

veRL for Training Coding Agent

姓名

张弛 verl项目发起人

巫锡斌 verl核心maintainer

📍 北京

QCon

全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AI Ops
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

📍 北京

AICon

全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

📍 上海

QCon

全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月

5月

6月

8月

10月

12月

📍 上海

AICon

全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

📍 深圳

AICon

全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

📍 北京

AICon

全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LM Ops
- 具身智能

极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播



参会咨询



查看会议

01

Introduction to verl

verl in 2025

- Same front-end code
- Auto-backend selection



<https://github.com/volcengine/verl>



<https://arxiv.org/abs/2409.19256>



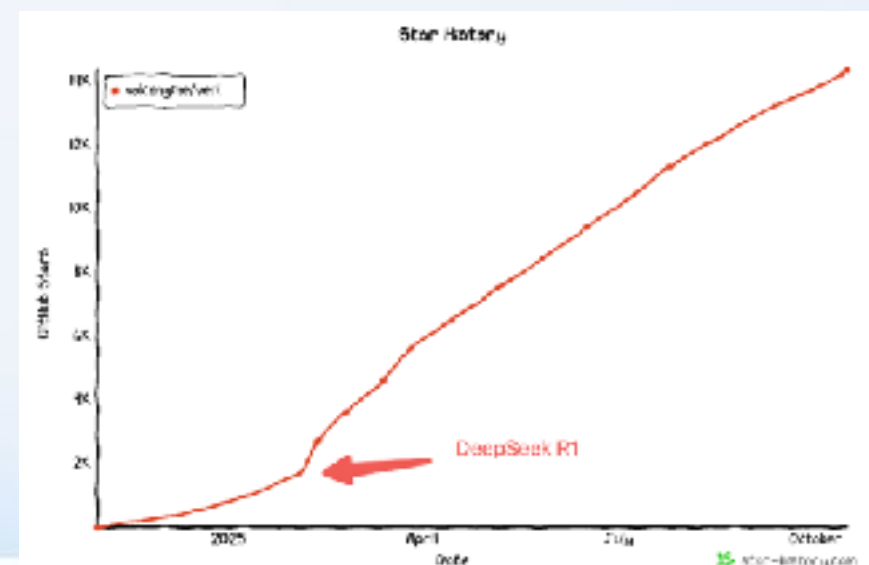
14.6k star, 2.3 fork

Contributors 398



Awesome work using verl

- [TinyZero](#): a reproduction of DeepSeek R1 Zero recipe for reasoning tasks Stars 12k
- [SkyThought](#): RL training for Sky-T1-7B by NovaSky AI team. Stars 2.3k
- [simpleRL_rezero](#): SimpleRL Zero: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild Stars 1.3k
- [Easy-R1](#): Multi-modal RL training framework Stars 1.5k
- [OpenManus-RL](#): LLM Agents RL tuning framework for multiple agent environments. Stars 1.4k
- [rlm](#): async RL training with [verl-pipeline](#) Stars 1.2k
- [RAGEN](#): a general-purpose reasoning agent training framework Stars 1.3k
- [Search-R1](#): RL with reasoning and searching (tool-call) interleaved LLMs Stars 1.3k
- [ReSearch](#): Learning to Reason with Search for LLMs via Reinforcement Learning Stars 1.2k
- [Skywork-DR1](#): Skywork open reasoner series Stars 1.2k
- [JOREL](#): Scaling tool-integrated RL Stars 1.3k
- [Absolute Zero Reasoner](#): A no human curated data self-play framework for reasoning Stars 1.3k
- [verl-agent](#): A scalable training framework for long-horizon LLM/VLM agents, along with a new algorithm GPU Stars 1.1k
- [RL-Factory](#): An easy and efficient RL post-training framework for Agentic Learning Stars 1.1k
- [ReTool](#): ReTool: reinforcement learning for strategic tool use in LLMs. Code release is in progress...
- [verl-tool](#): An unified and easy-to-extend tool agent training framework based on verl Stars 1.1k
- [PRIME](#): Process reinforcement through implicit rewards Stars 1.1k
- [MemAgent](#): MemAgent: Reshaping Long-Context LLM with Multi-Conv RL-based Memory Agent Stars 1.1k
- [PULARIS](#): A Post-training recipe for scaling RL on Advanced Reasoning models Stars 1.1k
- [GUL-R1](#): GUL-R1: A Generalist R1-style Vision-Language Action Model For GUI Agents Stars 1.1k
- [DeepRetrieval](#): RL Training of Search Agent with Search/Retrieval Outcome Stars 1.1k
- [Code-R1](#): Reproducing R1 for Code with Reliable Rewards Stars 1.1k
- [DeepResearcher](#): Scaling deep research via reinforcement learning in real-world environments Stars 1.1k
- [VAGEN](#): Training VLM agents with multi-turn reinforcement learning Stars 1.1k

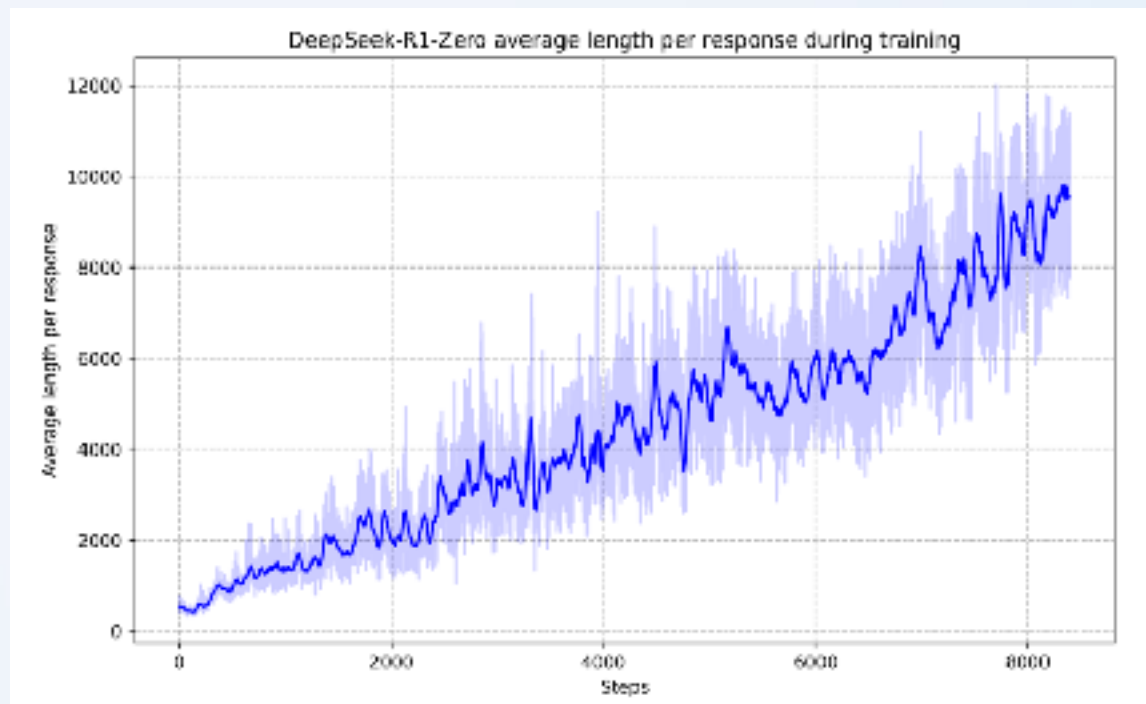




Reinforcement Learning is important

Reinforcement learning (RL) at Bytedance

- Classic Alignment with Human values
- Reasoning: O1/Claude-3.7 performance on math benchmarks
- Image/video/music generation
- Agentic LLM tool using
- Desktop operator, coding assistant, Gaming...



