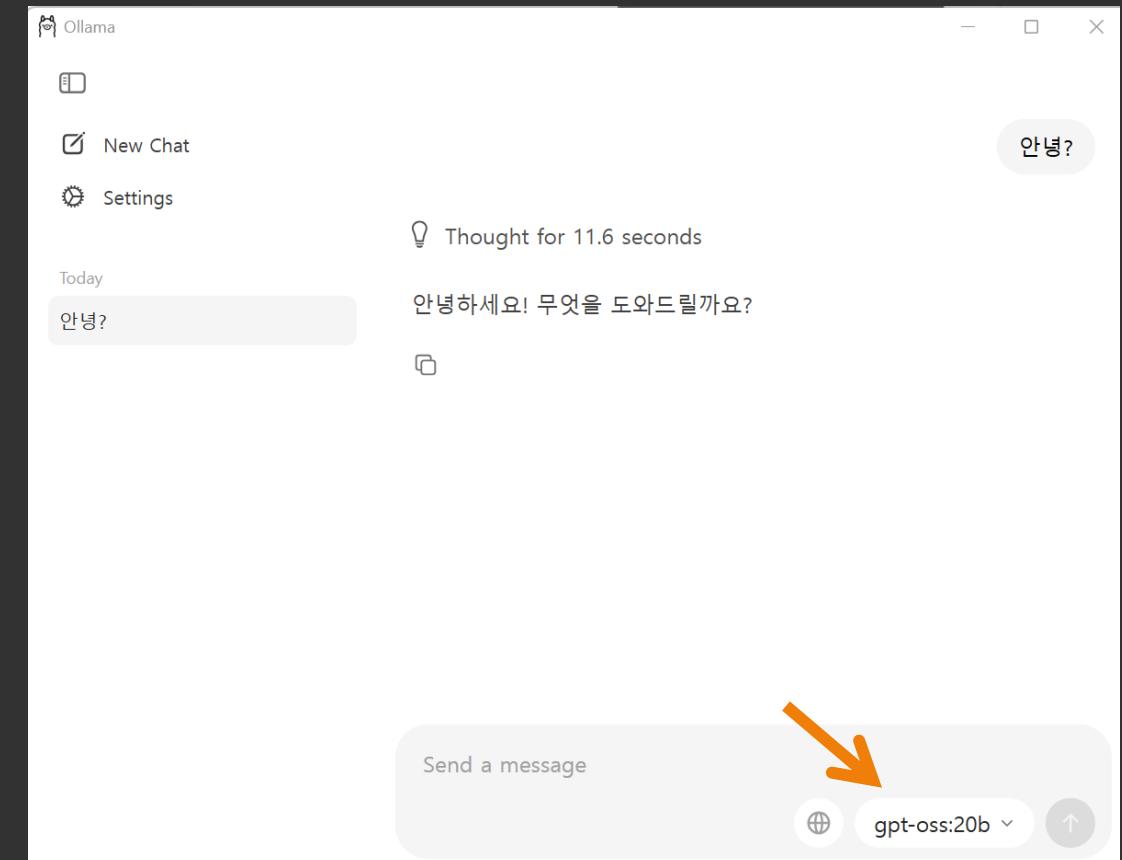
- **Ollama**

- <https://ollama.com/>
- 누구나 쉽게 사용하도록 하는 것을 추구, windows 도 지원
- 동시 사용이 느리다는 단점이 있음

- **LM Studio**

- **vLLM**

- **sglang**



Local LLM Serving with Ollama

○ Ollama 서버 테스트

```
#bash
$ curl http://localhost:11434/v1/chat/completions -H "Content-Type: application/json" -d
'{"model":"gpt-oss:20b", "messages":[{"role":"user", "content":"hello"}]}'

#powershell
irm http://localhost:11434/v1/chat/completions -Method Post -ContentType "application/json" -
Body '{"model":"gpt-oss:20b", "messages":[{"role":"user", "content":"hello"}]}'
```

○ CODEX 에서 연결 하기

- 현재 (25.10) 는 codex cli 에서만 가능함
- Windows 에서는 WSL 을 켜고 연결하기

Codex CLI with Ollama

- Codex cli 를 "-oss" 옵션을 주고 실행하면 자동으로 ollama 에 gpt-oss 셋업
- 기타 다른 모델을 사용하고 싶다면 config.toml 에 모델을 정의해주어야 함

```
> codex --oss
Downloading model: total 12.85 GB
success

>_ OpenAI Codex (v0.38.0)

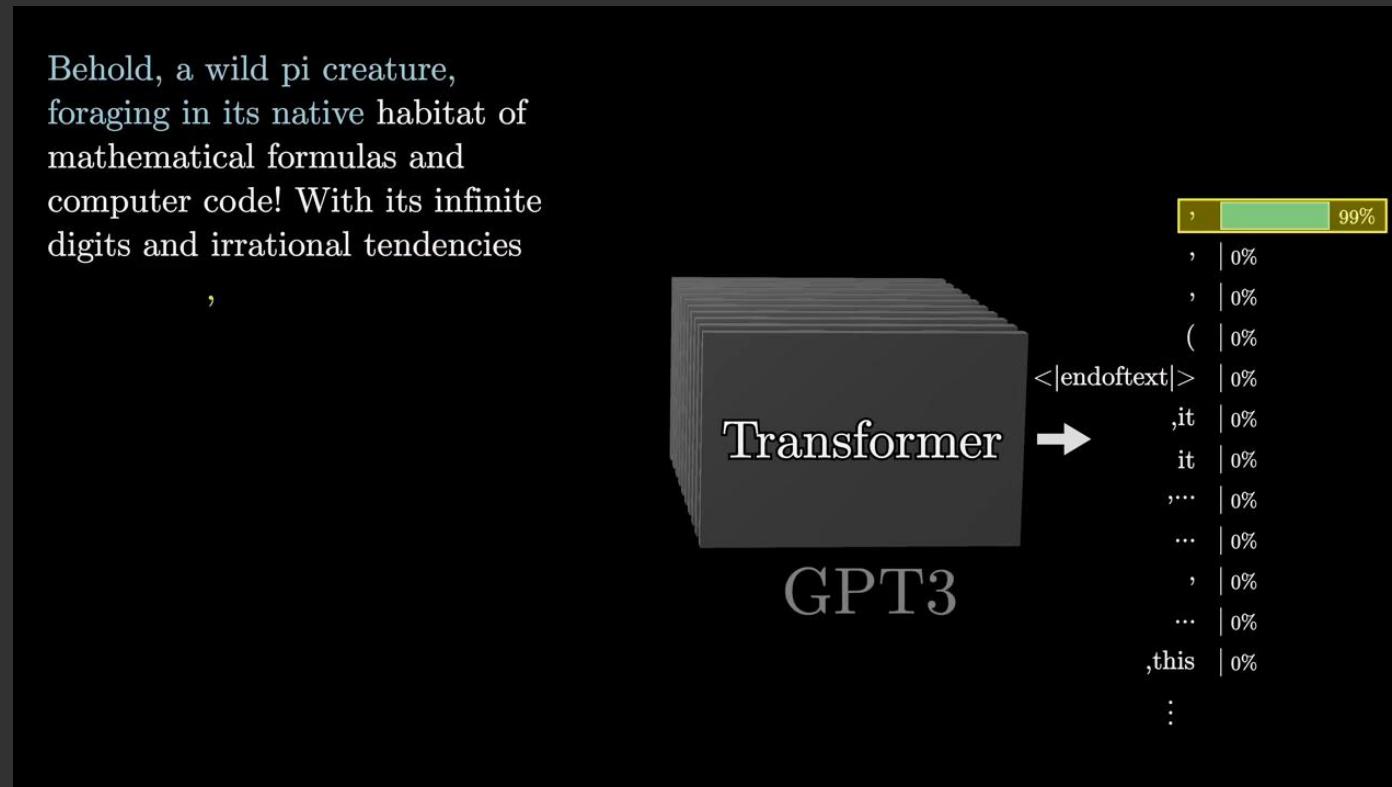
model:      gpt-oss:20b high    /model to change
directory: ~/Downloads
```

To get started, describe a task or try one of these commands:

/init – create an AGENTS.md file with instructions for Codex
/status – show current session configuration
/approvals – choose what Codex can do without approval
/model – choose what model and reasoning effort to use

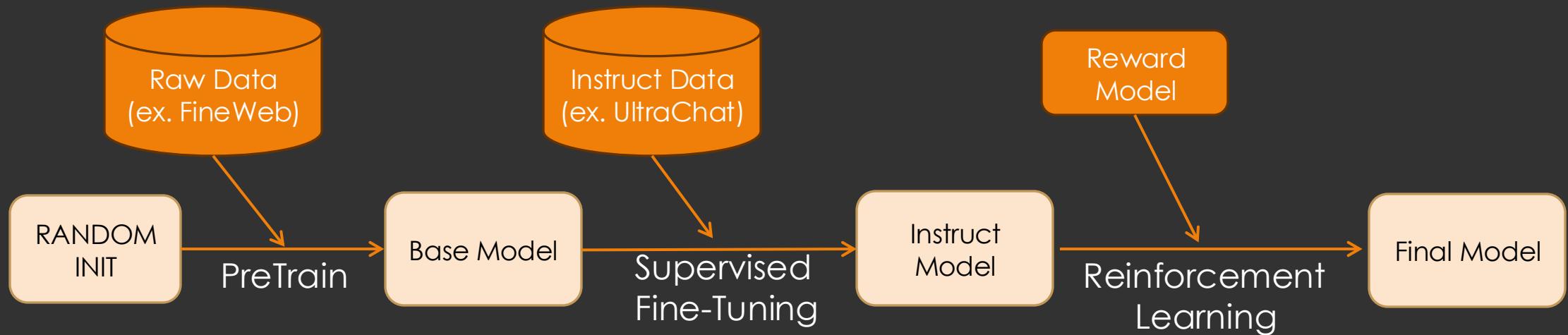
Problem Definition of LLMs

- 단순한 Next Token (Word) Prediction Model
- 사람의 Labeling 작업이 필요 없어지므로 **Training Data 의 무한 공급**이 가능
→ Scaling 을 통해 똑똑한 모델을 만드는 것이 가능함



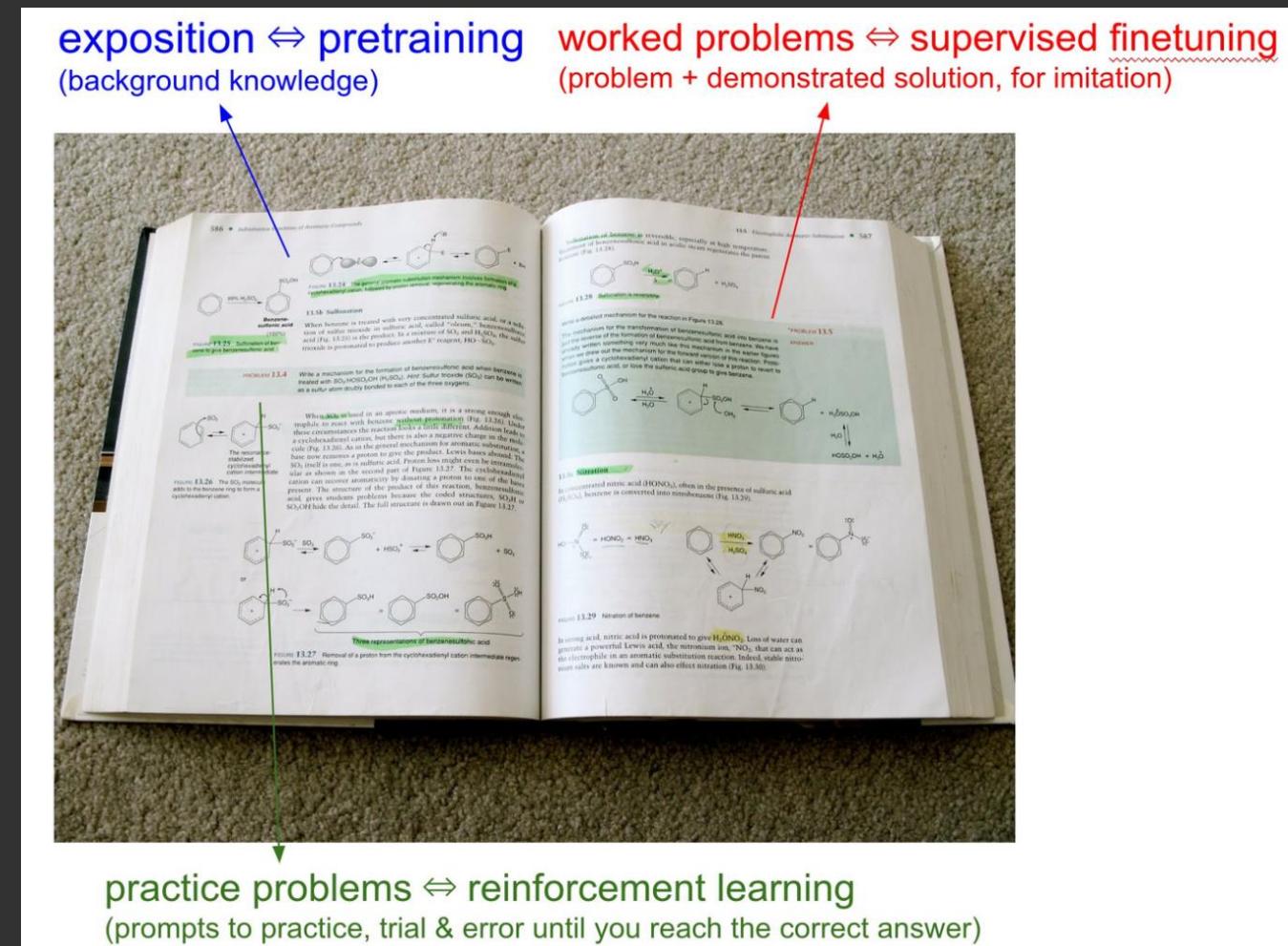
Building LLMs – The Recipe

- 1. Pre-Training
 - Base Model 이 생성됨
- 2. Supervised Fine Tuning (SFT) using Instruct Data
 - ChatGPT 스타일의 대화형 모델, Instruct 모델이 생성됨
- 3. Reinforcement Learning
 - 사람들이 좋아할만한 모델로 개선됨



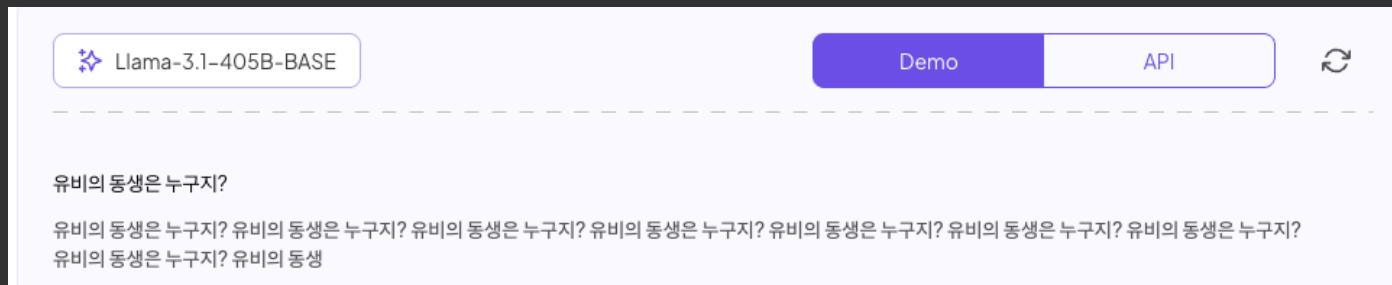
Building LLMs – The Recipe

- 1. Pre-Training
 - 책을 읽는 과정
- 2. Supervised Fine Tuning (SFT) using Instruct Data
 - 연습 문제를 푸는 과정
 - 기존의 고수가 작성한 해설지를 보고 학습
- 3. Reinforcement Learning
 - 문제를 직접 풀어보는 과정
 - 풀이 방법은 자유
 - 아무도 생각하지 못한 기발한 방법을 생각해 낼 수도 있음

