SWE-InfraBench: Evaluating Language Models on Cloud Infrastructure Code

Natalia Tarasova, Enrique Balp-Straffon, Aleksei Iancheruk, Yevhenii Sielskyi, Nikita Kozodoi, Liam Byrne, Jack Butler, Dayuan Jiang, Marcin Czelej, Andrew Ang, Yash Shah, Roi Blanco, Sergei Ivanov

Can LLMs Develop Infrastructure-as-Code?

✅ **SWE benchmarks:**

- SWE-Bench (ICLR2024) - 2K+ GitHub issues to be resolved by AI
- SWE-Bench-Verified - OpenAI cleaning on SWE-Bench
- HumanEval -164 manually curated Python coding problems
- CloudEval-YAML - Alibaba benchmark on testing LLMs for YAML

❓ **IaC benchmarks:**

- IaC-Eval – 458 Terraform questions to build from scratch

SWE-InfraBench: 100 Python CDK tasks with unit tests for objective evaluation

Request:

- Full IaC-"donor" repo is passed as context
- Original code is partially masked
- Natural language prompt instructs to add hidden code parts

Evaluation:

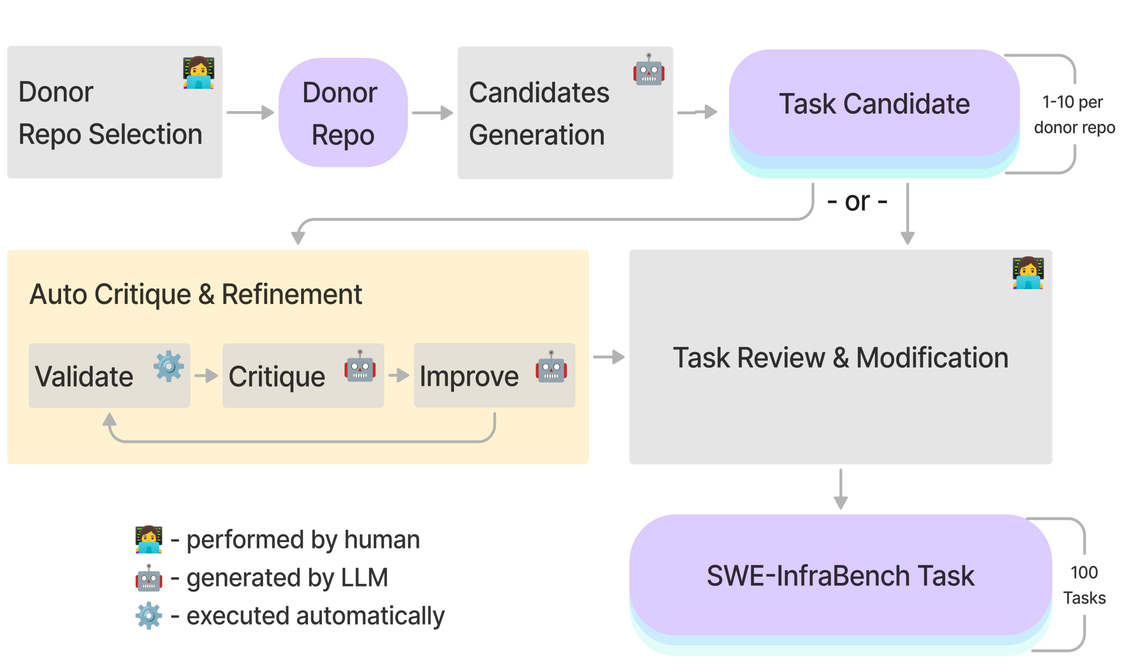
- Model response is integrated into the context codebase
- CloudFormation template is generated. Cloud Deployment is not Required
- Unit tests are run against CF template file to evaluate the solution