Introduction to Reinforcement Learning

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Supervised fine-tuning

- Learning from labeled examples
- Optimize a single model

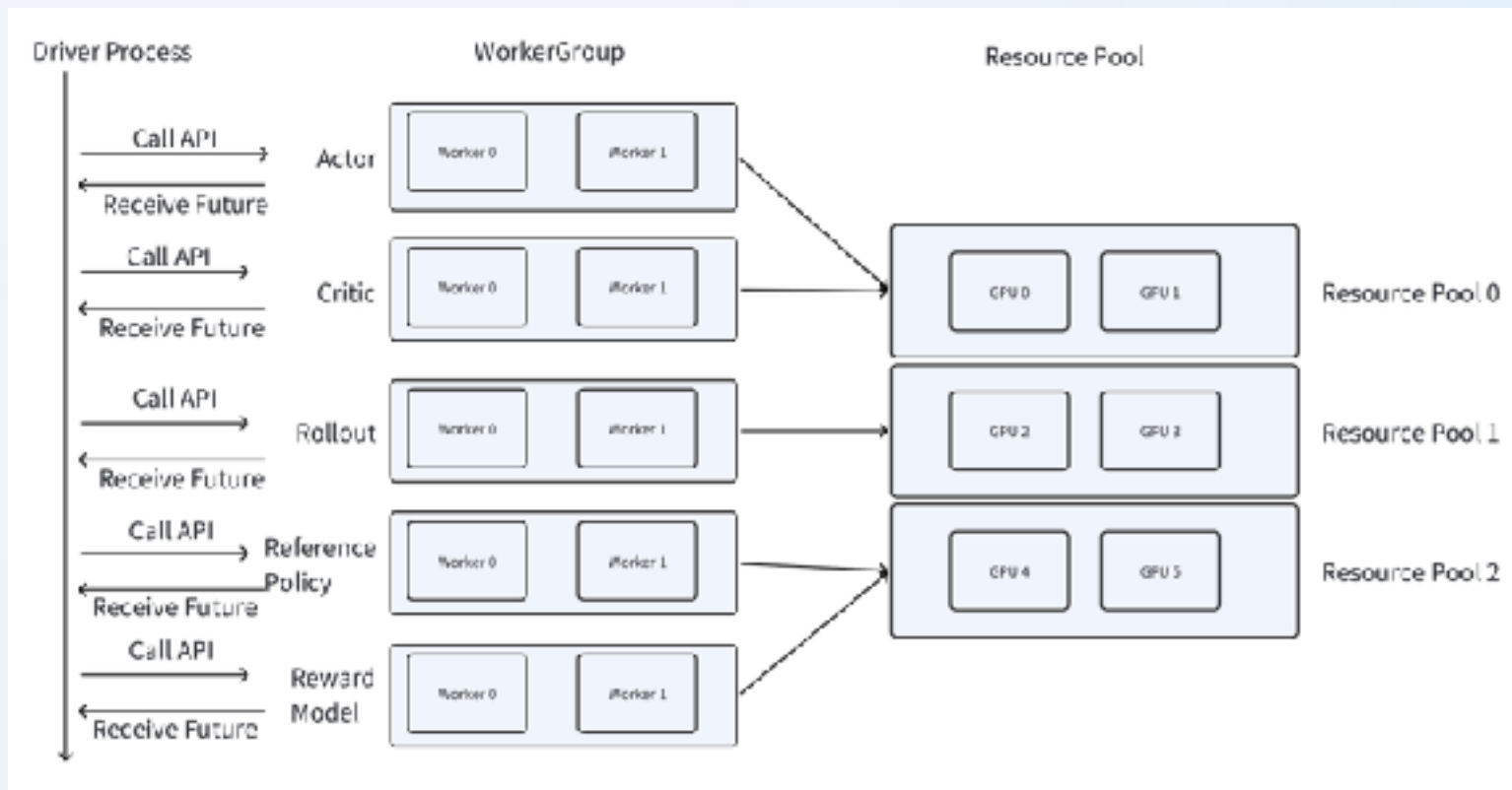
Reinforcement Learning

- Maximize the reward
- Play with multiple models in a single system

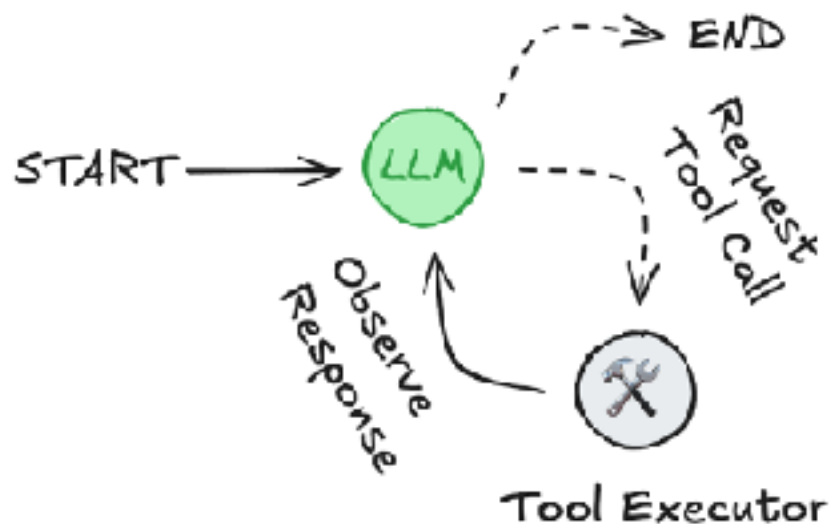
HybridFlow Programming Abstraction

Make LLM training/rollout as a **service**

- Classic Alignment with Human values
- Reasoning: O1/Claude-3.7 performance on math benchmarks
- Image/video/music generation
- Agentic LLM tool using
- Desktop operator, coding assistant, Gaming...



Agents



ReAct (from [langchain-ai](https://github.com/langchain-ai))

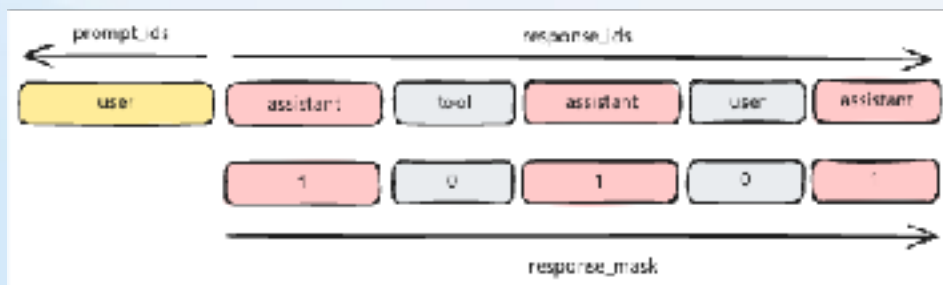
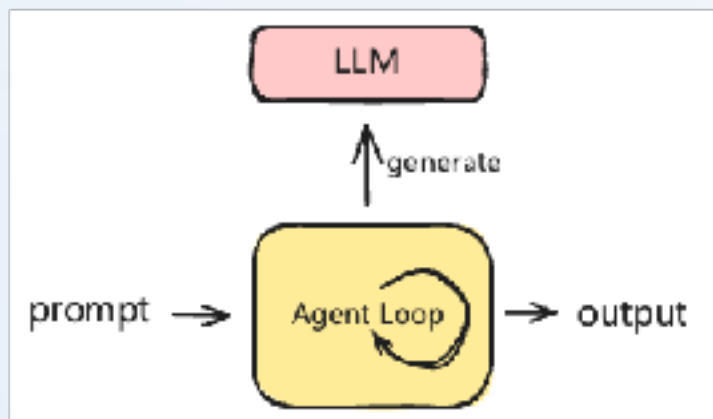
Agent: software systems that use AI to reasoning, planning, and memory and autonomy to make decisions, learn, and adapt.