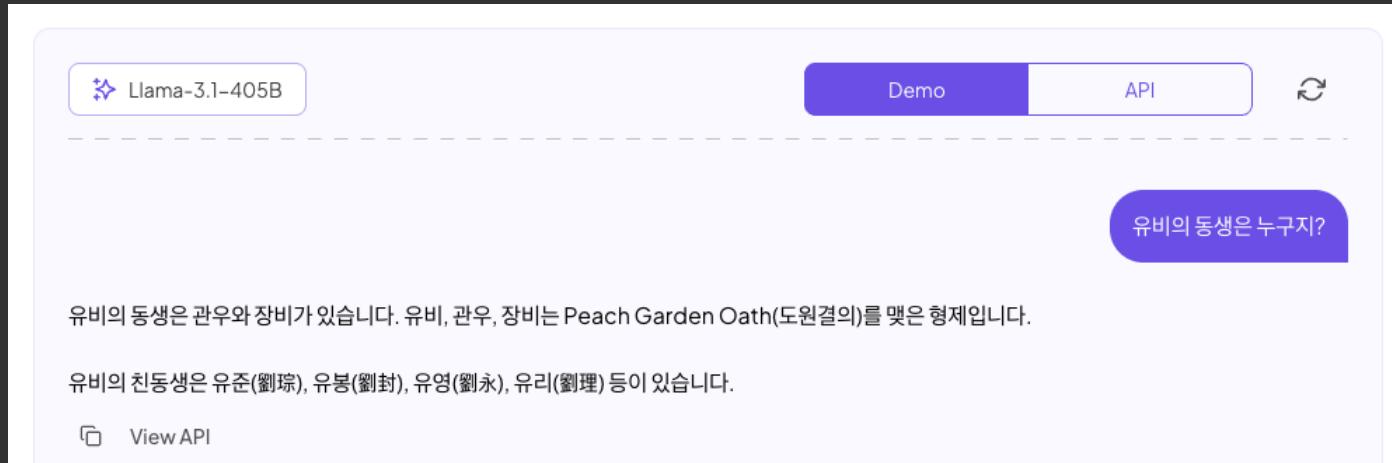


모델 분류 1. Base Model vs Instruct Model

- Base Model (= pretrained model)
 - Next Token Prediction 만 함, 대화 기능이 없음



- Instruct Model
 - 사람처럼 대화가 가능함
 - 현시점 기준 대부분의 LLM
 - + tool use 같은 기능들이 가능함

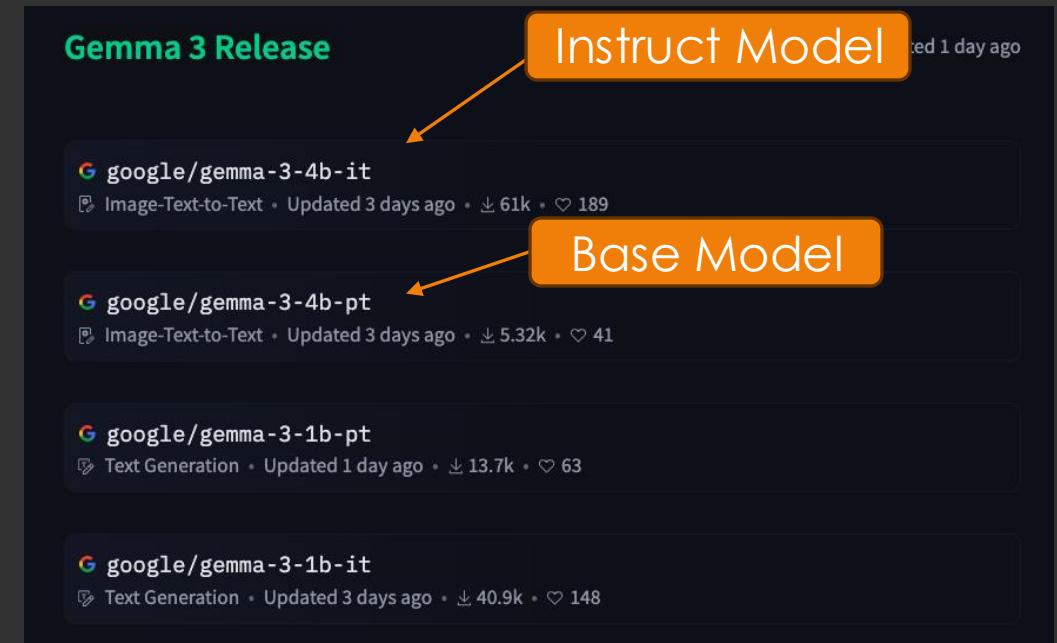


- Let's Test on HyperBolic
 - <https://hyperbolic.xyz/>

모델 분류 1. Base Model vs Instruct Model

- Base, Instruct 모델을 두 가지 다 제공하기도 함
 - 많은 모델들이 base 모델을 지원 안 하기도 합니다 (특히 한국... ㅠㅠ)
 - Base Model 이 있으면 보다 다양한 튜닝이 가능
- Instruct 모델은 보통 많은 기능을 추가함
 - Tool Use, Hallucination 방지, 기타 등등...

	Finetuned	Multilingual	Long context	Tool use	Release
Llama 3 8B	✗	✗ ¹	✗	✗	April 2024
Llama 3 8B Instruct	✓	✗	✗	✗	April 2024
Llama 3 70B	✗	✗ ¹	✗	✗	April 2024
Llama 3 70B Instruct	✓	✗	✗	✗	April 2024
Llama 3.1 8B	✗	✓	✓	✗	July 2024
Llama 3.1 8B Instruct	✓	✓	✓	✓	July 2024
Llama 3.1 70B	✗	✓	✓	✗	July 2024
Llama 3.1 70B Instruct	✓	✓	✓	✓	July 2024
Llama 3.1 405B	✗	✓	✓	✗	July 2024
Llama 3.1 405B Instruct	✓	✓	✓	✓	July 2024



Tokenizer

- 실제로 LLM 이 입력을 받는 단위 : Token
- Tiktoknizer - <https://tiktoknizer.vercel.app/>
- Instruct 모델의 Chat 형식 스페셜 토큰들 확인 가능

- GPT의 tokenizer 기준

<|im_start|>, <|im_sep|>, <|im_sep|>

해당 토큰들로 chat 형식 구현

The screenshot shows the Tiktoknizer interface with the following details:

- Model Selection:** gpt-4o
- System Message:** You are a helpful assistant
- User Message:** hello!
- Token count:** 18
- Message Log:** The log shows the tokens for the system and user messages, separated by <|im_sep|> tokens. The tokens are color-coded: yellow for <|im_start|>, green for <|im_end|>, blue for <|im_sep|>, red for <|im_start|>, purple for <|im_sep|>, and pink for <|im_start|>assistant<|im_sep|>.
- Token IDs:** A list of 18 token IDs: 200264, 17360, 200266, 3575, 553, 261, 10297, 29186, 200265, 200264, 1428, 200266, 24912, 0, 200265, 200264, 173781, 200266.
- Show whitespace:** A checkbox labeled "Show whitespace" is checked.

Tokenizer 가 야기하는 문제

- How Many 'r' s in Strawberry?

왜 이 간단한 질문에 대해 답을 하지 못할까?

- Strawberry 가 하나의 토큰 이기 때문
- Alphabet 단위의 tokenizer 를 사용한다면,
아주 쉽게 대답할 것!

```
<|im_start|>system<|im_sep|>You are a helpful assistan  
t<|im_end|><|im_start|>user<|im_sep|>How Many 'r's in  
strawberry?<|im_end|><|im_start|>assistant<|im_sep|>
```

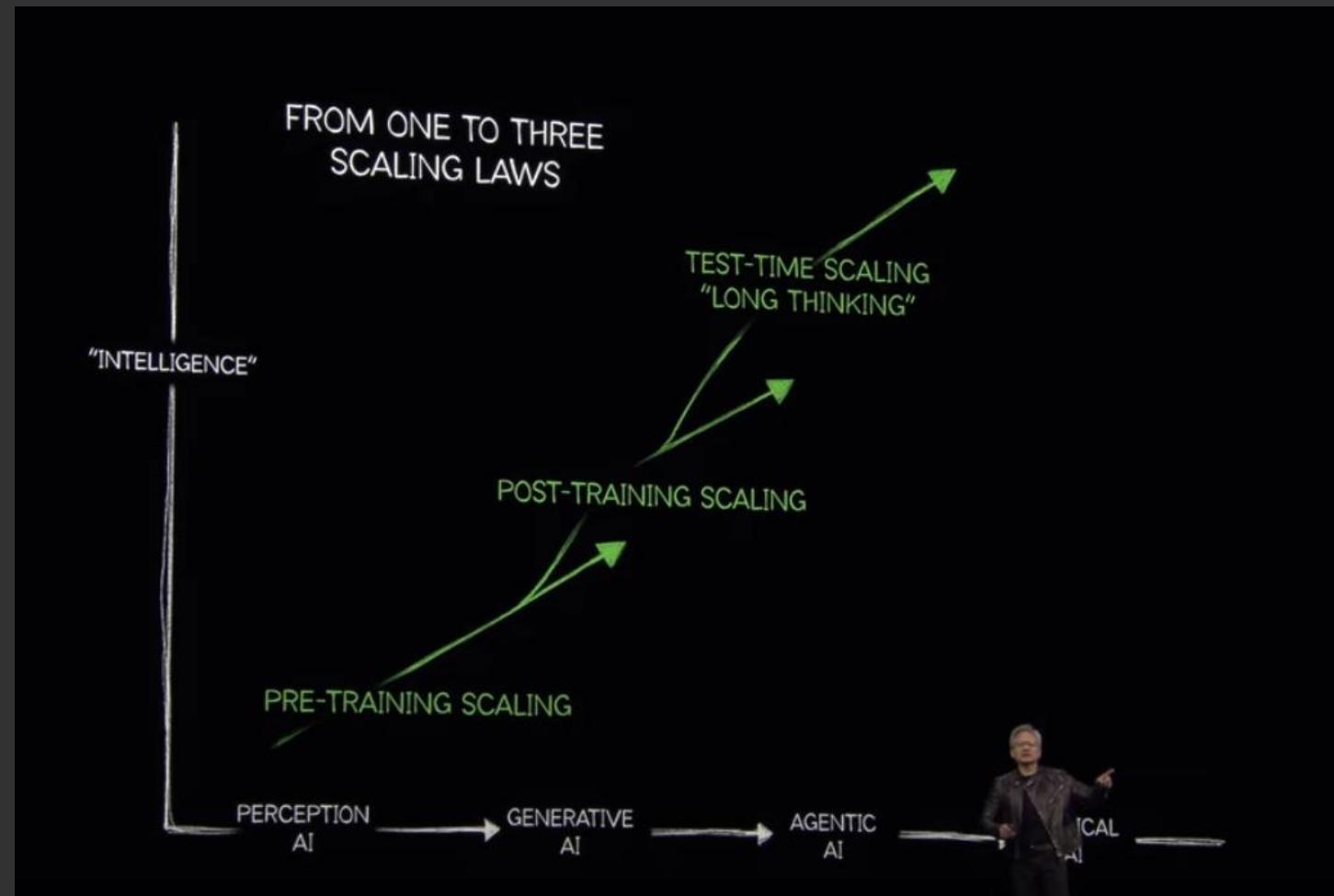
```
200264, 17360, 200266, 3575, 553, 261, 10297, 29186, 2  
00265, 200264, 1428, 200266, 5299, 11048, 461, 81, 88  
5, 306, 101830, 30, 200265, 200264, 173781, 200266
```

모델 분류 2. 용도에 따른 분류

- Embedding Model
 - Text 를 vector로 바꿔주는 모델, search 나 clustering 으로 사용, RAG 에 필수적인 모델
- Multi-Modal Model
 - Text 에 Image 나 Audio 와 같은 다른 형태의 Input/Output 이 추가된 모델
 - 24년 Gemini, GPT-4o 이후로 대부분 Multi Modal In/Out 을 지원하
 - LLM 에 Vision Encoder 를 추가하고, Audio 는 텍스트랑 호환되도록 토큰을 추가하는 방식이 일반적
- Reasoning Model
 - Test Time Scaling 을 통해 보다 똑똑한 대답을 하는 모델
 - OpenAI o1, Deepseek R1, Gemini Thinking, ...
- 기타 특수한 모델
 - dLLM (Diffusion LLM)
 - VLA (Vision-Language Action) Model
 - Coder Model
 - ...

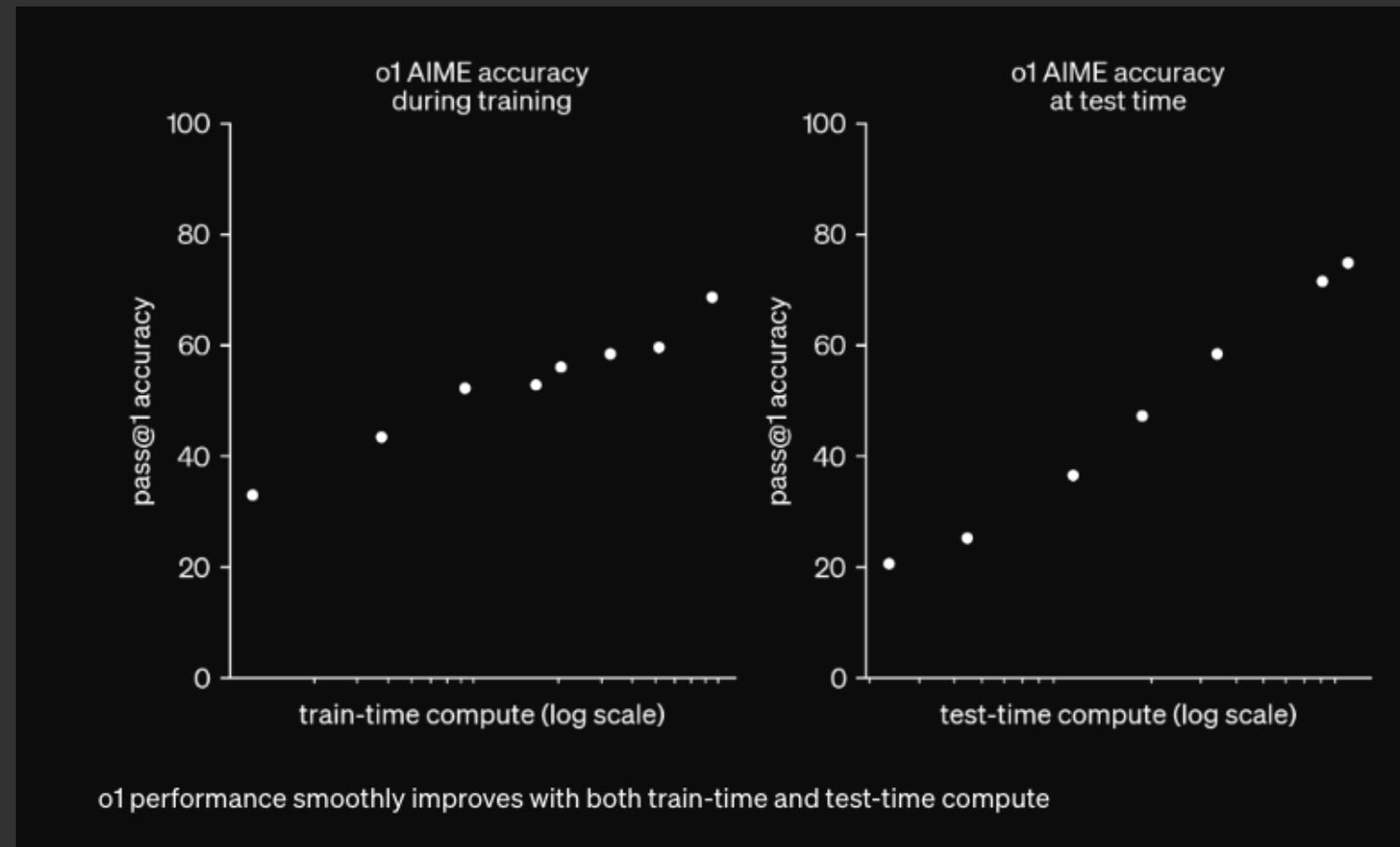
Test-Time Scaling

- OpenAI o1 & DeepSeek R1 과 같은 Reasoning 모델
- Pre Training Scaling
 - 사전 학습을 많이!
- Post Training Scaling
 - 양질의 Instruct Data
 - RLHF
- Test-Time Scaling
 - 생각을 오래하기



