Centro de Informática

UFPE

Evaluating Feedback Mechanisms in Code Generation

Alunos: Cleber Silva
Felipe Almeida
Pierre Oriá
Sarah Melo

Sumário

Introdução

Introdução

A ideia central é utilizar um mecanismo de feedback iterativo para melhorar a qualidade do código gerado. O pipeline implementado avalia a saída do modelo e, em caso de erro, fornece um retorno para orientar a geração de uma nova resposta.

Introdução

1. Implementação de um pipeline automatizado para avaliação e refinamento de código gerado por LLMs.

2. O processo segue uma abordagem iterativa, onde um problema de programação é fornecido ao modelo, e sua solução é testada e validada automaticamente.

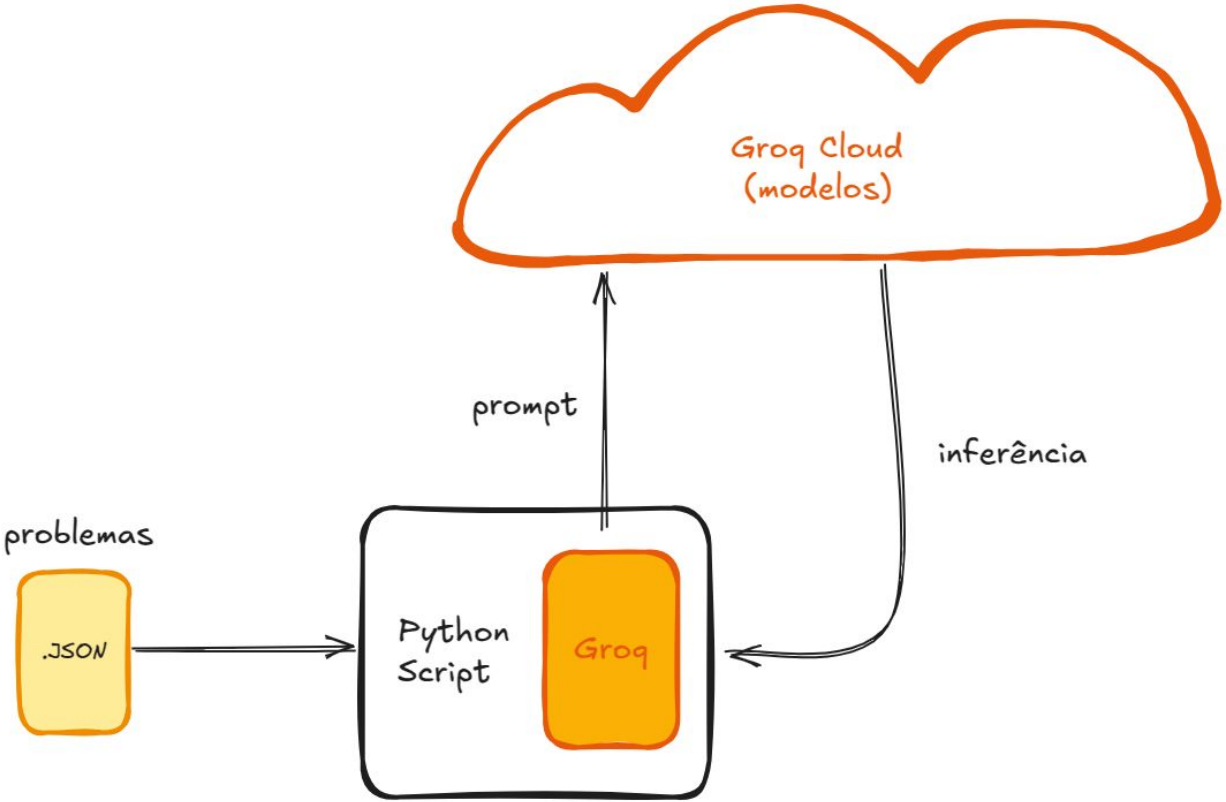
3. Caso a solução seja incorreta, os erros (**stack trace**) identificados são usados como feedback para gerar uma nova versão corrigida do código.