- Tool calling: Allowing the LLM to select and use various tools as needed.
- Memory: Enabling the agent to retain and use information from previous steps.
- Planning: Empowering the LLM to create and follow multi-step plans to achieve goals.

Agent RL: training LLM to make better decisions in complex, dynamic, real world.

Reinforcement Learning for Agents – AgentLoop

Rollout



AgentLoop: given a user prompt, execute user defined loop, output multi-turn chat history as trajectory.

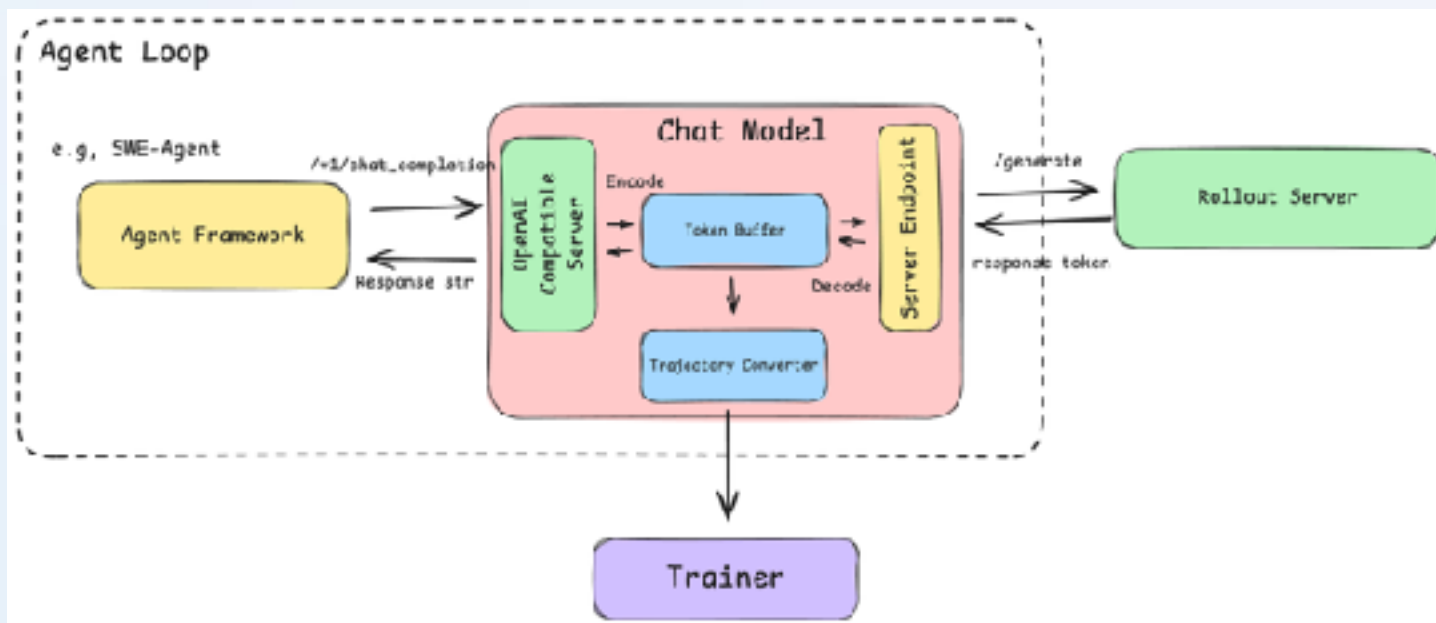
- Search: online web search
- MCP tools: image, video edit, ...
- Code sandbox: execute code, python, java, ...
- Virtual machine: operate browser, ppt, excel, ...
- Android emulator: operate app
- ...

```
class AgentLoopBase(ABC):
    @abstractmethod
    async def run(
        self,
        messages: list[dict[str, Any]],
        sampling_params: dict[str, Any],
    ) -> AgentLoopOutput:
        """Run agent loop to interact with LLM server and environment.

        Args:
            messages (List[Dict[str, Any]]): Input messages.
            sampling_params (Dict[str, Any]): LLM sampling params.

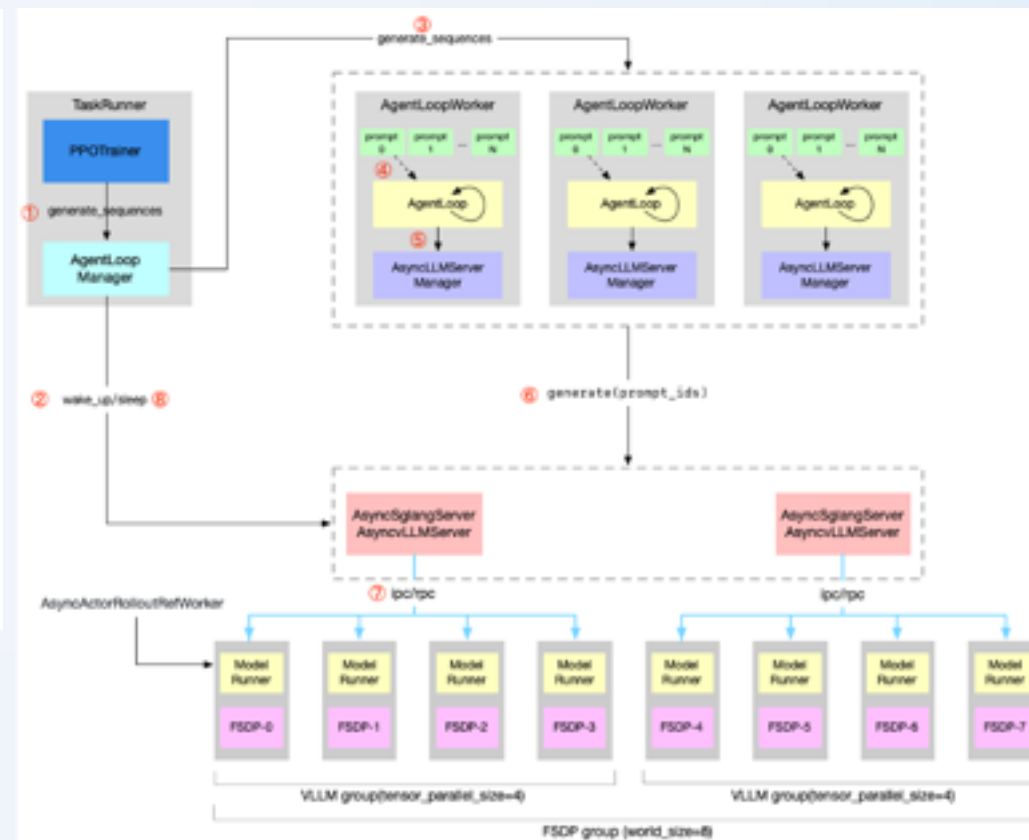
        Returns:
            AgentLoopOutput: Agent loop output.
        """
        raise NotImplementedError
```