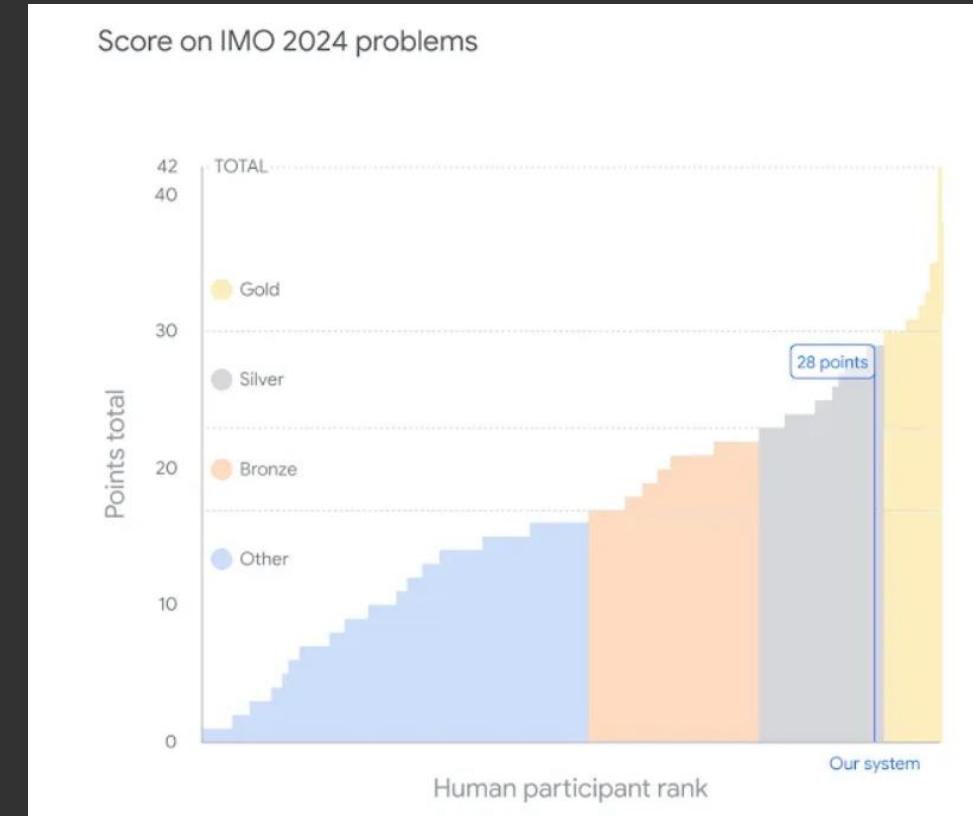
Train Time & Compute Time

- OpenAI o1
- Learning To Reason (Reinforcement Learning)
 - <https://openai.com/index/learning-to-reason-with-langs/>
 - Talk by Noam (OpenAI) 24.05
- 학습만 오래하는 것이 방법이 아니다
- 고민을 오래하는 것도 중요
 - 특히 수학, 코딩 등 고민이 필요한 문제



Case Study - AlphaProof by GDM (2024)

- AlphaProof, AlphaGeometry by Google DeepMind
 - <https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/>
- 2024.07, 인간 최고수 수준의 수학 능력 달성
- Our systems solved one problem within minutes and took up to three days to solve the others.
 - 원 대회 규정은 4.5시간씩 2일
 - Test-Time Scaling 이 큰 기여를 한 것으로 추정
 - Using LEAN (Verifiable Math Representation)
- (참고) 조합 문제는 0점, 나머지 대수/정수론/기하는 만점



Case Study – 2025 IMO

- 2025.07, 올해는 IMO 금메달
- OpenAI Internal Model
 - https://x.com/alexwei_/status/1946477747615989912
 - <https://www.youtube.com/watch?v=EEIPtofVe2Q&list=PLOhHNjZItNnMm5tdW61JpnyxeYH5NDDx8>
 - 2개월 3명, 모르면 모른다고 함
- Google Gemini 2.5 pro DeepThink
 - <https://deepmind.google/discover/blog/advanced-version-of-gemini-with-deep-think-officially-achieves-gold-medal-standard-at-the-international-mathematical-olympiad/>
- 시간 제한 동일, No internet, No Tool Call, 6 문제 중 5문제 맞춤
- RLVR 인가? Natural Language 의 수학 풀이는 Verifiable 하지 않다... (LEAN 을 쓰는 이유)
 - 그렇다면 어떻게? 새로운 방법이 있다고 의심하는 smolAI Founder, 단순 스케일링 이라는 Hyperbolic Founder
 - <https://www.youtube.com/watch?v=xIXIi6GZNGY>

TTS 에 대한 들어가기 앞서서

- "Model need time to think" by OpenAI Prompting Guide
- 아래 3가지의 답변 방식 중, **가장 좋은 (정답을 말할 확률이 높은) 답변은 무엇일까요?**

User: 에밀리는 23개의 사과랑 177개의 오렌지를 구입함. 오렌지는 개당 4달러임. 총 869달러를 지불했음. 사과의 개당 가격은?

사과는 7달러 입니다

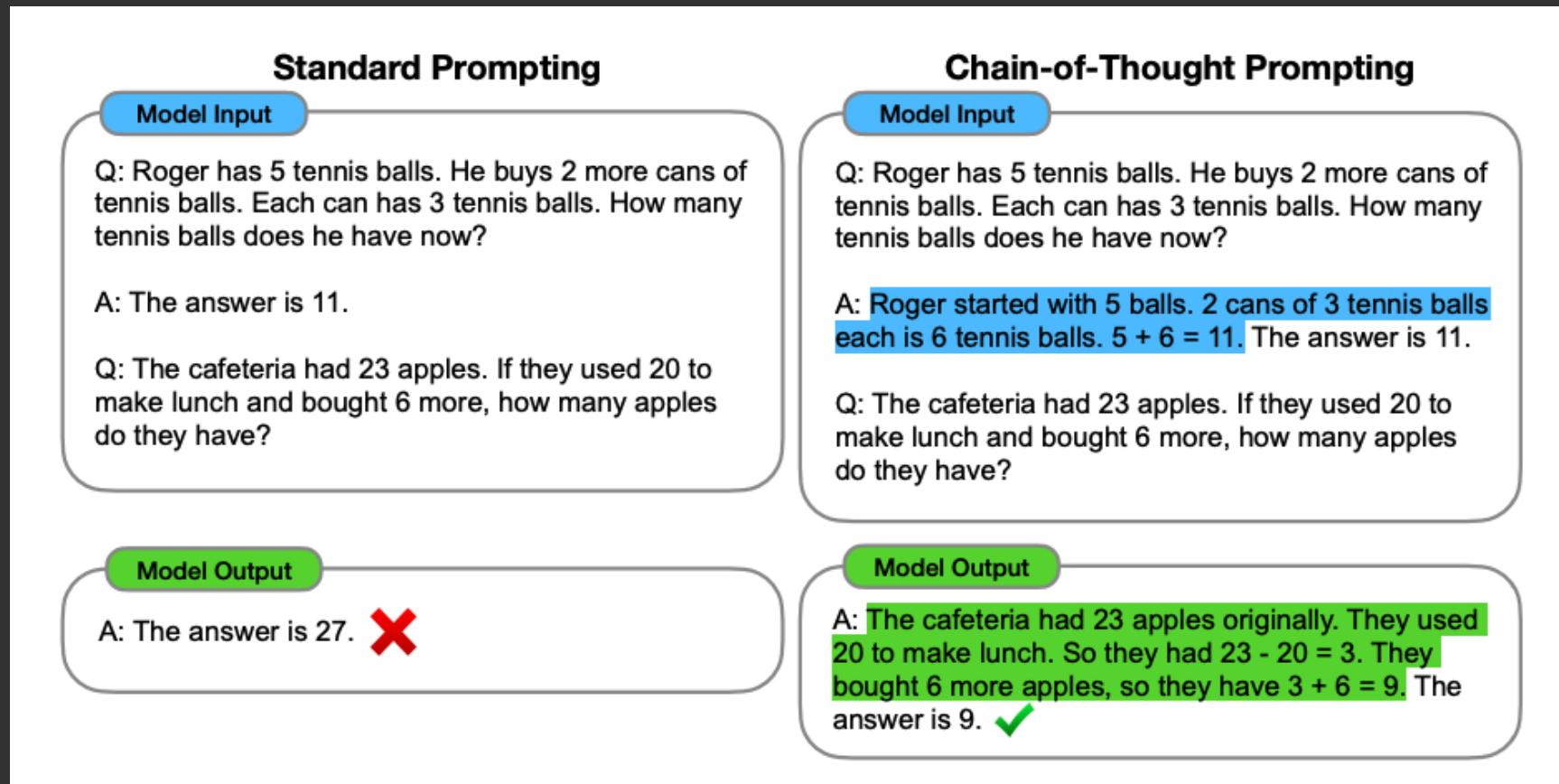
사과는 7달러 입니다. 왜냐하면, 오렌지 가격은 4달러이고 에밀리가 구매한 오렌지는 177개이고, ...

오렌지 가격은 4달러이고 에밀리가 구매한 오렌지는 177개이고, ...
그래서, 사과는 7달러 입니다.

- 정답을 도출하기까지의 연산량이 다름
- 사람도 풀이를 써야 어려운 문제를 풀 수 있음
- 두괄식을 원하면, 풀이&답변을 다시 재작성하면 됨

Chain of Thought Prompting

- Test Time Scaling 의 초기 접근 방법 - Chain of Thought Prompting
- 프롬프팅으로 TTS 를 유도, 당시에는 TTS 라는 용어는 없었음
- 가장 대중적이고 구현이 쉬운 방법, Reasoning 모델에는 잘 사용하지 않음



OpenAI o1

- OpenAI 가 공개한 첫 번째 Reasoning 모델
- ChatGPT 에서 사용하면 Reasoning 과정은 요약해서 보여줌
- API 사용하면 Reasoning 토큰은 숨김 처리 (가격을 차징됨)

While reasoning tokens are not visible via the API, they still occupy space in the model's context window and are billed as output tokens.

ChatGPT o1 pro ▾

공유하기 박 세부 정보

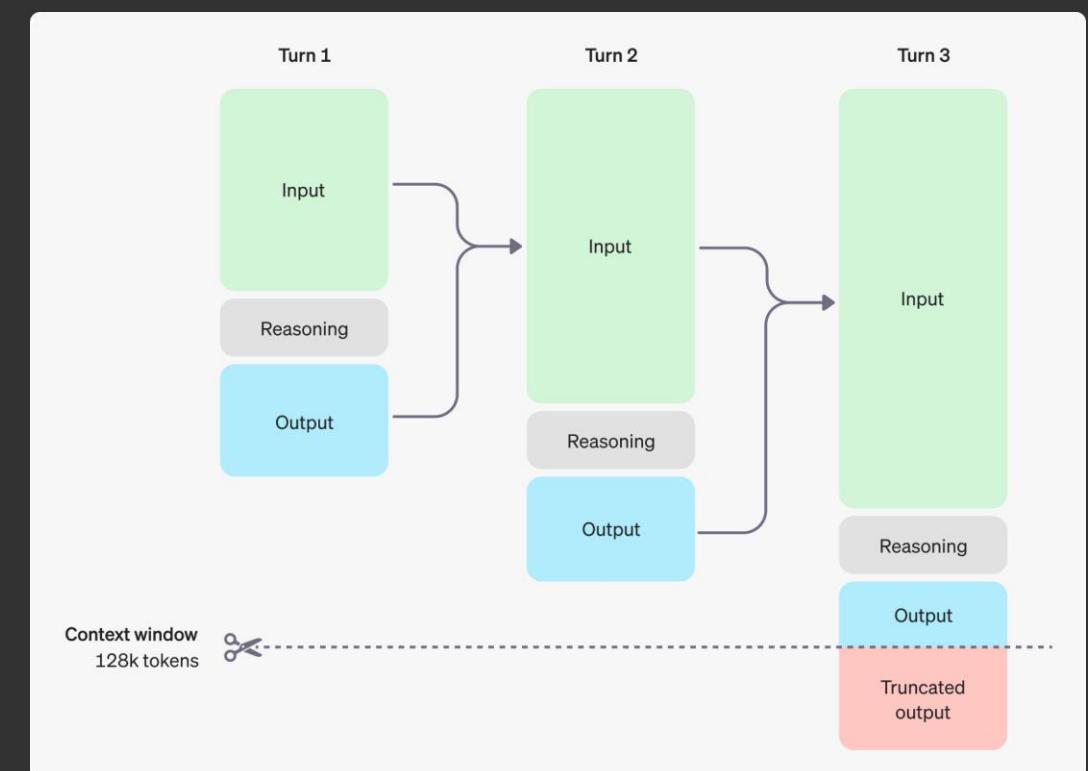
reasoning 모델에 대해서 설명해줘

추론 모델 설명
2m 44s 동안 이상 처리 완료

Reasoning 모델이란 인공지능(Artificial Intelligence) 분야에서 **추론(Reasoning)**을 수행하도록 설계된 모델들을 통칭하는 개념입니다. 즉, 입력된 정보(데이터, 문장, 상황 등)에 대한 해석을 바탕으로 결론을 도출하거나 의사결정을 할 수 있는 모델을 말합니다. 전통적으로는 기호적(Symbolic) 접근법과 확률적(Probabilistic)-통계적 접근법, 그리고 최근에는 대규모 신경망(딥러닝 기반)을 활용한 접근법이 혼합적으로 쓰이고 있습니다.

1. 기호적(Symbolic) Reasoning 모델

- 특징
 - 논리 규칙, 자식 그래프(knowledge graph) 혹은 온톨로지(ontology) 등을 활용해 체계적으로 추론 과정을 정의함.
 - 예: 전문가 시스템(Expert System), 논리 프로그래밍, 규칙기반 시스템 등.



Case Study – DeepSeek R1

- Reasoning 모델의 레시피를 모두에게 공개한 첫 번째 모델
- 2025년 1월 출시, OpenAI o1 수준의 모델
- OpenAI 의 Chief Research Officer 가 공개적으로 코어 아이디어가 같다고 이야기 함

Mark Chen

@markchen90

Congrats to DeepSeek on producing an o1-level reasoning model! Their research paper demonstrates that they've independently found some of the core ideas that we did on our way to o1.