Framework de avaliação

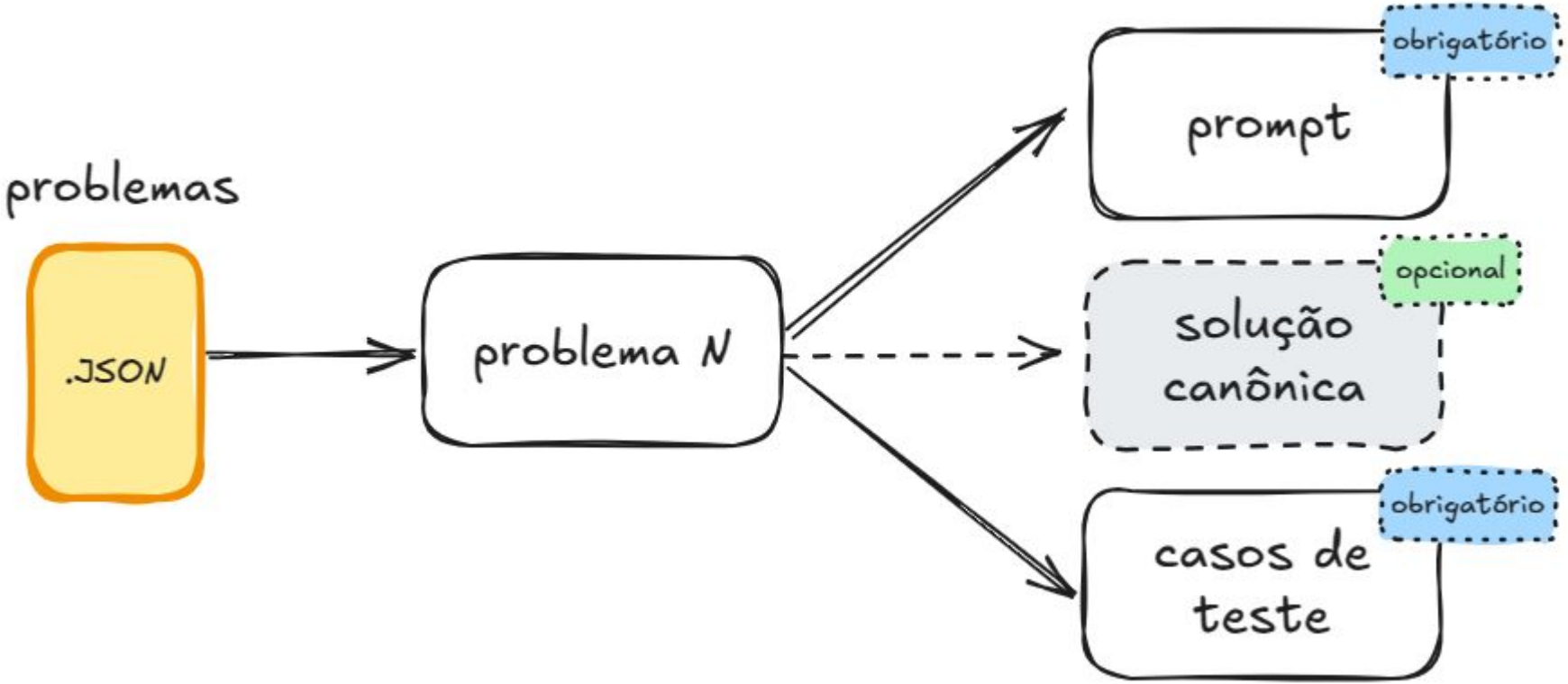
Framework de Avaliação - Entrada


```
{
  "task_id": "hamming_distance",
  "prompt": "def hamming_distance(N: int, S: str, T: str) -> int: \n\"\"\"\n    You are given a positive integer N and two strings S and T, each of length N and consisting of lowercase English letters. \n    Find the Hamming distance between S and T. That is, return the number of positions i (1-indexed) where S and T differ.\n\n    Constraints:\n      - 1 \u2264 N \u2264 100\n      - S and T are strings of length N containing only lowercase English letters. \n\"\"\"",
  "test": "def test():\n    assert hamming_distance(6, 'abcarc', 'agcahc') == 2\n    assert hamming_distance(7, 'atcoder', 'contest') == 7 \n    assert hamming_distance(8, 'chokudai', 'chokudai') == 0 \n    assert hamming_distance(10, 'vexknuampx', 'vzxikuamlx') == 4"
}
```