Reinforcement Learning for Agents – System

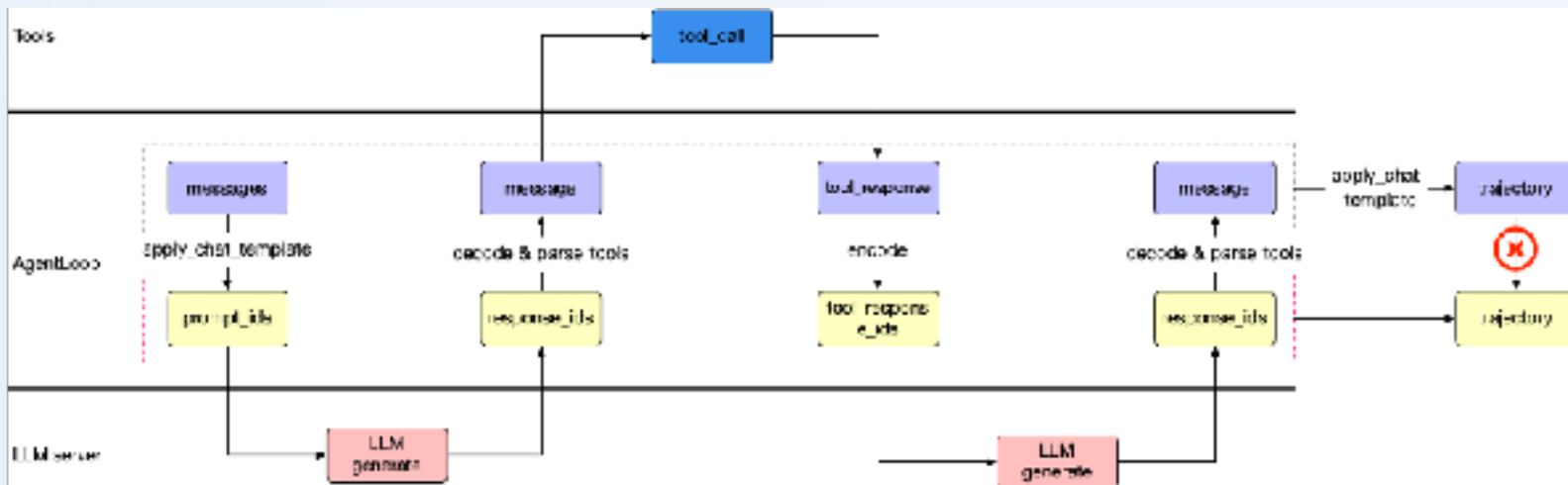


Highlight

- Server mode: vllm/sglang AsyncLLM engine
- Parallel running: asyncio loop run multiple prompts in parallel
- Load balance and sticky session: better kv cache utilization



- Lesson 1: token-in-token-out vs chat completion



- ✗ Tool call regex extraction is irreversible

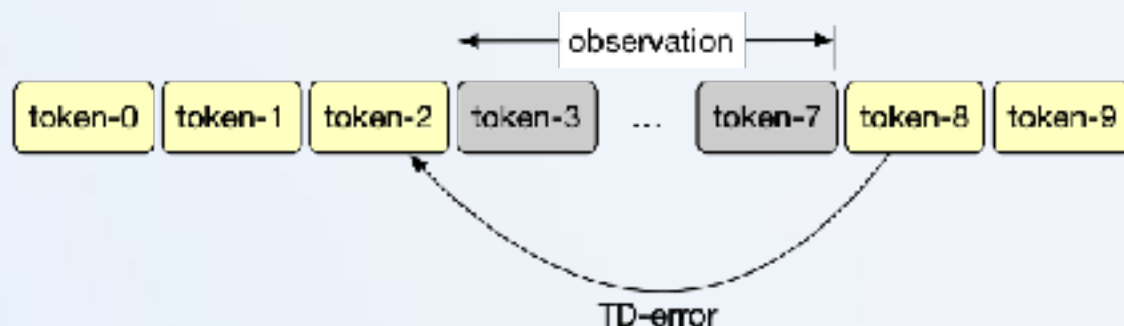
- ✗ Tokenizer encoder-decoder is irreversible

Qwen3 tokenizer

- decode([35946, 20412, 105165]) -> 我是中国人

– encode(我是中国人) -> [104198, 105165]

Pitfalls in Agent RL (cont')



- ✓ Mask out observation token in loss
- ✓ Skip observation token in GAE



Code Agent RL Training with verl



How To Evaluate LLM Code Ability?

Real world software engineering is more complicated than solving a few lines of code:

- Search
- Navigate
- View
- Edit
- Run

We need more comprehensive benchmark!

SWE-bench

The screenshot displays a code editor interface with several panels. The 'File Viewer' panel on the left shows a Python file named 'atmosphere.py' with line numbers 156 to 167. The code includes a function 'guenard84_pe' and a comment about a model developed by expanding Eq. 1 in [1]. The 'Find Files' panel on the right shows search results for 'atmosphere.py' in the directory '/pvlib__pvlib-python/pvlib/'. The 'Search Directory' panel shows search results for 'APPARENT_ZENITH_MODELS' in the directory '/pvlib__pvlib-python/'. The 'Search File' panel shows search results for 'APPARENT_ZENITH_MODELS' in the file '/pvlib__pvlib-python/pvlib/atmosphere.py'.

```
File Viewer
open atmosphere.py

(156 more lines above)
156: def guenard84_pe(temp_air, relative_humidity):
157:     """
158:     Calculates precipitable water (cm) from ambient air temperature (C)
159:     and relative humidity (h) using an empirical model. The
160:     accuracy of this method is approximately 20% for moderate PW (1-3
161:     cm) and less accurate otherwise.
162:
163:     The model was developed by expanding Eq. 1 in [1]:
164:
165:     .. math::
166:         Pe = 8.1 H_v / (R * T)
167:
168:     (76 lines omitted)

143:
144:
145: def bird_hulstrom88_sod(sod380, sod680):
146:     """
147:     Approximate broadband aerosol optical depth.
148:
149:     Bird and Hulstrom developed a correlation for broadband aerosol
150:     depth (AOD) using two wavelengths: 380 nm and 680 nm.
151:
152:     Parameters
153:     -----
154:     sod380 : numeric
155:         AOD measured at 380 nm. (unitless)
156:
157:     (180 more lines below)
```

Find Files
find_file atmosphere.py

Found 2 matches for "atmosphere.py" in /pvlib__pvlib-python/pvlib/
/pvlib__pvlib-python/pvlib/
/pvlib__pvlib-python/pvlib/
/pvlib__pvlib-python/pvlib/

Search Directory
search_dir APPARENT_ZENITH_MODELS

Found 4 matches for "APPARENT_ZENITH_MODELS" in /pvlib__pvlib-python/
/pvlib__pvlib-python/docs/sphinx/source/photosyn/v8.8.0.txt (1
matches)
/pvlib__pvlib-python/pvlib/atmosphere.py (2 matches)
/pvlib__pvlib-python/pvlib/location.py (1 matches)
End of matches for "APPARENT_ZENITH_MODELS" in /pvlib__pvlib-python/

Search File
search_file APPARENT_ZENITH_MODELS

Found 2 matches for "APPARENT_ZENITH_MODELS" in /pvlib__pvlib-python/pvlib/atmosphere.py:
Line 12: APPARENT_ZENITH_MODELS = ['simple', 'kasten1996',
'suttonyoung1988',
Line 15: AIRMASS_MODELS = APPARENT_ZENITH_MODELS +
TRUE_ZENITH_MODELS
End of matches for "APPARENT_ZENITH_MODELS" in /pvlib__pvlib-python/pvlib/atmosphere.py

SWE-bench, Carlos E. Jimenez, et al, 2023

SWE-bench

Scrape Github PRs from 12 popular repositories:

- **problem_statement:** The issue title and body
- **patch:** The gold patch, the patch generated by the PR
- **test_patch:** A test-file patch that was contributed by the solution PR
- ...

Mede Input

Instructions • 1 line
You will be provided with a partial code base and an issue statement explaining a problem to resolve.