3:11 AM · Jan 29, 2025 · 8.2M Views

- V3 의 학습 비용과 R1 의 성능은 실제 미국 주식 시장에 영향을 줄 정도의 영향력

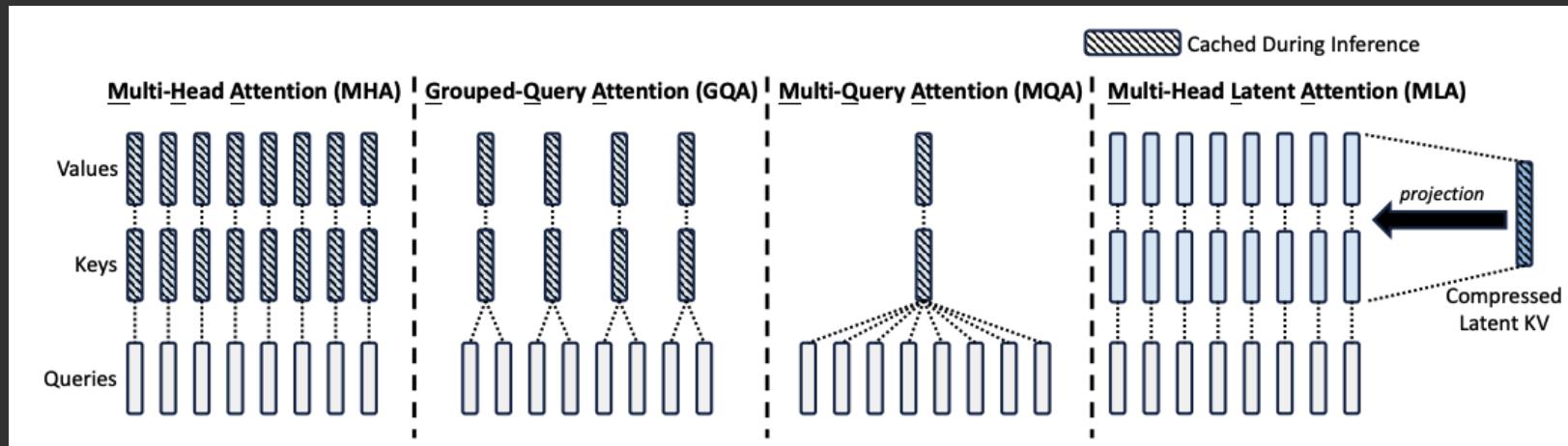
Case Study – DeepSeek V3



- 2024.12 월 발표
- Base & Chat Model of DeepSeek - <https://api-docs.deepseek.com/news/news1226>
- 671B MOE (Mixture of Experts) 구조, 37B 만 활성화
- GPT-4o 수준 (일각에서는 Llama4 가 V3보다 안 좋아서 출시를 미룬다는 소문이 있을 정도)
- OpenAI 대비 50배 저렴한 API 비용으로 서빙
 - 학습 비용 또한 100억원 안쪽으로 달성했다고 주장 (H800 x2000 을 이용함)
- 꽤나 오픈소스, Open Weight 모델
 - 본인들이 만든 도구들을 오픈소스로 많이 풀었음, 학습 프레임워크 뿐 아니라 파일 시스템도 포함될만큼 많음
 - 가장 핵심인 “학습 데이터”는 공개하지 않아, 재현은 불가능
- DeepSeek V3 0324 , 후속 버전도 출시

Case Study – DeepSeek V3

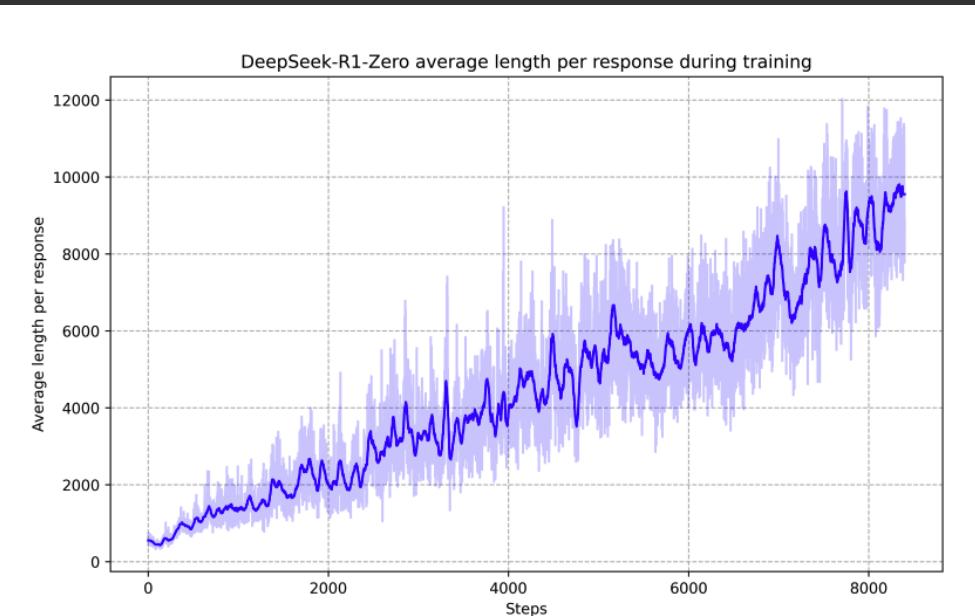
- DeepSeek Model's Key Innovative Techniques - <https://arxiv.org/pdf/2503.11486>
- Multi-head Latent Attention
 - LLAMA 에 적용된 Grouped Query Attention 이 가장 대중적이었음
 - Latent Attention 을 도입하여, 압축된 벡터 공간에서도 attention 이 가능함을 증명



- Mixture of Experts, FP8 학습, Multi-Token Prediction 등 여러 기법들을 성공적으로 도입함

Case Study – DeepSeek R1

- GRPO (Group Relative Policy Optimization) 사용
 - DeepSeek V3-Base에서 RL을 이용하여 DeepSeek R1-Zero를 생성
- 수학처럼 정답이 있는 문제 → Rule Based Reward
 - 풀이를 자유도 있게 하고 정답률만 고려하여 GRPO
 - 평균 대답길이가 점점 길어짐
 - Aha moment가 관찰이 되기 시작함



Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a+x}} = x$ is equal to

Response: <think>

To solve the equation $\sqrt{a - \sqrt{a+x}} = x$, let's start by squaring both ...

$$(\sqrt{a - \sqrt{a+x}})^2 = x^2 \implies a - \sqrt{a+x} = x^2.$$

Rearrange to isolate the inner square root term:

$$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$$

...

Wait, wait. Wait. That's an aha moment I can flag here.

Let's reevaluate this step-by-step to identify if the correct sum can be ...

We started with the equation:

$$\sqrt{a - \sqrt{a+x}} = x$$

First, let's square both sides:

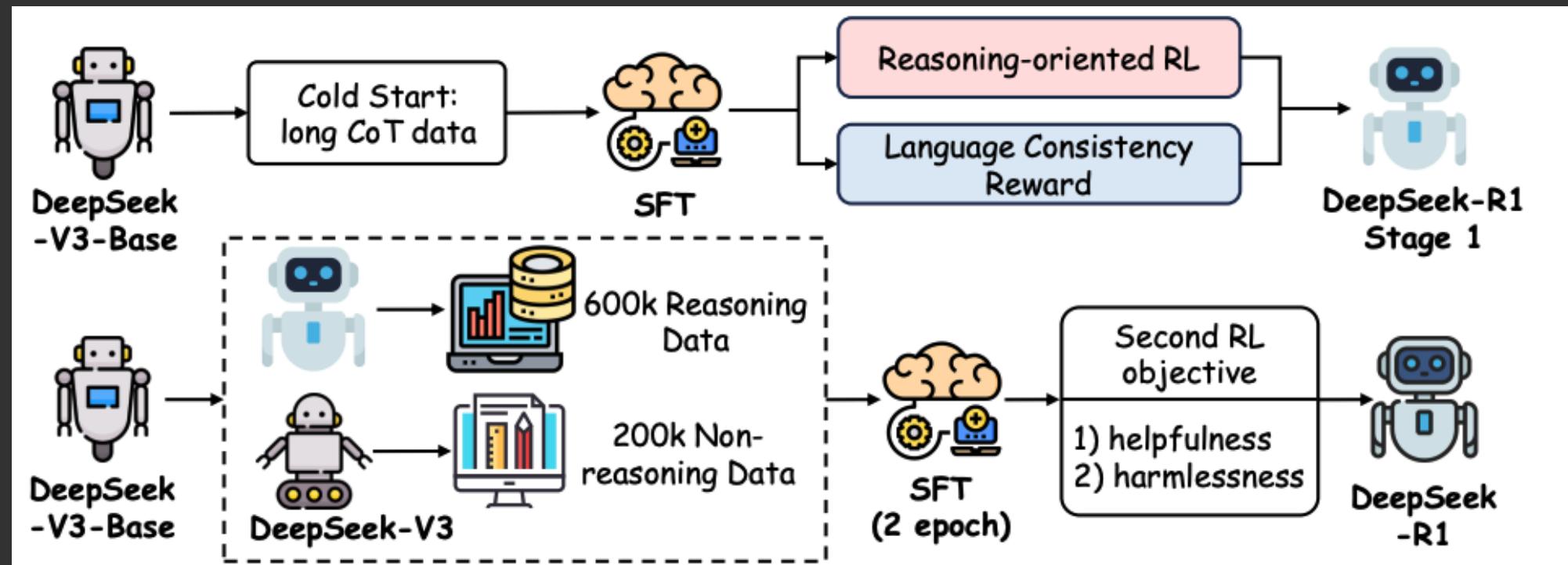
$$a - \sqrt{a+x} = x^2 \implies \sqrt{a+x} = a - x^2$$

Next, I could square both sides again, treating the equation: ...

...

Case Study – DeepSeek R1

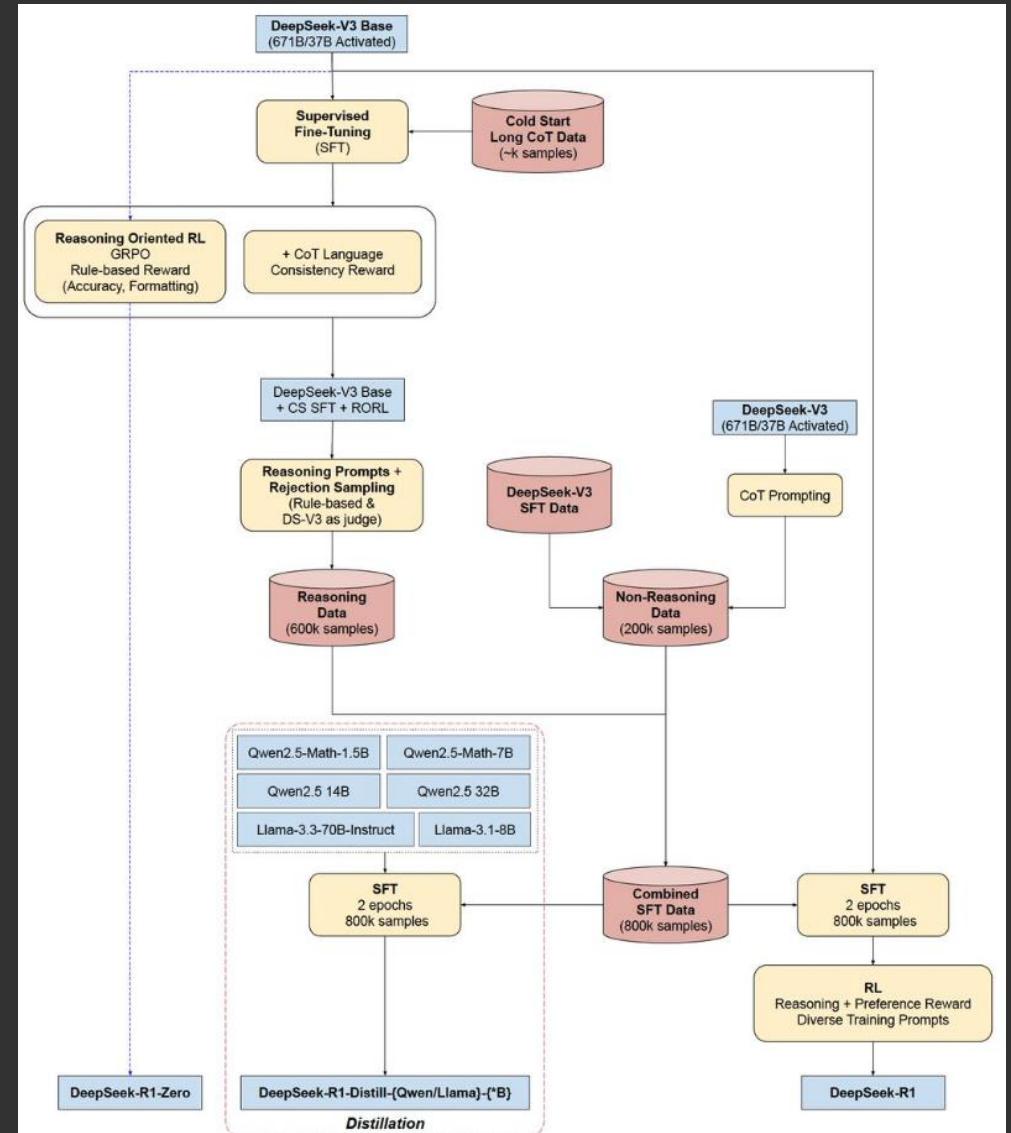
- DeepSeek R1 Zero 로 생성된 Reasoning Data = Cold Start, long CoT Data
- Reasoning Data 로 DeepSeek V3 Base를 학습 → DeepSeek-R1 Stage 1
 - 앞서 사용한 Zero 를 만들 때 사용한 RL (GRPO)
 - 답변에 일관성을 체크하여 reward 부여



Case Study – DeepSeek R1

- Stage1 모델에 Reasoning Prompting + Rejection Sample
 - Rule base 또는 V3에게 LLM as Judge
 - 양질의 Reasoning 데이터 확보
- Non – Reasoning 데이터도 수집
 - 원래 V3 Base → V3 만들 때 사용한 SFT 데이터
 - V3에서 프롬프팅으로 추출한 데이터
- Reasoning + Non-Reasoning 양질의 데이터 확보
- V3 Base에서 위 데이터로 학습 & Preference 학습 추가
- 양질의 데이터를 다른 모델에 학습 시키면 R1 Distill 모델
 - 다양한 R1 Distill 모델을 만들어서 배포함

<https://x.com/SirrahChan/status/1881488738473357753>



Case Study – DeepSeek R1

- **DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning**
 - <https://arxiv.org/abs/2501.12948>
- QwQ 32B Preview 를 기반으로 RL 과 Distillation 테스트
 - QwQ 32B Preview 는 Alibaba Qwen 의 reasoning 모델, 현재는 정식 출시
 - Distillation >>> 원본 모델 > RL 모델

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCodeBench
	pass@1	cons@64	pass@1	pass@1	pass@1
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9
DeepSeek-R1-Zero-Qwen-32B	47.0	60.0	91.6	55.0	40.2
DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2

Table 6 | Comparison of distilled and RL Models on Reasoning-Related Benchmarks.