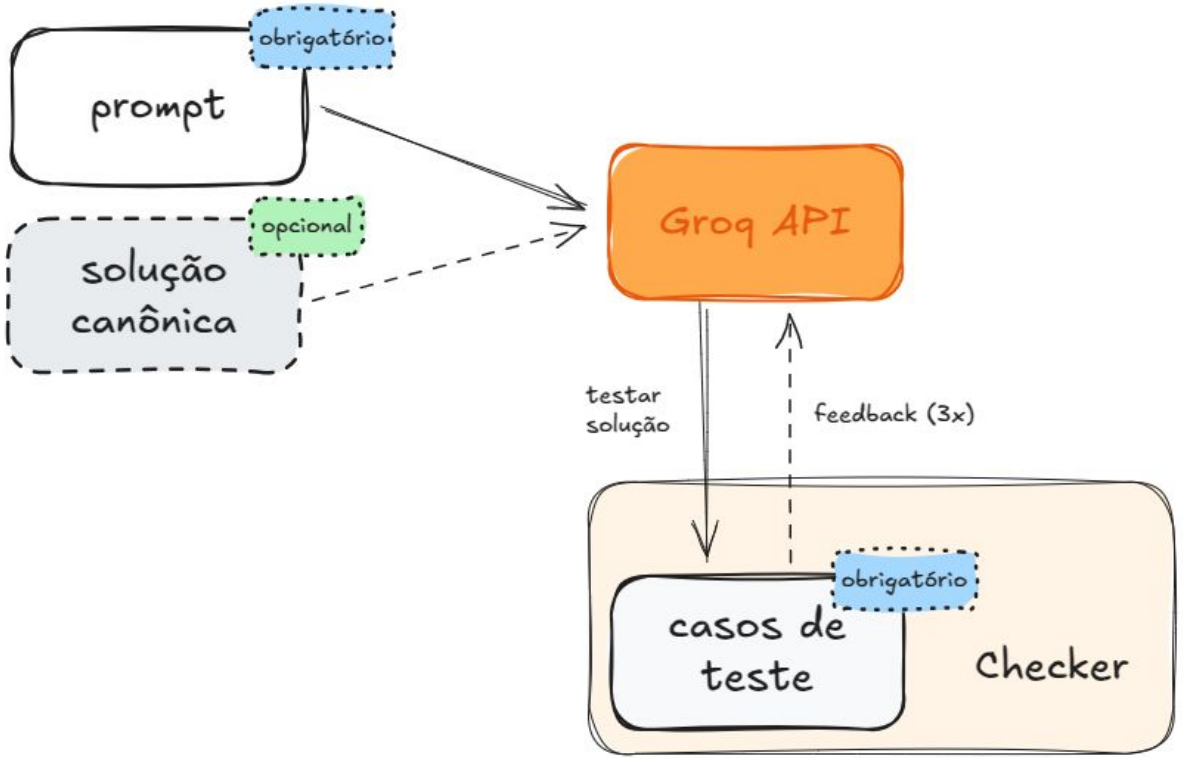
Framework de Avaliação - Pipeline



Framework de Avaliação - Pipeline



Framework de Avaliação - Pipeline



Framework de Avaliação - Questões

Fontes

- AtCoder
- CSES

Framework de Avaliação - Modelos

Meta

llama3-8b-8192

llama3-70b-8192

llama-3.3-70b-versatile

llama-3.1-8b-instant

Qwen

qwen-2.5-coder-32b

Google DeepMind