Issue • 47 lines
napoleon_use_param should also affect 'other parameters' sectionSubject: napoleon_use_param should also affect 'other parameters' section
Problem
Currently, napoleon always renders the Other parameters section as if napoleon_use_param was False, see source

```
def _parse_other_parameters_section(self, section: str) -> List[str]:  
    # type: (unicode) -> List[unicode]  
    return self._format_fields(C('Other Parameters', ...  
  
def _parse_parameters_section(self, section: str) -> List[str]:  
    # type: (unicode) -> List[unicode]  
    fields = self._consume_fields()  
    if self._config.napoleon_use_param: ...
```

Code • 1043 lines
► README.rst • 102 lines
► sphinx/ext/napoleon/docstring.py • 1295 lines
► Additional instructions • 57 lines

Gold Patch

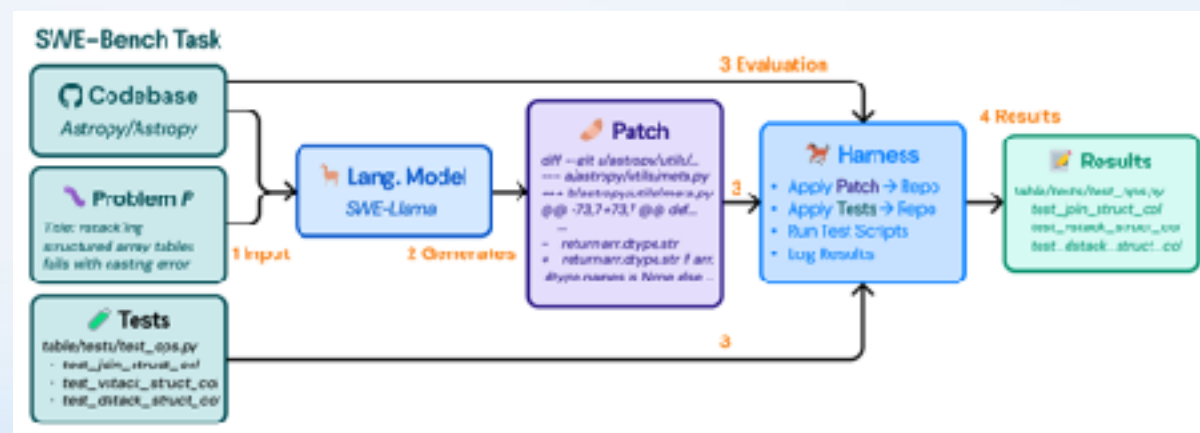
```
sphinx/ext/napoleon/docstring.py  
def _parse_other_parameters_section(self, section: str) -> List[str]:  
    - return self._format_fields(C('Other Parameters', self._consume_fields()))  
    + if self._config.napoleon_use_param:  
    +     # allow to declare multiple parameters at once (ex: x, y: int)  
    +     fields = self._consume_fields(multiple=True)  
    +     return self._format_docutils_params(fields)  
    + else:  
    +     fields = self._consume_fields()  
    +     return self._format_fields(C('Other Parameters', fields))
```

Generated Patch

```
sphinx/ext/napoleon/docstring.py  
def _parse_other_parameters_section(self, section: str) -> List[str]:  
    - return self._format_fields(C('Other Parameters', self._consume_fields()))  
    + return self._format_docutils_params(self._consume_fields())
```

Generated Patch Test Results

```
FAILED NumpyDocstringTest (test_yield_types)  
PASSED TestNumpyDocstring (test_escape_args_and_ovargs_1)  
PASSED TestNumpyDocstring (test_escape_args_and_ovargs_2)  
PASSED TestNumpyDocstring (test_escape_args_and_ovargs_3)  
PASSED TestNumpyDocstring (test_pep526_annotations)  
FAILED NumpyDocstringTest (test_parameters_with_class_reference)  
FAILED TestNumpyDocstring (test_token_type_invalid)  
===== 2 failed, 45 passed, 0 warnings in 3.16s =====
```



SWE-bench

SWE-bench has been widely adopted to evaluate LLM code ability.

Bash Only Verified Lite Full Multimodal					
Bash Only evaluates all LMs with a minimal agent on SWE-bench Verified (details)					
Filter: Open Source ▾ All Tags ▾					
Model	% Resolved	Avg. #	Org	Date	Size
 Claude 4.5 Sonnet (20250929)	70.60	50.56	M	2025-09-29	1.1.3.3
 Claude 4 Opus (20250514)	67.60	51.13	M	2025-05-14	1.0.0
 GPT-5 (2025-08-07) (medium reasoning)	65.00	50.28	G	2025-08-07	1.7.0
 Claude 4 Sonnet (20250514)	64.93	50.37	M	2025-07-23	1.0.0
 GPT-5 mini (2025-08-07) (medium reasoning)	59.80	50.04	G	2025-08-07	1.7.0
 o3 (2025-04-16)	50.40	50.03	G	2025-07-23	1.0.0
 Qwen3-Coder 480B/435B Instruct	55.40	-	Q	2025-08-02	1.0.0
 GLM-4.5 (2025-08-22)	54.20	50.00	G	2025-08-22	1.9.1
 Gemini 2.5 Pro (2025-05-06)	53.60	50.29	G	2025-07-25	1.0.0
 Claude 3.7 Sonnet (20250219)	52.00	50.05	M	2025-07-23	1.0.0
 o1 (2025-05-14)	45.00	45.00	G	2025-07-23	1.0.0

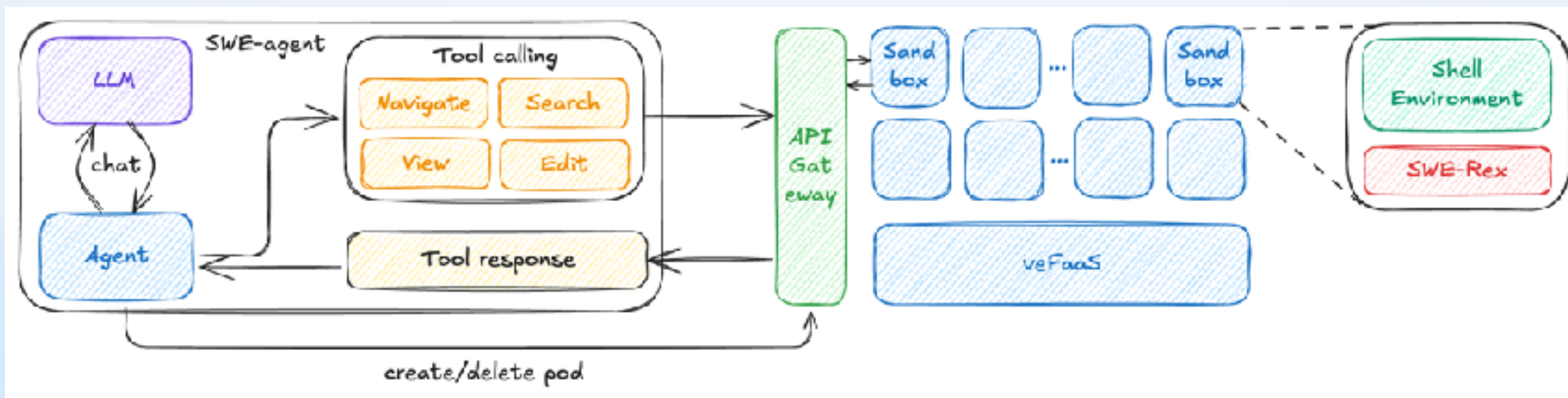
Bash Only <u>Verified</u> Lite Full Multimodal				
Verified is a human-filtered subset of 500 instances [details]				
Filter: Open Source ▾ All Tags ▾				
Model	% Resolved	Org	Date	Size
 TFAR + Coqito-SWE-Code	78.80		2025-09-28	🔗
TFAR	75.20		2025-09-12	🔗
 Lingxi-v1.5_Claude-4-sonnet-20250514	74.60		2025-07-20	🔗
 JeyCode	74.60		2025-09-15	🔗
Refact.ai Agent	74.40		2025-09-03	🔗
 OpenHerds + GPT-5	73.80		2025-09-07	🔗
 Meatless Tools + Claude 4 Sonnet	70.00	-	2025-09-11	🔗
 mini-SWE-agent + Claude 4.5 Sonnet (20250929)	70.60		2025-09-20	🔗
Refact.ai Agent	70.40		2025-09-15	🔗
 OpenHalls + Claude 4 Sonnet	70.40		2025-09-24	🔗
Acument Agent v1	70.40		2025-09-10	🔗