- 시사하는 점
 - 양질의 데이터는 아주 큰 효과를 가져다 준다
 - RL 의 효과를 보기 위해서는 **시작점의 지능이 충분히 좋아야 한다**

HuggingFace



- Model / Dataset 저장소 - AI 분야의 github
 - git으로 버전 관리
 - Models Hub, 사실상 모든 회사가 Weight 공개를 Hugging Face 를 이용해서 함
 - Datasets, 데이터도 hub 처럼 활용
- Library
 - [Transformers](#)! Diffusers! Datasets! ...
 - PyTorch 기반의 파이썬 라이브러리, 가장 대중적으로 많이 사용됨
- Spaces
 - Application 을 호스팅 함, 유료 결제시 GPU 도 사용 가능
 - Gradio / Streamlit 기반
- Posts
 - 블로그 형식, 양질의 연구 또는 개발 관련 자료가 업로드 됨

HuggingFace - Models

- 모델 탐색 - <https://huggingface.co/models>
- 모델들은 Multi-tagged
 - Task, Library, Dataset, License, Language 등 다양한 기준으로 검색

The screenshot shows the Hugging Face Models page. On the left, there's a sidebar with tabs for Tasks, Libraries, Datasets, Languages, Licenses, and Other. Under Tasks, there are sections for Multimodal (Audio-Text-to-Text, Image-Text-to-Text, Visual Question Answering, Document Question Answering, Video-Text-to-Text, Visual Document Retrieval, Any-to-Any) and Computer Vision (Depth Estimation, Image Classification, Object Detection, Image Segmentation, Text-to-Image, Image-to-Text, Image-to-Image, Image-to-Video, Unconditional Image Generation). The main area displays a list of 1,529,193 models. At the top of this list, there's a 'Filter by name' input field. To the right of the input field, there are navigation tabs for Models, Datasets, Spaces, Posts, Docs, Enterprise, and Pricing. Below these tabs, there are buttons for 'Full-text search' and 'Sort: Trending'. Three red arrows point to these elements: one to the 'Tasks' tab, one to the 'Models' tab, and one to the 'Sort: Trending' button.

Models 1,529,193

Filter by name

Full-text search

Sort: Trending

Models

1,529,193

Filter by name

mistralai/Mistral-Small-3.1-24B-Instruct-2503

Image-Text-to-Text • Updated about 16 hours ago • ↓ 57.2k • 856

sesame/csm-1b

Text-to-Speech • Updated 7 days ago • ↓ 25.1k • 1.48k

canopylabs/orpheus-3b-0.1-ft

Text-to-Speech • Updated 3 days ago • ↓ 16.9k • 310

tencent/Hunyuan3D-2mv

Image-to-3D • Updated 4 days ago • ↓ 4.22k • 291

ByteDance/InfiniteYou

Text-to-Image • Updated about 20 hours ago • 237

ds4sd/SmolDocling-256M-preview

Image-Text-to-Text • Updated 3 days ago • ↓ 23k • 751

manycore-research/SpatialLM-Llama-1B

Text Generation • Updated 2 days ago • ↓ 1.28k • 495

google/gemma-3-27b-it

Image-Text-to-Text • Updated 1 day ago • ↓ 533k • 927

LGAI-EXAONE/EXAONE-Deep-32B

Text Generation • Updated 4 days ago • ↓ 3.7k • 249

Qwen/QwQ-32B

Text Generation • Updated 12 days ago • ↓ 551k • 2.46k

HuggingFace - Models

- Github 처럼 Model Card (Readme), Files
 - 당연히 다운받아서 사용 가능

```
from transformers import AutoModelForCausalLM, AutoTokenizer  
  
model_name = "Qwen/QwQ-32B"  
  
model = AutoModelForCausalLM.from_pretrained(  
    model_name,  
    torch_dtype="auto",  
    device_map="auto"  
)  
  
...  
  
generated_ids = model.generate(  
    **model_inputs,  
    max_new_tokens=32768  
)  
  
...  
  
response = tokenizer.batch_decode(generated_ids, skip_special_tokens=True)[0]  
print(response)
```

- Model 학습 경로 추적
 - Base Model, Finetunes, Quantizations, ...
- Inference Test 가 바로 가능 (셋업한 경우)

Qwen/QwQ-32B

Model card

Introduction

Inference Providers

Downloads last month: 551,168

Model tree for Qwen/QwQ-32B

Base model

- Finetuned (48)
- Adapters
- Finetunes
- Merges
- Quantizations

This repo contains the QwQ 32B model, which has the following features:

- Type: Causal Language Models

Benchmark	QwQ-32B	DeepSeek-R1-0718	OpenAI-o1-mini	DeepSeek-R1-Distil-Uomo-708	DeepSeek-R1-Distil-Qwen-32B
AIME24	79.3	79.6	71.6	77.6	71.4
LiveCodeBench	64.8	65.9	57.8	72.1	67.3
LiveBench	73.1	71.4	69.3	74.0	70.0
IIEval	82.0	81.8	84.8	79.2	79.8
BFCL	64.4	63.2	62.8	63.5	63.5

HuggingFace FineWeb – Dataset

- FineWeb Dataset Page - <https://huggingface.co/datasets/HuggingFaceFW/fineweb>
- 51.3 TB, 24,989,712,068 Rows, 15T tokens
- Data Studio에서 열람
- 이 Datasets을 활용해서 만들어진 모델 추적
- Datasets library와 연동
 - Lazy 처리 등 유용한 기능 구현 완료

```
from datasets import load_dataset

# use name="sample-10BT" to use the 10BT sample
fw = load_dataset("HuggingFaceFW/fineweb", name="CC-MAIN-2024-10", split="train", streaming=True)
```

Datasets: HuggingFaceFW/fineweb 2.05k Follow HuggingFaceFW 461

Tasks: Text Generation Modalities: Tabular Text Formats: parquet Languages: English Size: 10B - 100B

ArXiv: arxiv:2306.01116 arxiv:2109.07445 arxiv:2406.17557 DOI: doi:10.5796/hf/2493 Libraries: Datasets Dask Croissant +1 License: odc-by

Dataset card Data Studio Files and versions Community 57

Subset (108) default Split (1) train

Downloads last month 274,879

Use this dataset Edit dataset card

Homepage and Repository: huggingface.co

Point of Contact: please create a discussion on the Community tab

License: Open Data Commons Attribution License (ODC-By) v1.0

Size of downloaded dataset files: 51.3 TB

Size of the auto-converted Parquet files: 24,989,712,068

Models trained or fine-tuned on HuggingFaceFW...

- arnir0/Tiny-LLM
- sapienzanlp/Minerva-7B-base-v1.0
- deepvk/RuModernBERT-base
- sergeyzh/BERTA
- mraedermacher/WORLD_ARCHIVE-II-GGUF

FineWeb

- (참고) HF FineWeb
 - Pre-train 용 데이터셋
 - Space 참조 - <https://huggingface.co/spaces/HuggingFaceFW/blogpost-fineweb-v1>

실습 – Serving with vllm

- vllm 을 이용해서 만들어진 모델 serving
 - <https://github.com/vllm-project/vllm>
 - FastAPI 기반 OpenAI API Compatible 한 서버 생성

```
$ pip install vllm  
$ vllm serve jonhpark/llama3.1-8b-kowiki-instruct-16bit  
--max-model-len 2048 --chat-template ./template.jinja
```

- Huggingface 에서 업로드한 모델 로드 (로컬 파일도 가능)
- Max Model length, OOM 피하기 위해 길이 제한
 - 하드웨어 사양 / 유저 요구사항 따라 다르게 설정
- Chat Template
 - OpenAI 스펙 & Instruct Tuned 매칭 (TBD)

The screenshot shows the FastAPI documentation interface. At the top, it displays the title "FastAPI 0.1.0 OAS 3.1" and a link to "/openapi.json". Below this, there's a section titled "default" containing a list of API endpoints:

- GET /health Health
- GET /load Get Server Load Metrics
- GET /ping Ping
- POST /ping Ping
- POST /tokenize Tokenize
- POST /detokenize Detokenize
- GET /v1/models Show Available Models
- GET /version Show Version

Below this, there is a detailed view of the POST /v1/chat/completions endpoint:

- Parameters: No parameters
- Request body (required): application/json
- Example request body (JSON):

```
{  
    "messages": [  
        {  
            "content": "체첸 공화국에 대해서 설명해",  
            "role": "user"  
        }  
    ]  
}
```

- NVIDIA Triton 이 자사 제품의 경쟁자를 내장 할 정도 인가?
- LMSYS Chatbot Arena 도 vLLM 이 호스팅
- LambdaLabs, SkyPillot, Runpod, ... 관련 도구들 모두 vLLM integrate 안내가 있음

The screenshot shows a Lambda Docs page titled "How to serve the Llama 3.1 8B and 70B models using Lambda Cloud on-demand instances". The page includes a sidebar with navigation links like "ON-DEMAND CLOUD", "Cloud API", "File systems", etc. The main content area contains a heading, a call-to-action button ("Apply for Cloud Credits"), and a detailed description of the process. It also lists prerequisites and steps to start the vLLM API server.



- 참조 – [LambdaLabs](#) Cluster, [vLLM x NVIDIA](#) (24.09)

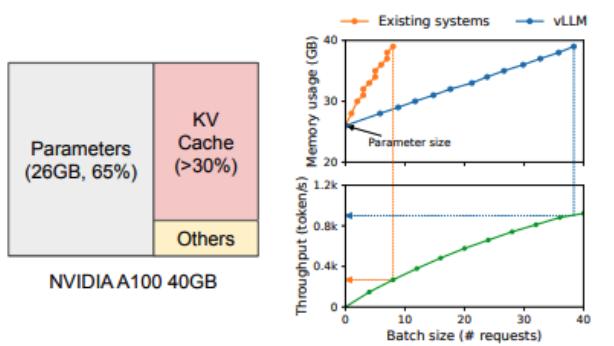
- Paged Attention
 - KV Caching 을 LUT 로 만들어 들고 있다가 다시 사용하는 방법!
 - <https://openai.com/index/api-prompt-caching/>
 - 이제는 모든 framework (ex. TGI) 들이 지원하는 사실상의 업계 표준

Efficient Memory Management for Large Language Model Serving with *PagedAttention*

Woosuk Kwon^{1,*} Zhuohan Li^{1,*} Siyuan Zhuang¹ Ying Sheng^{1,2} Lianmin Zheng¹ Cody Hao Yu³
Joseph E. Gonzalez¹ Hao Zhang⁴ Ion Stoica¹
¹UC Berkeley ²Stanford University ³Independent Researcher ⁴UC San Diego

Abstract

High throughput serving of large language models (LLMs) requires batching sufficiently many requests at a time. However, existing systems struggle because the key-value cache (KV cache) memory for each request is huge and grows and shrinks dynamically. When managed inefficiently, this memory can be significantly wasted by fragmentation and redundant duplication, limiting the batch size. To address this problem, we propose PagedAttention, an attention algorithm inspired by the classical virtual memory and paging techniques in operating systems. On top of it, we build



The figure consists of two parts. The top part is a pie chart titled 'NVIDIA A100 40GB' showing the distribution of memory usage: Parameters (26GB, 65%), KV Cache (>30%), and Others. The bottom part contains two line graphs. The left graph plots 'Memory usage (GB)' against 'Batch size (# requests)', comparing 'Existing systems' (orange line with circles) and 'vLLM' (blue line with diamonds). The right graph plots 'Throughput (tokens/s)' against 'Batch size (# requests)', also comparing 'Existing systems' (orange line with circles) and 'vLLM' (blue line with diamonds).

- <https://arxiv.org/pdf/2309.06180> (23.09)
- 실시간 증명 - <https://simon-mo-workspace.observablehq.cloud/vllm-dashboard-v0/github>

빠르다

- Low Latency
- High Throughput

OpenAI Compatible API

- Role 플레이 대화 형식의 API 가 표준형
 - System 으로 설정 후,
 - User / Assistant 가 주고 받는 형식

```
from openai import OpenAI
client = OpenAI()

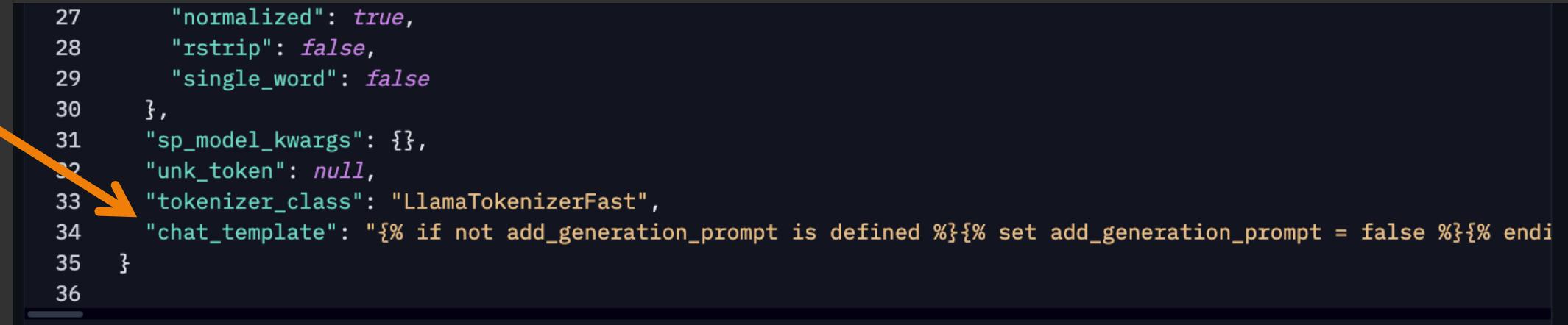
completion = client.chat.completions.create(
    model="gpt-4o",
    messages=[
        { "role": "system", "content": "You are a helpful assistant." },
        { "role": "user", "content": "Hello!" }
    ]
)

print(completion.choices[0].message)
```

- 같은 형식 (OpenAI Compatible) 의 API call 을 받기 위해서는 멀티턴 대화를 파싱 하는 로직이 필요
 - Chat Template
- (참고) OpenAI 는 새로운 Response API 를 발표, 기본 형식으로 채택
 - <https://platform.openai.com/docs/api-reference/responses>

Chat Template

- Huggingface 에 업로드된 모델들은 tokenizer_config.json 에 chat template 을 내장하고 있음
 - (ex) https://huggingface.co/deepseek-ai/DeepSeek-V3-0324/blob/main/tokenizer_config.json



```
27      "normalized": true,
28      "rstrip": false,
29      "single_word": false
30    },
31    "sp_model_kwargs": {},
32    "unk_token": null,
33    "tokenizer_class": "LlamaTokenizerFast",
34    "chat_template": "{% if not add_generation_prompt is defined %}{% set add_generation_prompt = false %}{% endi
35  }
36
```