gemma2-9b-it

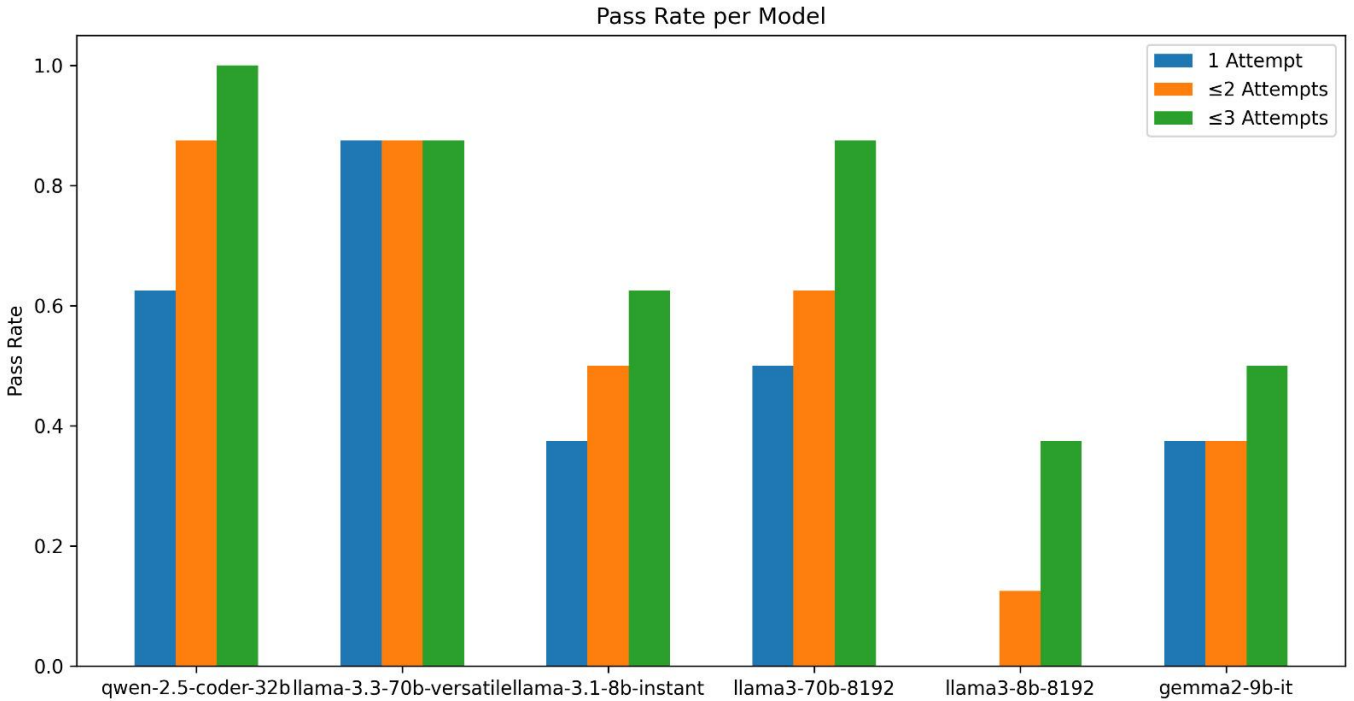
- Propósito geral
- Geração de código

Framework de Avaliação - Modelos

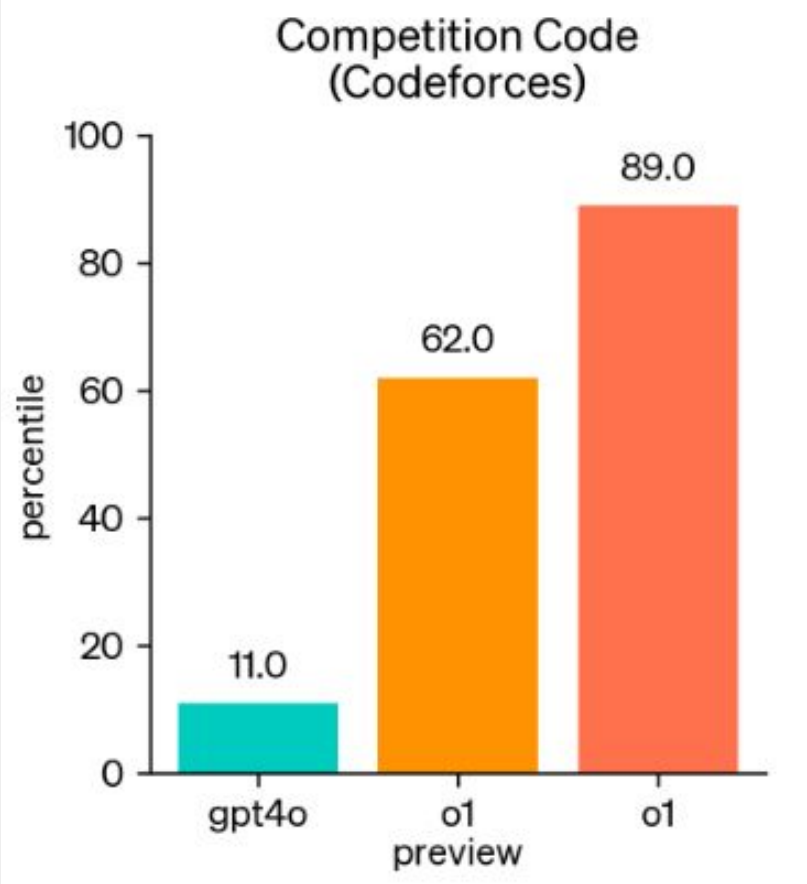
Modelo	Número de parâmetros
llama3-8b-8192	9 bilhões
llama3-70b-8192	70 bilhões
llama-3.3-70b-versatile	70 bilhões
llama-3.1-8b-instant	8 bilhões
qwen-2.5-coder-32b	32 bilhões
gemma2-9b-it	9 bilhões