[SWE-bench Leaderboards](#)

Build Large Scale SWE-bench Infrastructure

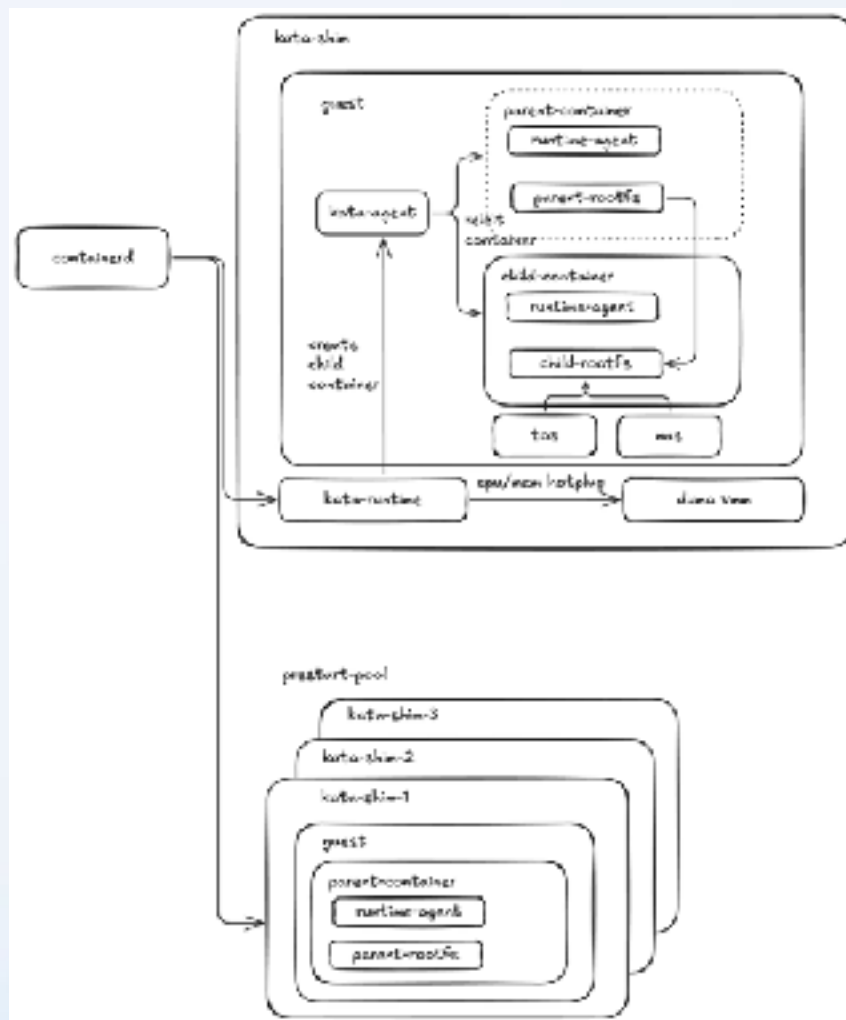
SWE-bench Infrastructure

- **SWE-agent**: enable LLM to autonomously use tools to fix issues.
- **SWE-Rex**: runtime interface for interacting with sandboxed shell environments.
- **veFaaS**: Volcano Engine Function as a Service, provides image cache, sandbox isolation, fast deployment.



veFaaS

- Kata container with strong isolation
- Image warmup with nydus and P2P distribution
- Image affinity scheduling

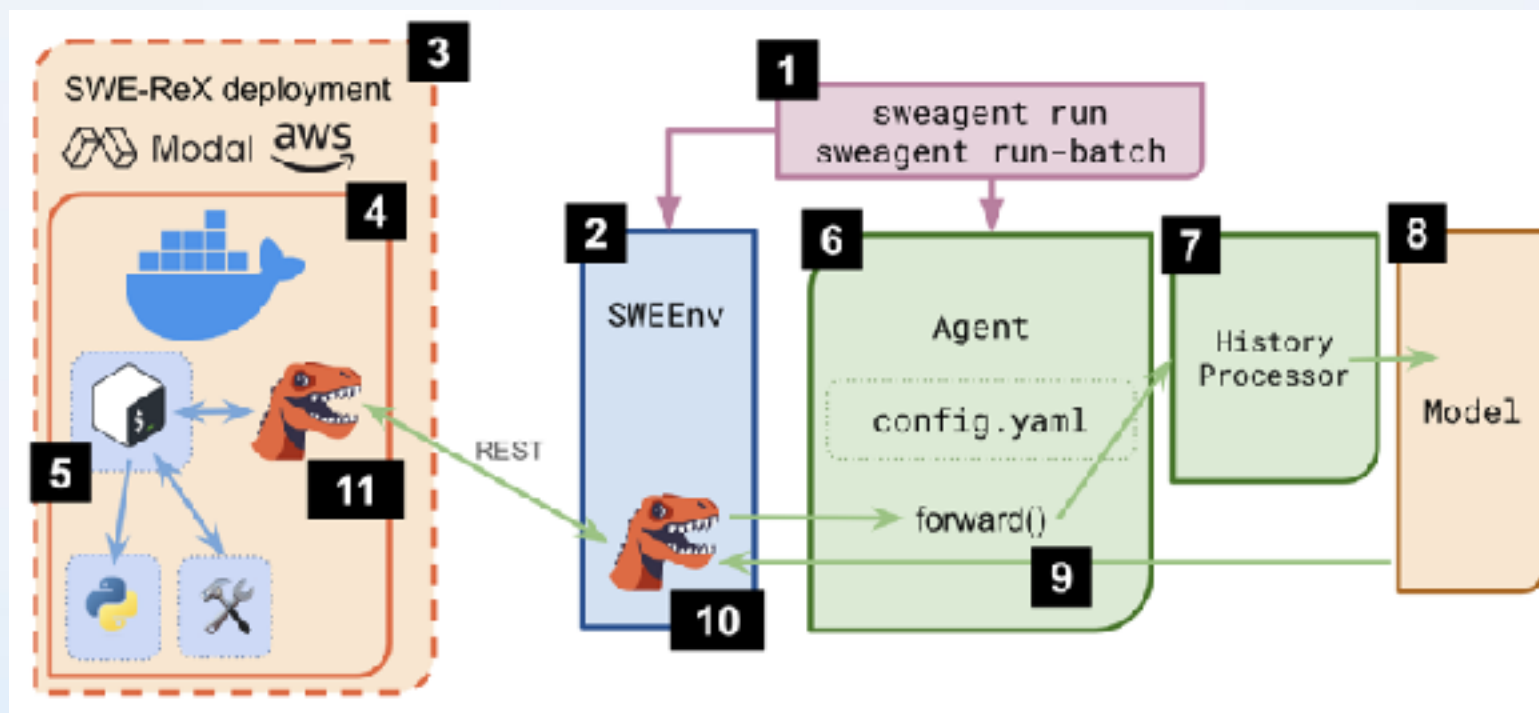


SWE-agent

Step 1~5: setup container, install tools, and initialize shell session

Step 6: setup agent with tool config yaml

Step 7~11: agent query model, parse action and execute shell command



[SWE-agent architecture](#)