- .jinja 형식의 파일로도 관리

Llama3.1 – koAlpaca Ninja File

- .jinja 형식의 파일로 관리
- 실습에서 학습한 형식의 포맷
 - "### 지침: "
 - "### 응답: "
- Messages 형태가 들어오면 변환

```
{% if messages[0]['role'] == 'system' %}  
    {% set system_message = messages[0]['content'] | trim + '\n\n' %}  
    {% set messages = messages[1:] %}  
{% else %}  
    {% set system_message = '' %}  
{% endif %}  
  
{ { bos_token + system_message } }  
{% for message in messages %}  
    {% if (message['role'] == 'user') != (loop.index0 % 2 == 0) %}  
        {{ raise_exception('Conversation roles must alternate  
user/assistant/user/assistant/...') }}  
    {% endif %}  
  
    {% if message['role'] == 'user' %}  
        {{ '### 지침:\n' + message['content'] | trim + '\n\n' }}  
    {% elif message['role'] == 'assistant' %}  
        {{ '### 응답:\n' + message['content'] | trim + eos_token + '\n\n' }}  
    {% endif %}  
{% endfor %}  
  
{% if add_generation_prompt %}  
    {{ '### 응답:\n' }}  
{% endif %}
```

vllm Chat Test

- OpenAI API 와 같은 형식으로 json curl
- Response 도 형식에 맞게 응답
- 다른 라이브러리, 모델들과 호환 가능

```
curl -X 'POST'  
'https://epbhkrfpdsdyjqb.tunnel-pt.elice.io/v1/chat/completions'  
-H 'accept: application/json'  
-H 'Content-Type: application/json'  
-d '{ "messages": [  
    { "content": "체첸 공화국에 대해서 설명해", "role": "user" }  
] }'
```

```
{ "id": "chatcmpl-459c6068127b4aba96e58a2c12ba91e6",  
  "object": "chat.completion",  
  "created": 1743147300,  
  "model": "jonhpark/llama3.1-8b-kowiki-instruct-16bit",  
  "choices": [ { "index": 0, "message": { "role": "assistant", "reasoning_content": null, "content": "체첸 공화국은 러시아 남부 카프카스 지방에 위치한 공화국으로, 1991년 소련의 붕괴 후 독립을 선언했다. 이 공화국은 북쪽으로 스타브로풀 지방, 동쪽으로 카라차아이-체첸 공화국, 남쪽으로 조지아, 동쪽으로 아르메니아, 서쪽으로 아제르바이잔과 국경을 ... 발전을 이룰 수 있으며, 이슬람교, 러시아 정교회 등 다양한 종교를 믿고 있으며, 체첸 공화국은 러시아 연방 내에서 가장 빠르게 성장하는 경제를 보이고 있다." }, "logprobs": null, "finish_reason": "stop", "stop_reason": null } ],  
  "usage": { "prompt_tokens": 24, "total_tokens": 1059, "completion_tokens": 1035, "prompt_tokens_details": null },  
  "prompt_logprobs": null }
```

Function Call

- Function call 을 Chat 의 system prompt 에 삽입하기
 - <https://huggingface.co/datasets/glaiveai/glaive-function-calling-v2>

SYSTEM: You are a helpful assistant with access to the following functions.
Use them if required -

```
{  
  "name": "convert_temperature",  
  "description": "Convert temperature from one unit to another",  
  "parameters": {  
    "type": "object",  
    "properties": {  
      "temperature": {  
        "type": "number",  
        "description": "The temperature value"  
      },  
      "from_unit": {  
        "type": "string",  
        "description": "The unit to convert from"  
      },  
      "to_unit": {  
        "type": "string",  
        "description": "The unit to convert to"  
      }  
    },  
    "required": ["temperature", "from_unit", "to_unit"]  
  }  
  
  { "name": "get_current_date",  
    "description": "Get the current date", "parameters": {} }
```

USER:

Hi, I need to convert a temperature from Celsius to Fahrenheit.
The temperature is 30 degrees Celsius.

Function Call

- Function call 을 Chat 의 system prompt 에 삽입하기
 - <https://huggingface.co/datasets/glaiveai/glaive-function-calling-v2>

USER:

Hi, I need to convert a temperature from Celsius to Fahrenheit.
The temperature is 30 degrees Celsius.

ASSISTANT: <functioncall>

```
{"name": "convert_temperature",
"arguments": '{"temperature": 30, "from_unit": "Celsius", "to_unit": "Fahrenheit"}'
<|endoftext|>
```

FUNCTION RESPONSE: {"converted_temperature": 86}

ASSISTANT: The converted temperature from 30 degrees Celsius to Fahrenheit is 86 degrees Fahrenheit. <|endoftext|>

Function Call → Tool Call

- Function call 을 넘어서 다양한 Tool Call 으로 확장
 - Web Search, File Search, Code Interpreter, Image Generation, Computer Use ...
- Model Context Protocol 로 확장됨

```
{  
  "model": "gpt-4o",  
  "input": "우리 팀 주간 리포트를 요약하고 최신 이슈를 덧붙여 주세요.",  
  "tools": [  
    { "type": "file_search" },  
    { "type": "web_search" },  
    { "type": "code_interpreter", "container": { "type": "auto" } },  
    { "type": "mcp", "server_label": "stripe", "server_url": "https://your-mcp-server/tools" }  
  ],  
  "tool_choice": "auto"  
}
```

OpenAI Harmony

- Gpt-oss 와 함께 공개된 OpenAI 의 새로운 Chat Template
- 이제는 Chat 에서의 역할을 넘어서 Channel 이 추가

```
<|start|>system<|message|>You are ChatGPT, a large language model trained by OpenAI.  
Knowledge cutoff: 2024-06  
Current date: 2025-06-28
```

Reasoning: high

```
# Valid channels: analysis, commentary, final. Channel must be included for every message.  
Calls to these tools must go to the commentary channel: 'functions'.<|end|>
```

```
<|start|>developer<|message|># Instructions  
{instructions}<|end|>
```

```
<|start|>user<|message|>What is 2 + 2?<|end|>  
<|start|>assistant
```

```
<|channel|>analysis<|message|>User asks: "What is 2 + 2?" Simple arithmetic. Provide answer.<|end|>  
<|start|>assistant<|channel|>final<|message|>2 + 2 = 4.<|return|>
```

OpenAI Harmony - Tools

```
<|start|>system<|message|>You are ChatGPT, a large language model trained by OpenAI.  
Knowledge cutoff: 2024-06  
Current date: 2025-06-28  
  
Reasoning: high  
  
# Valid channels: analysis, commentary, final. Channel must be included for every message.  
Calls to these tools must go to the commentary channel: 'functions'.<|end|><|start|>developer<|message|># Instructions  
  
Use a friendly tone.  
  
# Tools  
  
## functions  
  
namespace functions {  
  
    // Gets the location of the user.  
    type get_location = () => any;  
  
    // Gets the current weather in the provided location.  
    type get_current_weather = (_: {  
        // The city and state, e.g. San Francisco, CA  
        location: string,  
        format?: "celsius" | "fahrenheit", // default: celsius  
    }) => any;  
  
} // namespace functions<|end|>  
  
<|start|>user<|message|>What is the weather like in SF?<|end|><|start|>assistant
```



OpenAI Harmony - Tools

- Tool Call → 결과 반환 → 실제 LLM 의 답변 생성까지 channel 로 구분되어 대답

```
<|channel|>analysis<|message|>Need to use function get_current_weather.<|end|>  
<|start|>assistant<|channel|>commentary to=functions.get_current_weather  
<|constrain|>json<|message|>{"location": "San Francisco"}<|call|>
```

```
<|start|>functions.get_current_weather to=assistant  
<|channel|>commentary<|message|>{"sunny": true, "temperature": 20}<|end|>
```

```
<|channel|>final<|message|> 현재 샌프란 시스코 날씨는 20도이고 ... <end|>
```

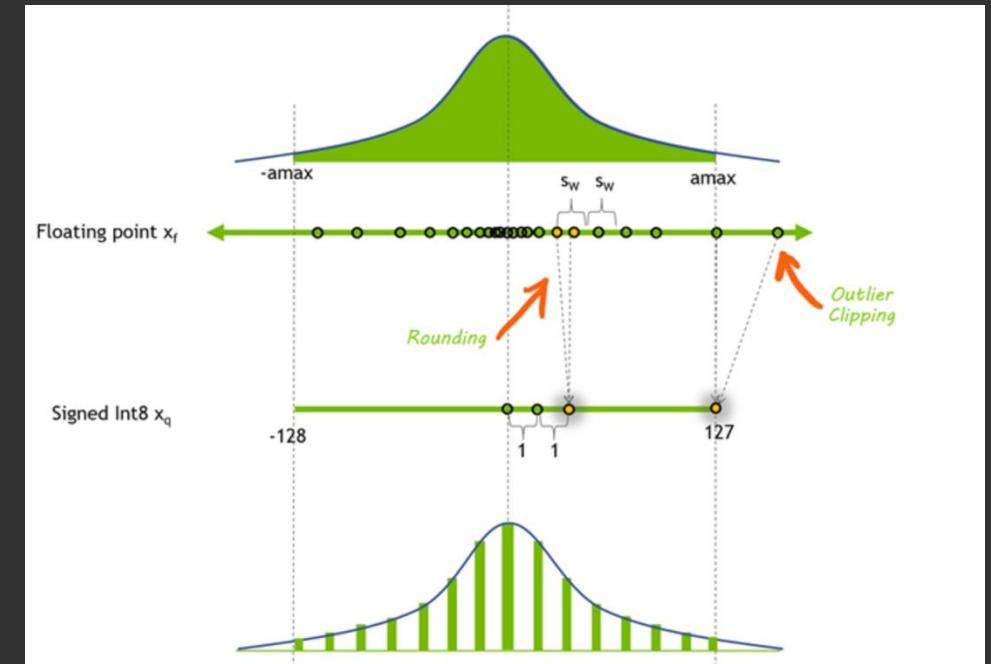
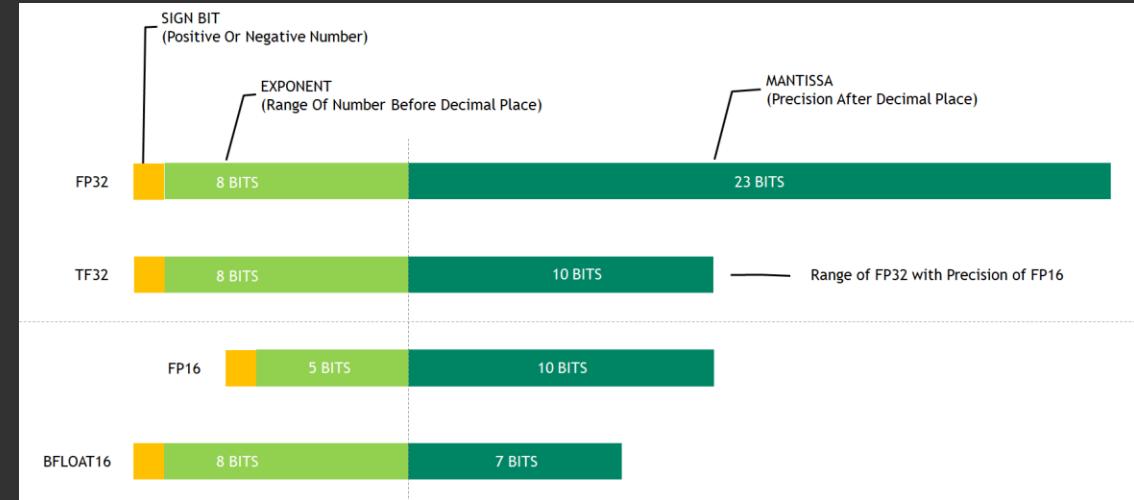
LLM Serving Frameworks

- LLM Model
 - <https://huggingface.co/docs/transformers/index>
- HuggingFace TGI (Text Generation Inference)
 - Compatibility, Supported Quantization, Prometheus/Grafana, Docker Image, ...
 - SLOW!!
- Ollama
 - Easy, but SLOW
- TensorRT
 - NVIDIA backed but SLOW
- Sglang
- vLLM
 - FAST – Low Latency, High throughput



Quantization

- Bfloat16
 - NVIDIA GPU
 - V100 미지원, A100 부터 지원
 - 요즘 모델은 기본 BF16
- Float vs Fixed (INT)
- 추가 Quantization
 - Bitsandbytes
 - GPTQ
 - AWQ
- 기타 경량화 기법
 - Distillation
 - [DL] Pruning, Weight Clustering



Quantized Model

- 예시는 Unsloth의 Bitsandbytes 4bit 모델
- 가장 호환성이 좋은 quantized 모델
- 이 외에도 수 많은 Quantized 모델들이 있음



This is a GitHub repository page for the model `unsloth/gemma-3-12b-it-unsloth-bnb-4bit`. The page includes the following details:

- Repository name: `unsloth/gemma-3-12b-it-unsloth-bnb-4bit`
- Language: Python
- Stars: 5
- Followers: 4.26k
- Topics: Image-Text-to-Text, Transformers, Safetensors, English, gemma3, unsloth, gemma, google, conversational, text-generation-inference, 4-bit precision, bitsandbytes, arxiv:28 papers, License: gemma

Quantized Model

- Unslot 기준 Model 로드시 4bit 를 정해주면 사용 가능

```
model, tokenizer = FastModel.from_pretrained(  
    model_name = "unslot/gemma-3-4b-bit",  
    max_seq_length = 2048,  
    load_in_4bit = True, # 4 bit quantization to reduce memory  
    load_in_8bit = False,  
    full_finetuning = False,  
    # token = "hf_...",  
)
```

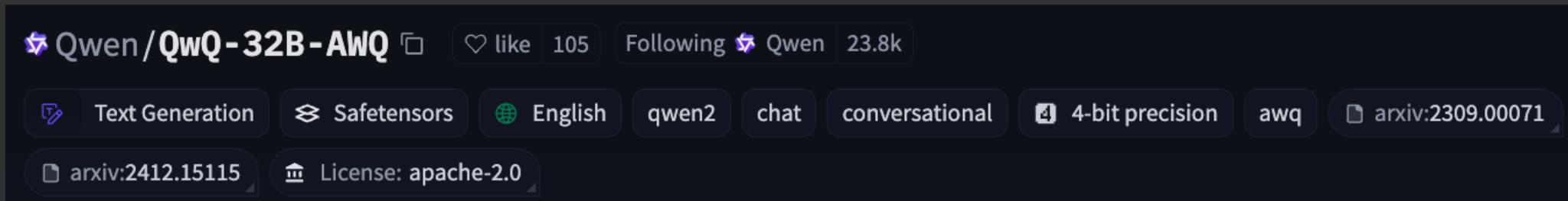
- Vllm 기준 Model 로드시 quantization 옵션을 부여하면 사용 가능

```
$ vllm serve unslot/Meta-Llama-3.1-8B-Instruct-bnb-4bit --quantization bitsandbytes --load-format bitsandbytes
```

```
$ vllm serve Qwen/Qwen2.5-32B-Instruct-AWQ --quantization awq
```

Quantized Model

- Qwen 같은 경우는 공식적으로 AWQ Quantatized 모델 제공



- AWQ (Activation Aware Weight Quantization)
 - <https://arxiv.org/abs/2306.00978>
 - 최근 가장 많이 사용되고 있는 Quantization 기법

Quantized Model 선택

- 메모리를 기준으로

7B BF16 (14+@ GB) 가 좋은가요?