Example Task: API Gateway Integration with EventBridge

```
{
  "task_id": "67ad6ef1-fb3e-45e6-b5e1-83ae385528b5",
  "entry_point": "api-eventbridge-lambda+api_gateway_integration",
  "prompt": "Generate code to create an API Gateway
  ↪  SampleAPI-EventBridge-Multi-Consumer
          that integrates with the event producer with proxy
          ↪  integration.
          Do not add a catch-all route in api routing (use
          ↪  proxy=False).
          Add a resource named 'items' to the root of the API and
          ↪  create a
          POST method for this resource.",
  "cdk_version": "2.178.2",
  "context": {
    "app.py": "#!/usr/bin/env python3\n\nfrom aws_cdk import
    ↪  App\n\nfrom api_eventbridge_lambda.api_eventbridge_lambda
    ↪  import ApiEventBridgeLambdaStack...",
    "api_eventbridge_lambda/api_eventbridge_lambda.py": "from
    ↪  constructs import Construct\nfrom aws_cdk import...",
    "lambda/event_consumer_lambda.py": "import json\nimport
    ↪  logging\n\nlogger = logging.getLogger()...",
    "lambda/event_producer_lambda.py": "import json\nimport
    ↪  boto3\nimport datetime..."
  },
  "canonical_solution": {
    "api_eventbridge_lambda/api_eventbridge_lambda.py": [
      "--- without_solution\n\n+++ with_solution\n\n@@ -102,0 +103,7
      ↪  @@\n\n+        # defines an API Gateway REST API resource
      ↪  backed by our \"atm_producer_lambda\" function.\n\n+
      ↪  api = api_gw.LambdaRestApi(self,
      ↪  'SampleAPI-EventBridge-Multi-Consumer',\n\n+
      ↪  handler=event_producer_lambda,\n\n+
      ↪  proxy=False\n\n+                                  )\n\n+
      ↪  items = api.root.add_resource(\"items\")\n\n+
      ↪  items.add_method(\"POST\")  # POST /items\n"
    ]
  },
  "tests": {
    "test_api_gateway_integration.py": "import aws_cdk as cdk\nfrom
    ↪  aws_cdk.assertions import Template, Match..."
  }
}
```

Content between masking tags is "canonical solution"