SWE-agent

SWE-agent implementation

```

1 class SWEAgent:
2
3     @Type.runnable()
4     def forward(self, history: list[dict[str, Any]] -> StepOutput:
5         self._hook.on_model_query(messages=history, agent=self.name)
6
7         # Step 1: Query the model with tools
8         output = self.model.query(history, tools=self.tools.config.tools)
9         step_output = output["message"]
10
11         # Step 2: Parse the actions from the model output
12         step_thought, step_action = self.tools.parse_actions(output)
13         step_thinking_blocks = output.get("thinking_blocks", [])
14         if output.get("tool_calls") is not None:
15             step_tool_call_ids = [call["id"] for call in output["tool_calls"]]
16             step_tool_calls = output["tool_calls"]
17             step_extra_info["raw"] = output.get("raw", "")
18             self.logger.info(
19                 f"THOUGHT:\n{step_thought}\n\nACTION:\n{step_action.strip()}"
20             )
21             self._hook.on_actions_generated(step=step)
22
23         # Step 3: Handle the action
24         return self.handle_action(step)
25
26     @Type.runnable()
27     def run(
28         self,
29         env: SWEEnv,
30         problem_statement: ProblemStatement | ProblemStatementConfig,
31         output_dir: Path = Path("."),
32     ) -> AgentRunResult:
33         self.setup_env(env, problem_statement=problem_statement, output_dir=output_dir)
34
35         # Run action/observation loop
36         self._hook.on_run_start()
37         step_output = StepOutput()
38         while not step_output.done:
39             step_output = self.step()
40             self.save_trajectory()
41             self._hook.on_run_done(trajectory=self.trajectory, info=self.info)
42
43         data = self.get_trajectory_data()
44         return AgentRunResult(info=data["info"], trajectory=data["trajectory"])

```

Tool definition

```

tools:
  str_replace_editor:
    signature: |
      str_replace_editor commands: paths [(file_text)] [view_range] [old_str] [new_str] [insert_line]
    # This description was taken from openhands:
    # https://github.com/All-Hands-AI/OpenHands/blob/main/openhands/agenthub/codeact_agent/function_calling.py
    docstring: >
      Custom editing tool for viewing, creating and editing files
      * State is persistent across command calls and discussions with the user
      * If 'path' is a file, 'view' displays the result of applying 'cat -n'. If 'path' is a directory, 'view' lists non-hidden files and all entries up to 2 levels deep
      * The 'create' command cannot be used if the specified 'path' already exists as a file
      * If a 'command' generates a long output, it will be truncated and marked with '<response clipped>'
      * The 'undo_edit' command will revert the last edit made to the file at 'path'

    Notes for using the 'str_replace' command:
    * The 'old_str' parameter should match EXACTLY one or more consecutive lines from the original file. Be mindful of whitespaces!
    * If the 'old_str' parameter is not unique in the file, the replacement will not be performed. Make sure to include enough context in 'old_str' to make it unique
    * The 'new_str' parameter should contain the edited lines that should replace the 'old_str'

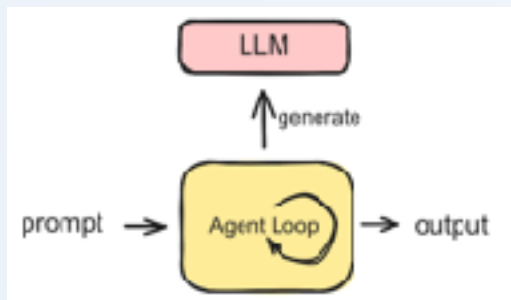
    arguments:
    - name: command
      type: string
      description: "The commands to run. Allowed options are: 'view', 'create', 'str_replace', 'insert', 'undo_edit'."
      required: true
      enum: ["view", "create", "str_replace", "insert", "undo_edit"]
    - name: path
      type: string
      description: "Absolute path to file or directory, e.g. '/testbed/file.py' or '/testbed'."
      required: true
    - name: file_text
      type: string
      description: "Required parameter of 'create' command, with the content of the file to be created."
      required: false
      argument_format: "file_text {value}"
    - name: old_str
      type: string
      description: "Required parameter of 'str_replace' command containing the string in 'path' to replace."
      required: false
      argument_format: "old_str {value}"
    - name: new_str
      type: string
      description: "Optional parameter of 'str_replace' command containing the new string (if not given, no string will be added). Required parameter of 'insert' command"
      required: false
      argument_format: "new_str {value}"
    - name: insert_line

```



Train SWE-agent with verl: rollout

1. AgentLoop Abstraction



AgentLoop: given a user prompt, execute user defined loop, output multi-turn chat history as trajectory.

- **Search**: online web search
- **MCP**: image, video edit, ...
- **Code sandbox**: execute code, python, java, ...
- **Virtual machine**: operate browser, ppt, excel, ...
- **Android emulator**: operate app
- ...

```
class AgentLoopBase(ABC):
    @abstractmethod
    async def run(
        self,
        messages: list[dict[str, Any]],
        sampling_params: dict[str, Any],
    ) -> AgentLoopOutput:
        """Run agent loop to interact with LLM server and environment.

        Args:
            messages (List[Dict[str, Any]]): Input messages.
            sampling_params (Dict[str, Any]): LLM sampling params.

        Returns:
            AgentLoopOutput: Agent loop output.
        """
        raise NotImplementedError
```



Train SWE-agent with verl: rollout

2. SWEAgentLoop

```
class SWEAgentLoop(AgentLoopBase):
    async def run(self, sampling_params: dict[str, Any], **kwargs) -> AgentLoopOutput:
        tool_config_data = agent_config_data.get("tools", {})
        tool_config = ToolConfig(-

        default_agent_config = DefaultAgentConfig(-

        deployment_config = VefasDeploymentConfig(-

        env_config = EnvironmentConfig(-
        env_config.deployment = deployment_config

        problem_statement = TextProblemStatement(-

        single_run_config = RunSingleConfig(-

        single_run = RefRunSingle.from_config(single_run_config)

        # replace chat model and agent.
        model = ChatModel(-
        agent = SWEAgent.from_config(default_agent_config, model=model)

        try:
            result = await agent.run(single_run)
            output_dir = single_run.output_dir
            patch_output_dir = Path(output_dir) / instance_id
            patch_output_file = patch_output_dir / f"{instance_id}.pred"

            test_spec = make_test_spec(metadata)
            with open(patch_output_file, "r") as f:
                pred = json.load(f)

            vefas_config = VefasDeploymentConfig(-
            deployment = VefasDeployment.from_config(vefas_config)

            eval_result = await run_eval(-
            print(eval_result)

            return model.convert_to_agent_output(result, eval_result)
```

← SWE-agent instance

← Run specific task: image, commit, problem, ...

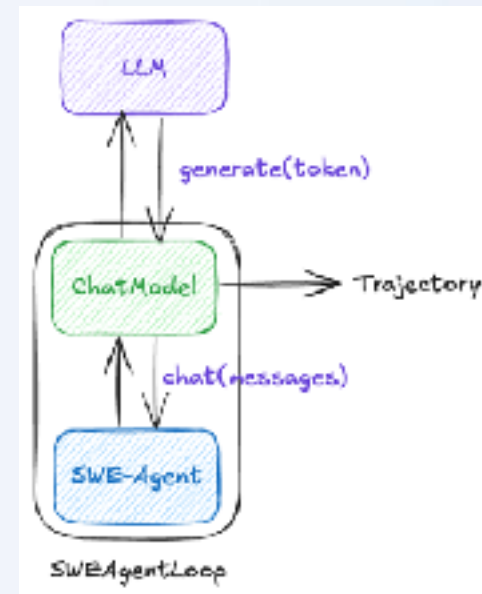
← Run tests with patch to get reward score

← Convert messages to AgentLoopOutput



Train SWE-agent with verl: rollout

3. Bridge the gap between chat completion and token-in-token-out



ChatModel

- Encode messages and tools to tokens in request
- Decode tokens and
- Keep chat history in

```
{
  "content": "How many vertical asymptotes does the graph of  $y = \frac{1}{x^2 + x - 6}$  have? Let's think step by step and output the final answer within \boxed{}.",
  "role": "user",
  "content": "To determine the number of vertical asymptotes for the function  $y = \frac{1}{x^2 + x - 6}$ , we need to find the values of  $x$  where the denominator equals zero, as these points are where the function is undefined.",
  "role": "assistant",
  "tool_calls": [
    {
      "id": "call_4d823672f48d45158ede5e49",
      "function": {
        "arguments": "({code: \"from sympy import symbols, solve\\nrx = symbols('x')\\nroots = solve(x**2 + x - 6, x)\\nroots})\",
        "name": "code_interpreter",
        "type": "function",
        "index": 0
      },
      "role": "tool",
      "content": "[-3, 2]\\n"
    }
  ],
  "content": "The roots of the equation  $x^2 + x - 6 = 0$  are  $x = -3$  and  $x = 2$ . These are the values of  $x$  where the denominator is zero, which means the function  $y = \frac{1}{x^2 + x - 6}$  is undefined at the",
  "role": "assistant"
}
```