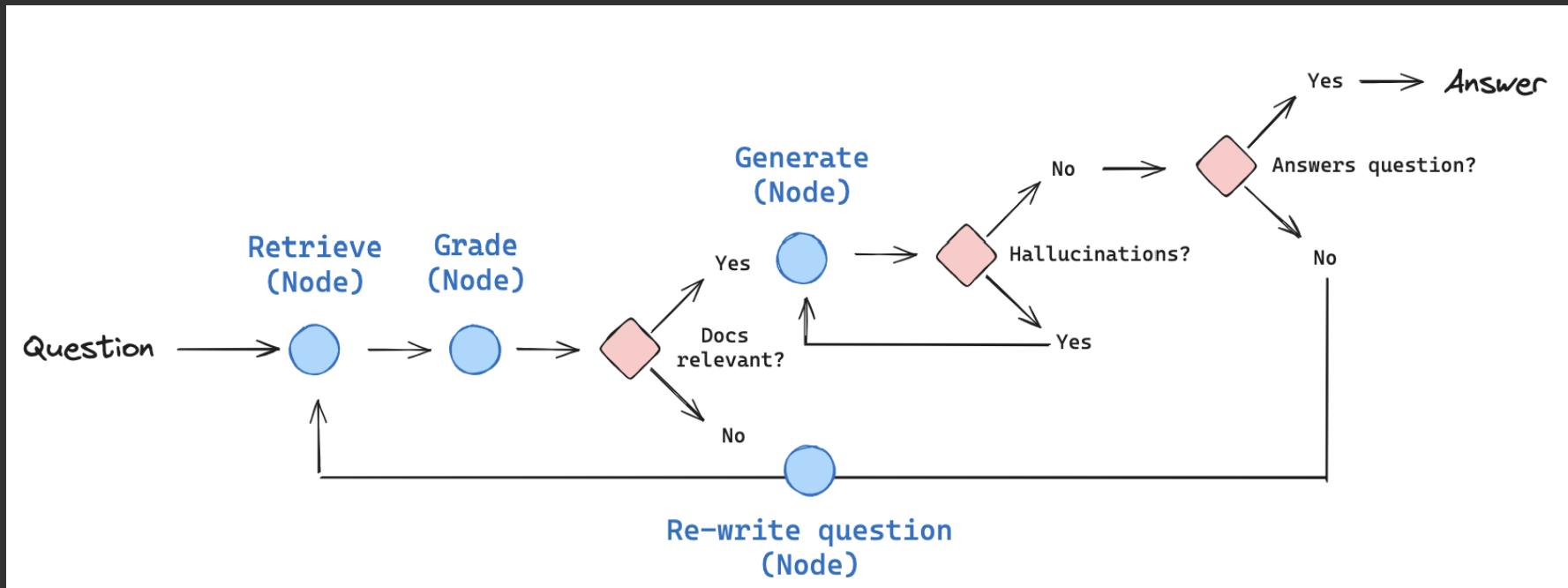
14B FP8 (14+@ GB) 가 좋은가요?

32B 4bit (16+@ GB) 가 좋은가요?

- 대부분의 경우 큰 모델의 quantized 버전 (32B 4bit) 이 좋음
- 안정적 (일관적) 이지 못하는 경향이 있음
- 4bit 보다 작아지면, 갑자기 너무 성능이 떨어지는 현상 관찰됨
- 100B 가 넘어가면 경향이 달라지기도 함
- Fine-Tune 은 가능하면 BF16 을 기준으로 하는 것이 좋음
 - DeepSeek 는 FP8 로 학습하여 8bit 도 충분할 수 있다는 의견이 있음

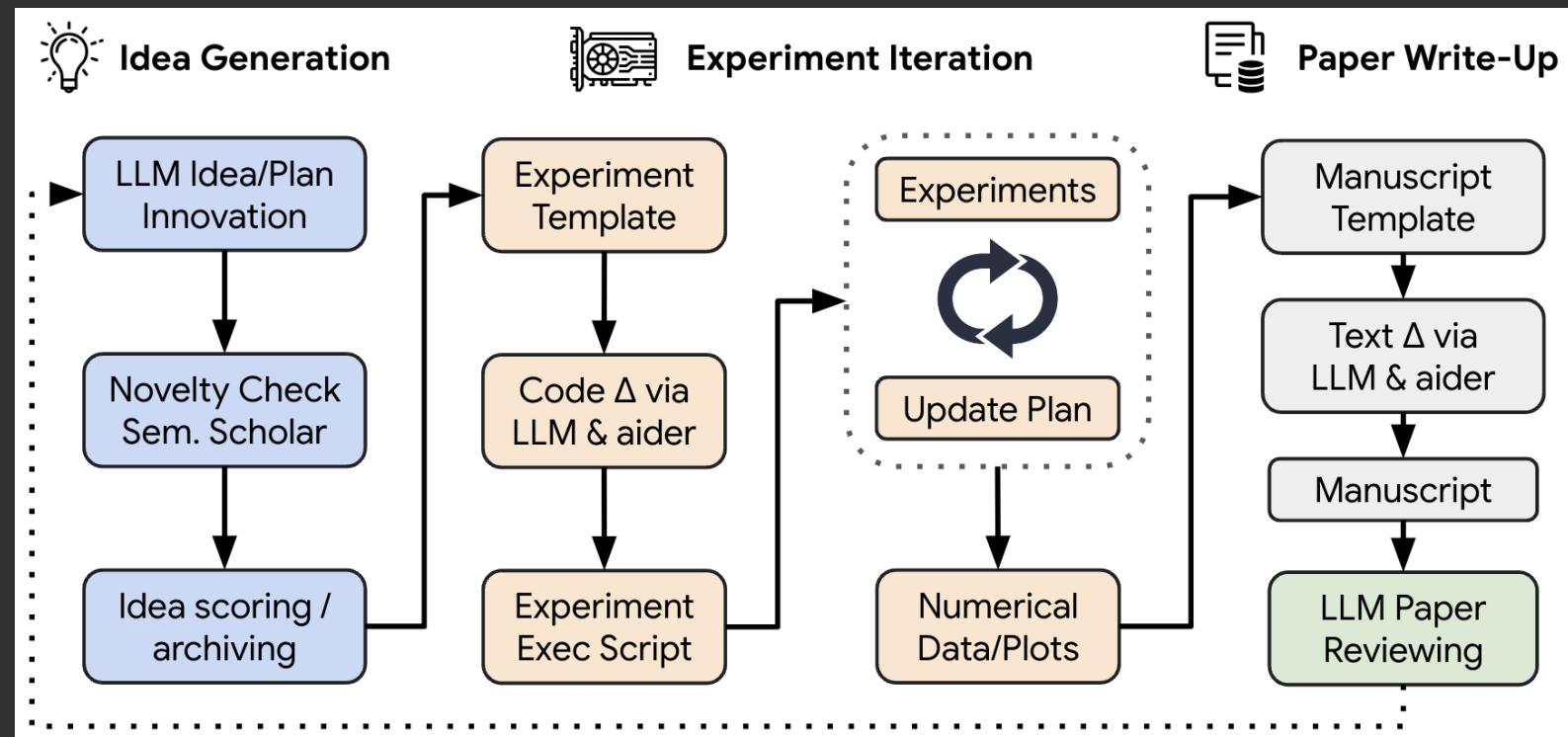
Agentic System 도 Test Time Scaling

- 단순하게 LLM 을 여러 번 사용하는 방법
 - LLM 이 답변하고,
 - 그 답변을 LLM 이 평가하고,
 - 별로라면 다시 LLM 이 보완해서 답변하는 기초적인 Agent
- LLM 이 여러 번 inference 를 하니 Test Time 이 늘어난 것



SAKANA AI, AI-generated paper in ICLR

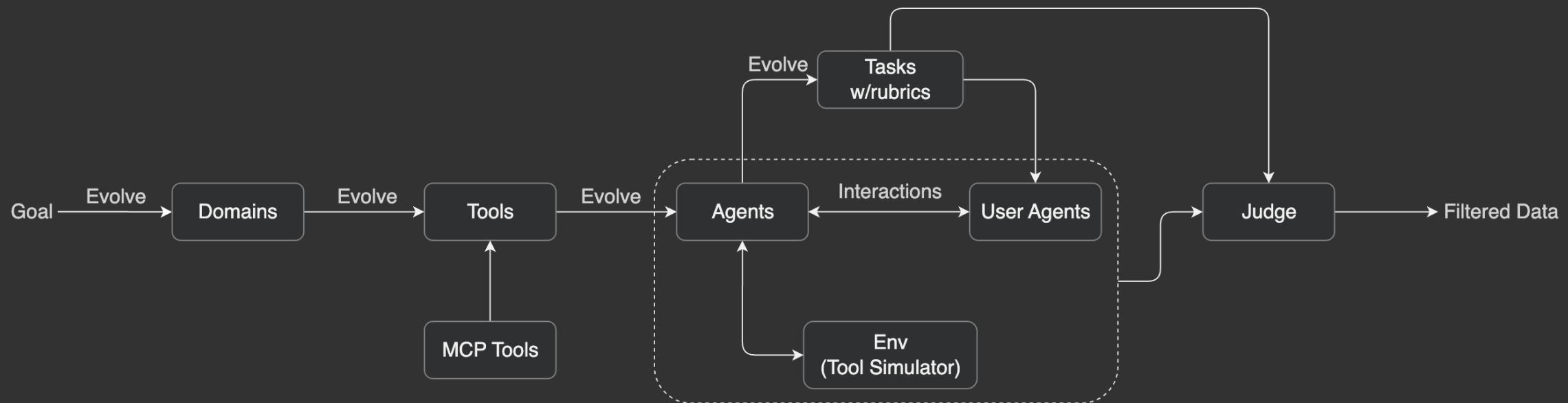
- World's first 100% AI-generated paper passes peer review
 - <https://sakana.ai/ai-scientist-first-publication-jp/>
- 성공적인 예시, LLM 을 여러 번 사용하는 Agent
 - 모델 하나만의 일은 아니지만, TTS 의 좋은 예시
- ICLR 2025 워크샵에 accept, AI paper 라서 publish 되진 않음
 - 리뷰어들에게 AI paper 가 섞여 있음을 사전에 공지
 - 제출된 3개중 1개의 accept
- 출처에 논문과 리뷰가 있음



Kimi K2 - Open Agentic Intelligence

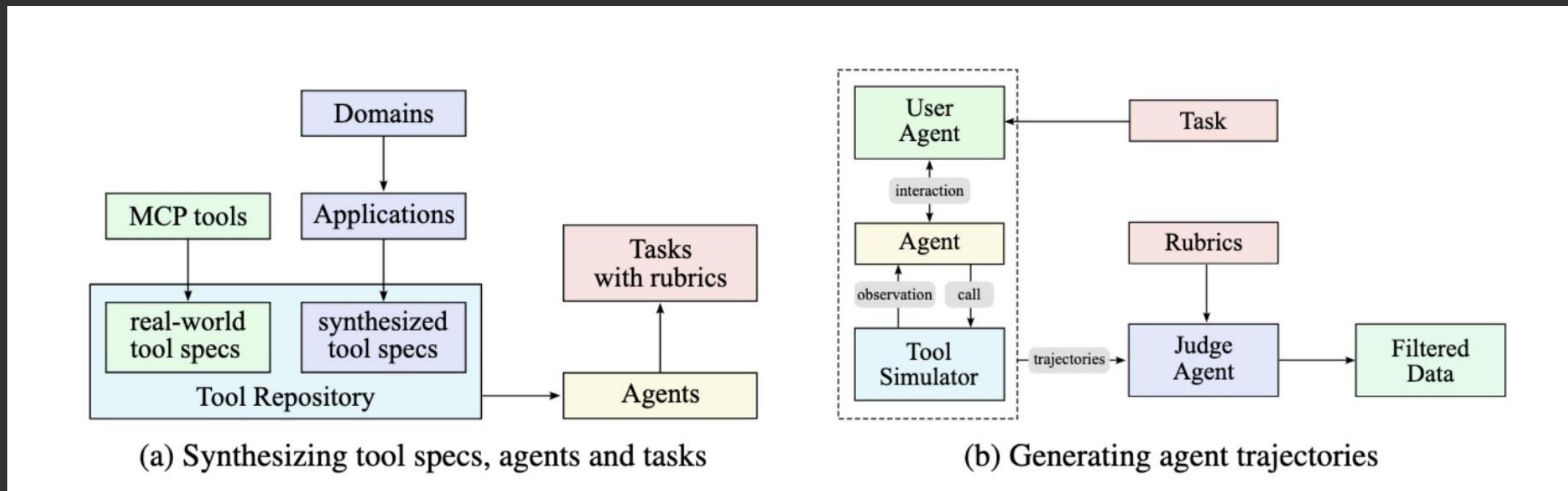
- <https://moonshotai.github.io/Kimi-K2/>
- 1T Open Weight Model, MoE 32B activated, Base 도 같이 공개
- “optimized for agentic tasks, Kimi K2 does not just answer; it acts.”
- 요약하면 Tool Calling 을 적재 적소에 잘 하고, 그 결과를 잘 활용하는 모델
→ Claude Code 에 Opus 대신 쓰면 되겠네!

Large Scale Agentic Data Synthesis



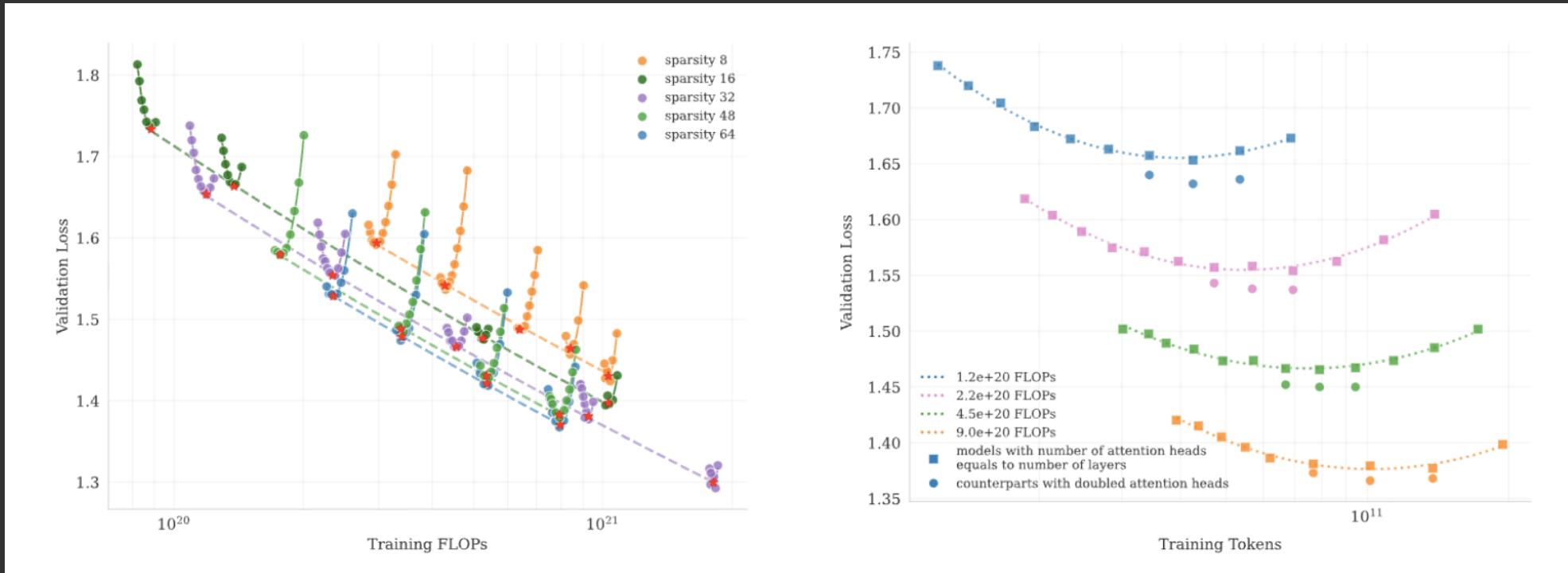
Kimi K2 - Open Agentic Intelligence

- MCP from github --> Synthetic MCP
- 가상의 Agents --> 가상의 Task & Rubrics
- Task 를 풀려보고, Rubrics 으로 채점하면서 Agent trajectories 를 생성
- https://github.com/MoonshotAI/Kimi-K2/blob/main/tech_report.pdf



MoE (Mix of Experts)