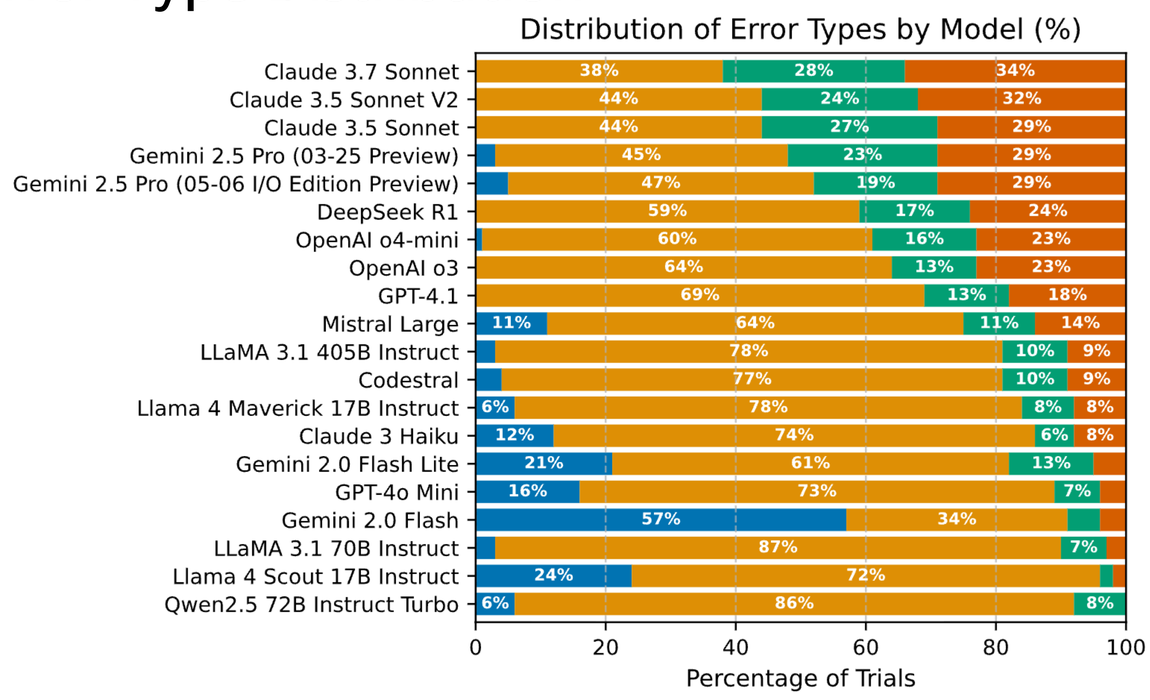
Benchmark Construction



LLMs Benchmarking

Source	Company	Model	Correctness	Generation Success	Passed Tests Share
Proprietary	Anthropic	Claude 3.7 Sonnet †	**34%**	**100%**	**53.1%**
		Claude 3.5 Sonnet V2	32%	**100%**	47%
		Claude 3.5 Sonnet	29%	**100%**	47.9%
		Claude 3 Haiku	8%	88%	10.7%
	Google	Gemini 2.5 Pro (03-25 Preview) †	29%	97%	42.5%
		Gemini 2.5 Pro (05-06 I/O Edition Preview) †	29%	95%	41.1%
		Gemini 2.0 Flash Lite	5%	79%	11.5%
		Gemini 2.0 Flash	4%	43%	6.9%
	OpenAI	OpenAI o3 †	23%	**100%**	30.8%
		OpenAI o4-mini †	23%	99%	32.1%
		GPT-4.1	18%	**100%**	26.4%
		GPT-4o Mini	4%	84%	7%
Open-Source	DeepSeek	DeepSeek R1 †	24%	**100%**	34.9%
	Mistral	Mistral Large	14%	89%	20.3%
		Codestral	9%	96%	15%
	Meta	LLaMA 3.1 405B Instruct	9%	97%	13%
		LLaMA 3.1 70B Instruct	3%	97%	7.7%
		LLaMA 4 Maverick 17B Instruct	8%	94%	13%
		LLaMA 4 Scout 17B Instruct	2%	76%	2.4%
	Alibaba	Qwen2.5 72B Instruct Turbo	0%	94%	2.7%

Error Type Distribution



Consistency Analysis

Model	Average Correctness	pass@1	pass@2	pass@5
Claude 3.7 Sonnet	**28.8%**	**34%**	36%	41%
Gemini 2.5 Pro (03-25 Preview)	26.4%	28%	**38%**	**47%**
DeepSeek R1	24.6%	24%	33%	46%
OpenAI o3	22.8%	23%	30%	45%
LLaMA 3.1 405B Instruct	6.8%	8%	11%	13%

Agents Benchmarking

Model	Correctness (↑)				
	One-Turn	Two-Turn V_L	Two-Turn V_H	Two-Turn + RAG V_L	Two-Turn + RAG V_H
Claude 3.7 Sonnet	34.0%	44.0%	64.0%	51.0%	60.0%
Claude 3.5 Sonnet V2	32.0%	52.0%	56.0%	52.0%	65.0%
Gemini 2.5 Pro (03-25 Preview)	29.0%	41.0%	40.0%	41.0%	33.0%
DeepSeek R1	24.0%	40.0%	43.0%	45.0%	39.0%
GPT-4.1 (2025-04-14)	18.0%	40.0%	48.0%	46.0%	26.0%
Mistral Large	14.0%	21.0%	23.0%	14.0%	17.0%
LLaMA 4 Maverick 17B Instruct	8.0%	15.0%	18.0%	21.0%	21.0%

🔄 Two-Turn Agent

Results from the 1st attempt are returned to the agent for the 2nd try

🔍 Tests Verbosity Level

Low (basic error messages) and High (full-traceback) verbosity levels tried

ℹ️ RAG Enhancement

MCP server with AWS docs provided as a data source to the agent

Get the dataset

Download PDF