Train SWE-agent with verl: training

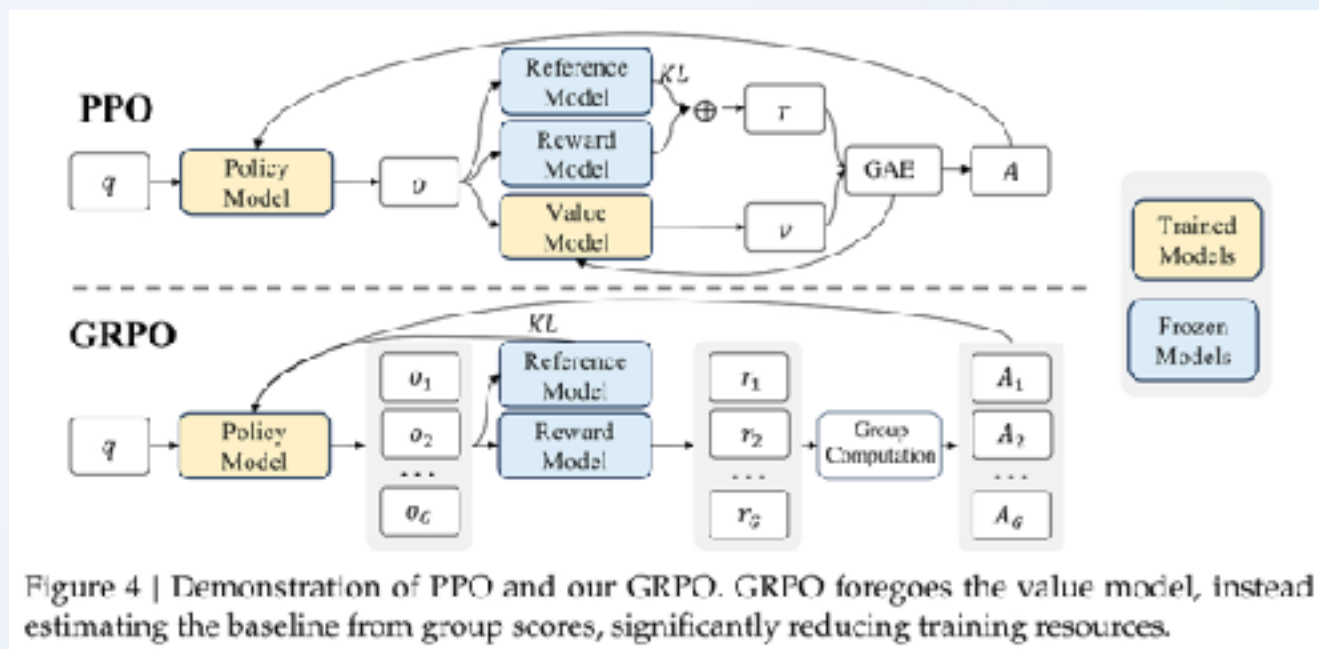
Reinforcement Learning with Verifiable Reward(RLVR)

- reward: model generated patch pass all test cases
- algorithm: PPO or GRPO/DAPO/GSPO

Group relative advantage estimation:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \dots, r_G\})}{\text{std}(\{r_1, r_2, \dots, r_G\})}$$

We will release the training recipe soon!



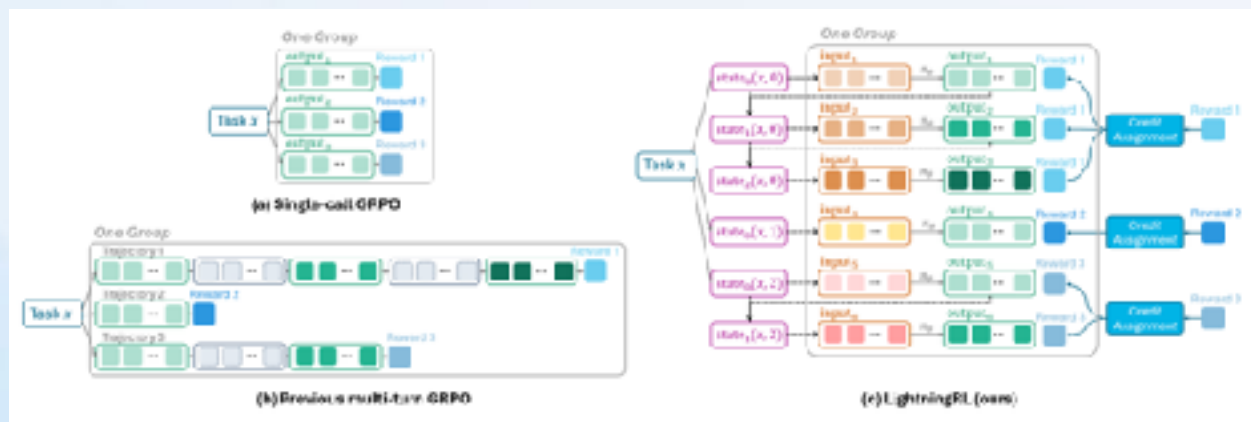
DeepSeekMath, Zhihong Shao, et al, 2024

Limitation and Future Work

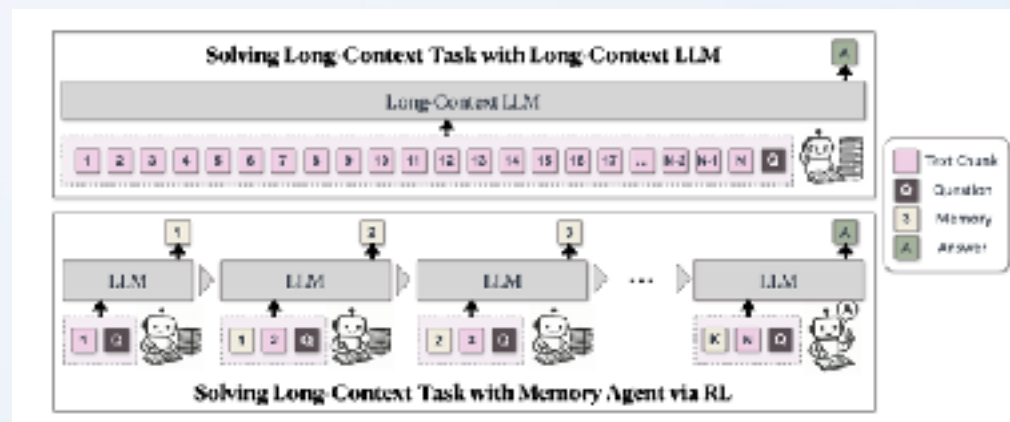
Limitation: AgentLoop chat history append only

- How to compress context?
- How to handle extreme long-context?

Some awesome works based on verl!



[microsoft/agent-lightning](https://microsoft.github.io/agent-lightning/)



[BytedTsinghua-SIA/MemAgent](https://bytedance.github.io/SIA/MemAgent/)

📍 北京

QCon

全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AI Ops
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

📍 北京

AICon

全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

📍 上海

QCon

全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月

5月

6月

8月

10月

12月

📍 上海

AICon

全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

📍 深圳

AICon

全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

📍 北京

AICon

全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LM Ops
- 具身智能

极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播



参会咨询



查看会议

THANKS

大模型正在重新定义软件

Large Language Model Is Redefining The Software