- Sparsity Scaling Law



- Qwen3 Next

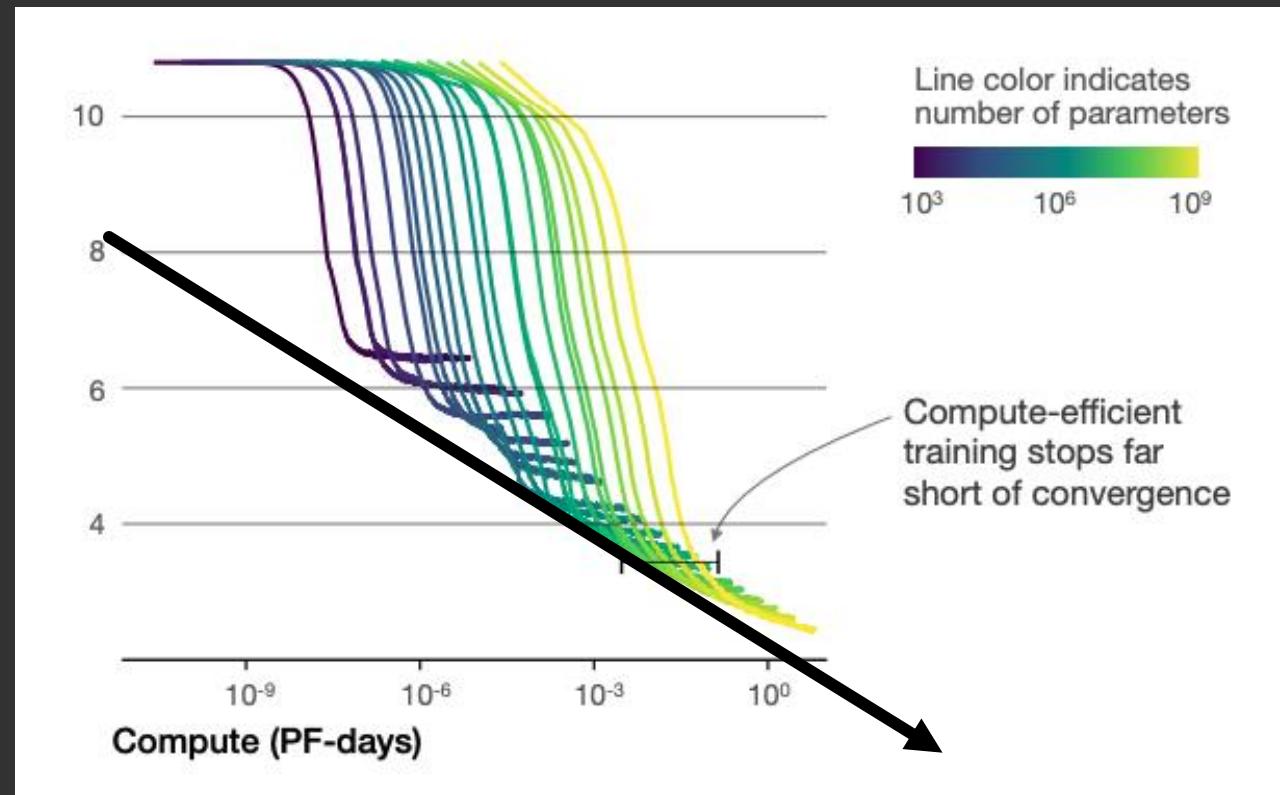
- <https://qwen.ai/blog?id=e34c4305036ce60d55a0791b170337c2b70ae51d&from=home.latest-research-list>

- Kanana MoE 개발기

- <https://tech.kakao.com/posts/716>

Scaling Laws

- Pre-Train Scaling 에 대한 정량적인 실험 결과
 - More Data, More Neuron, More Training Time → Better Performance
- 우리는 현재 어디에 와 있을까? → 아직 끝나지 않음...!



[ref] Kaplan et al (<https://arxiv.org/pdf/2001.08361>)

Scaling Laws - Limitation

- 하드웨어와 비용
 - 마크주커버그 왈 – Llama3 (405B) 학습에 \$10B 가 들었다
 - 전력 사용량이 한 State 가 감당하기 힘들 정도
- 과연 지속이 가능할까

Kyle Corbitt 
@corbtt

Spoke to a Microsoft engineer on the GPT-6 training cluster project. He kvetched about the pain they're having provisioning infiniband-class links between GPUs in different regions.

Me: "why not just colocate the cluster in one region?"
Him: "Oh yeah we tried that first. We can't put more than 100K H100s in a single state without bringing down the power grid." 🤦

[게시물 번역하기](#)

마지막으로 수정됨 오전 7:38 · 2024년 3월 26일 · 176.9만 조회수



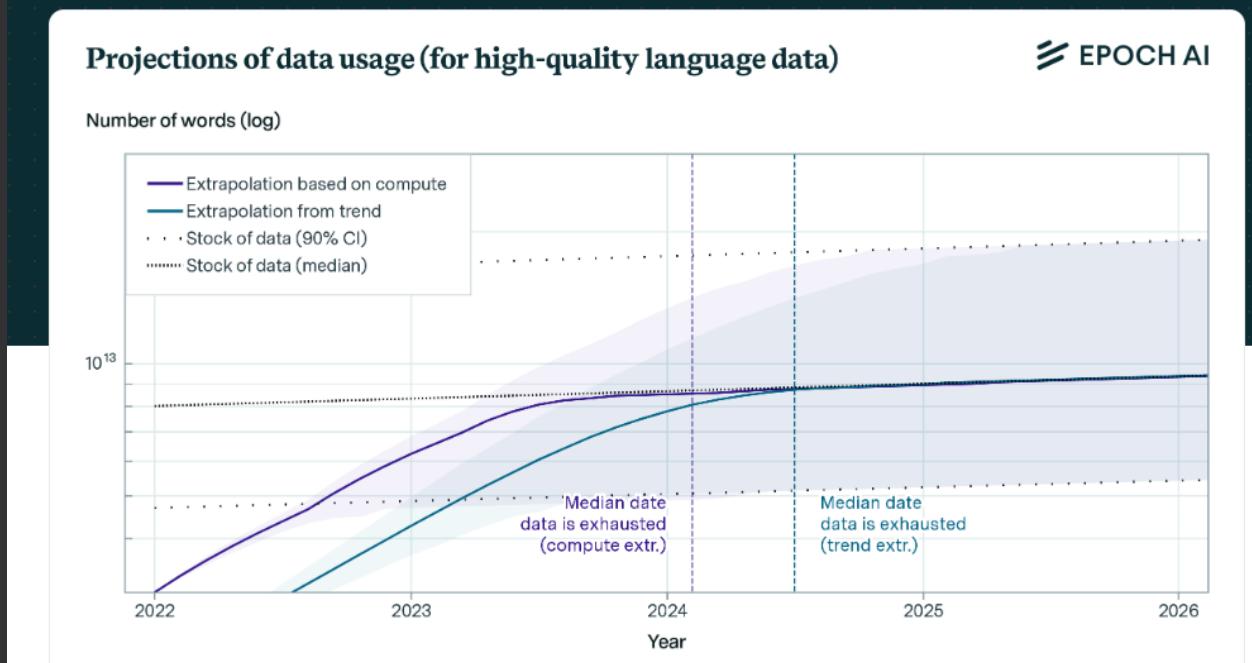
H100 GPU, 약 5천만 원

Scaling Laws - Limitation

- Running Out of Data
 - 인터넷에 존재하는 데이터는 이미 모두 학습 함 (=인간이 역사상 생성한 데이터)
- AI 가 직접 생성한 데이터로 알아서 발전해야함
 - DeepSeek R1 (?!?) 이 가능성을 보여줌

Will We Run Out of ML Data? Evidence From Projecting Dataset Size Trends

Based on our previous analysis of trends in dataset size, we project the growth of dataset size in the language and vision domains. We explore the limits of this trend by estimating the total stock of available unlabeled data over the next decades.



Scaling Laws – Llama4 & GPT4.5

- Llama4 (2025.04)
 - <https://ai.meta.com/blog/llama-4-multimodal-intelligence/>
 - Behemoth : 2T param (288B activated, 16 experts)
 - Scout : 109B with 10M context window, 40T token trained, native multimodal
- Pre-Training GPT-4.5
 - <https://www.youtube.com/watch?v=6nJZopACRuQ>
 - 100,000 GPUs to Train, 2 years project
 - vs Human, 데이터 학습 효율이 천문학적으로 비효율적
 - Compute bound → data bound
 - 10x intelligent than GPT4
 - Evaluate through OpenAI's internal code repo (인터넷에 데이터가 없어서 학습이 불가함)
 - Scaling Law 는 아직 유효함
 - **Dan:** “모든 인간 지식을 가장 짧은 프로그램으로 재현하려는 것이 사전 학습의 본질일 수 있습니다.”

Scaling Laws – Image Edit

- XIAOJUN Podcast w. xianyu (Resnet, StepFun)
 - <https://www.youtube.com/watch?v=vWrYHvSRz0s>
 - 모델이 커지는 데 문과 능력은 좋아지나, 이과 능력을 떨어졌던 것을 관찰했음
 - 추론 과정을 자꾸 뛰어넘기 시작함 (암산, 검산 생략 등)
 - ○ 이 이를 해결하기 위한 방법을 보여줌
 - 다음 과제는 Vision 공간에서의 Reasoning
 - 미로 찾기, 판서, computer use 가 그 예시
 - Vision & Language 가 같은 공간으로 합치기가 어려움
 - Image Edit 데이터가 Multimodal scaling 을 가능하게 해줄 것
 - 그리고 갑자기 나오는 Image Edit Model 들 (Qwen Image Edit, Gemini 2.5 Flash Image)
- Gemini 2.5 Flash Image (aka. Nano Banana)
 - Google Developers Release Note - <https://www.youtube.com/watch?v=H6ZXujE1qBA>
 - “Language Data 에는 Reporting Bias 가 있다”
 - 친구집에 갔다 와서 일반적인 소파 이야기를 하진 않음, 인상적이었던 사건들만 이야기 함

Scaling Laws – Next Scaling?

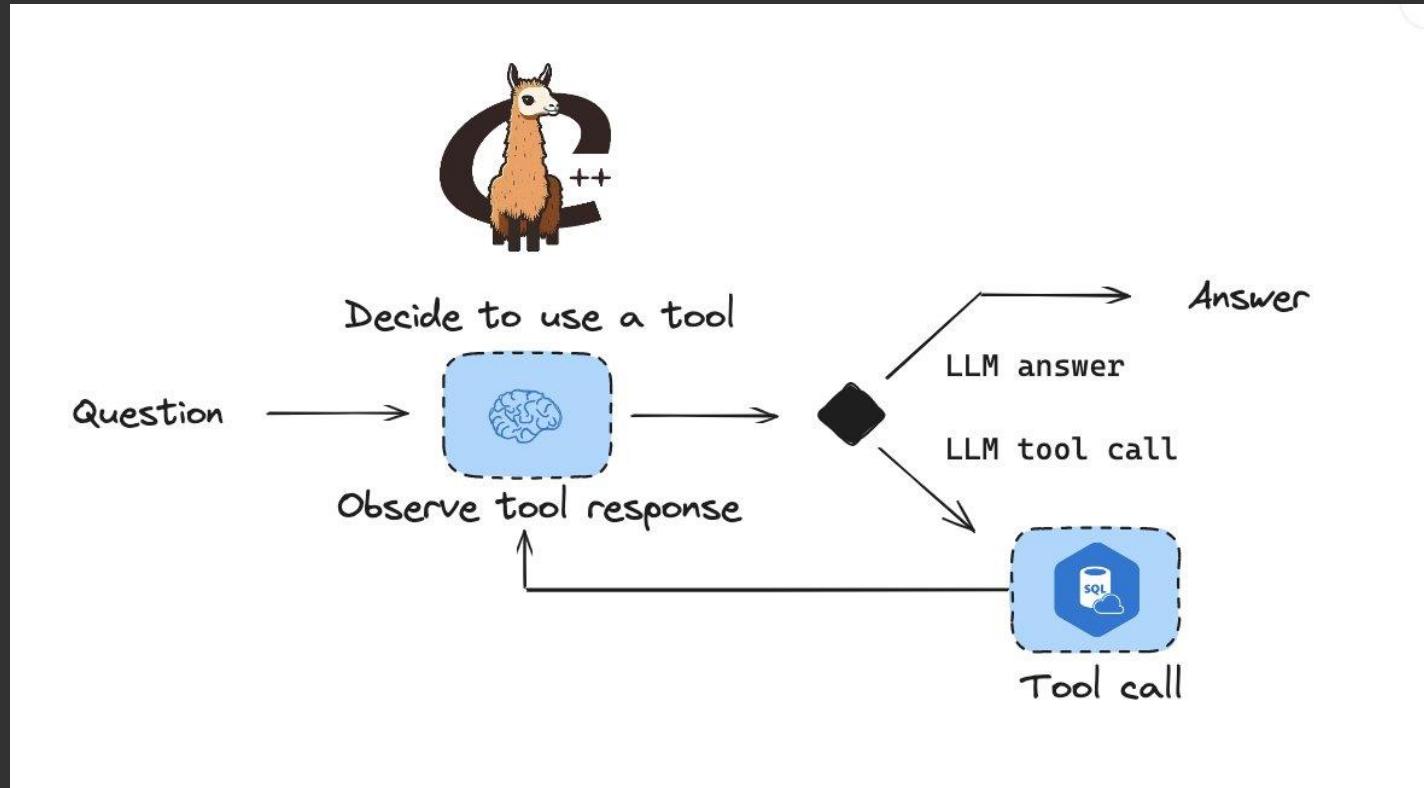
- Welcome to the era of experience, by David Silver & Richard Sutton
 - ref - DeepMind Media (<https://storage.googleapis.com/deepmind-media/Era-of-Experience%20/The%20Era%20of%20Experience%20Paper.pdf>)
- “인간-모방 AI의 속도 한계에 도달했으니, 다시 강화학습 정신으로 돌아가 경험을 주 데이터로 삼자.”
 - AI 는 환경과 상호작용을 직접 하면서 데이터를 수집해야 함
 - Reward Hacking 이 너무 쉬워서, verifiable 한 분야 부터 정복 될 것
 - ex. 코딩 Agents (OpenAI Codex, Claude Code, Google Jules, ...)
- The Physical Turing Test ... by Jim Fan
 - https://www.youtube.com/watch?v=_2NijXqBESI
 - Action model 들이 scaling 을 접근하는 방법

Scaling Laws – for AI Research

- 당분간은 Train Time Scaling 이 더욱 이뤄질 것이 확실시
 - (ex) GPT-4.5, Llama-x, DeepSeek-Vx, Qwen-x
 - 똑똑한 베이스 모델을 적용시킬 문제를 찾자
- Incentivize 는 어떤 효과를 앞으로 더 가져올까
 - Agent? Physical AI? 더 자유도 있는 응용처로 확장해 나갈 것
 - 데이터도 인간이 아니라 AI 가 생성해낼 것
- 결론
 - 문제를 찾고, 그 문제를 위한 핵심 데이터를 모으고, 데이터를 잘 가공/학습 시킬 준비를 하자

Tool Calling

- LLM 이 도구를 사용하는 법
- LLM 이 자체적으로 판단하여 도구를 사용
 - (ex) 구글링, python 코드 실행, ...
- API 호출로 연결, tool call 도 token 일 뿐



Tool Calling

- 모르는 지식이 나왔다면, "Unknown" Feature 가 활성화되면서 <SEARCH> + 검색어 토큰이 생성
 - 검색 결과가 Prompt 에 추가되어 inference 를 이어감
 - SFT 단계에서 Instruct 튜닝 하듯이 tool calling 하는 법도 학습
 - 가지고 있는 tool 은 system prompt 자리에 tool spec 들어 가게 됨
-
- Anthropic 의 MCP 도 같은 원리

Human: "Who is Orson Kovacs?"

Assistant: "

<SEARCH_START>Who is Orson Kovacs?<SEARCH_END>

[...]

Orson Kovacs appears to be ..."