Resultados

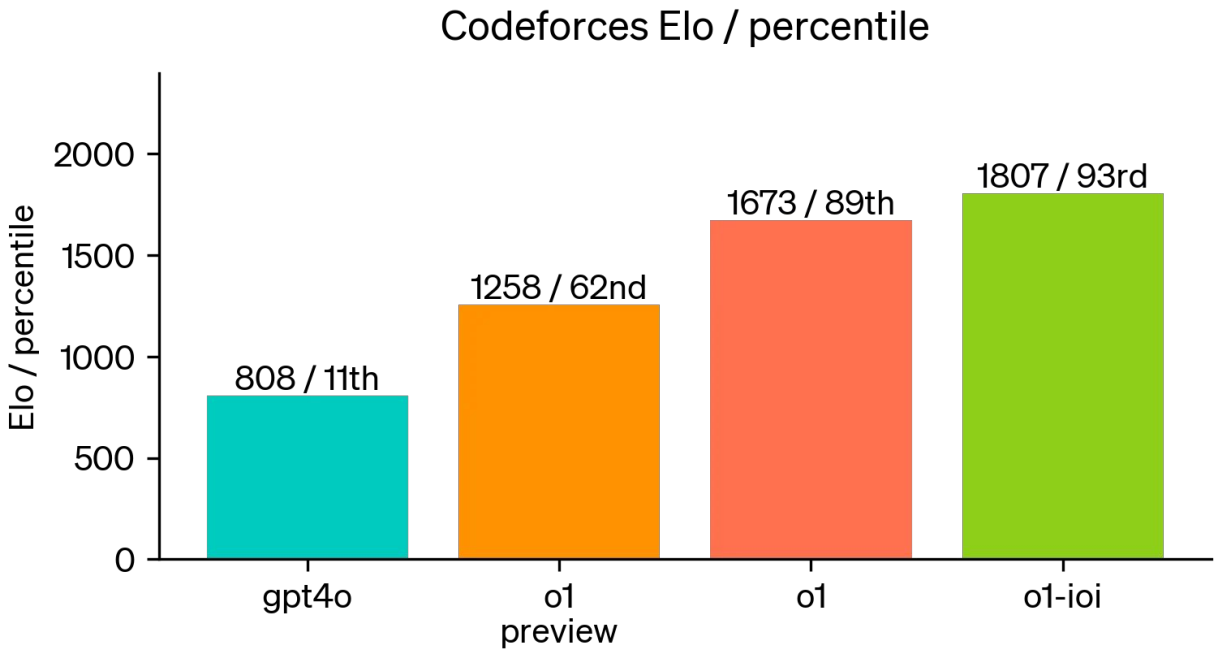
Resultados


Pass Rate per Model
Pass Rate
1.0
0.8
0.6
0.4
0.2
0.0
1 Attempt
≤2 Attempts
≤3 Attempts
qwen-2.5-coder-32b
llama-3.3-70b-versatile
llama-3.1-8b-instant
llama3-70b-8192
llama3-8b-8192
gemma2-9b-it

pass@1 o1


Competition Code
(Codeforces)
100
80
60
40
20
0
percentile
11.0
62.0
89.0
gpt4o
o1
preview
o1

Resultados - Comparativo o1-preview, o1 & o1-ioi



Trabalhos futuros

Trabalhos futuros

Utilizar mais modelos

- Focar em modelos de geração de código
- Utilizar modelos com maior número de parâmetro

Utilizar loops de feedback mais longos

- Pelos resultados, percebe-se que o número de feedbacks impacta na qualidade da resposta da LLM

Limitações

Limitações

LLMs disponíveis com poucos parâmetros

- O maior dos modelos utilizados por nós tinha 70 bilhões de parâmetros.
  - GPT-3, por exemplo, tinha 170 bilhões de parâmetros

Muito obrigado!

https://portal.cin.ufpe.br/

@cinufpe

Centro de Informática UFPE

Centro de Informática UFPE

@CInUFPE

Centro de Informática

UFPE