



Magocoder: Source Code Is All You Need

Empowering Code Generation with OSS-INSTRUCT

 [ise-uiuc/magocoder](https://github.com/ise-uiuc/magocoder)  [ise-uiuc](https://github.com/ise-uiuc)  [Magocoder_AI](https://twitter.com/Magocoder_AI)

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, Lingming Zhang

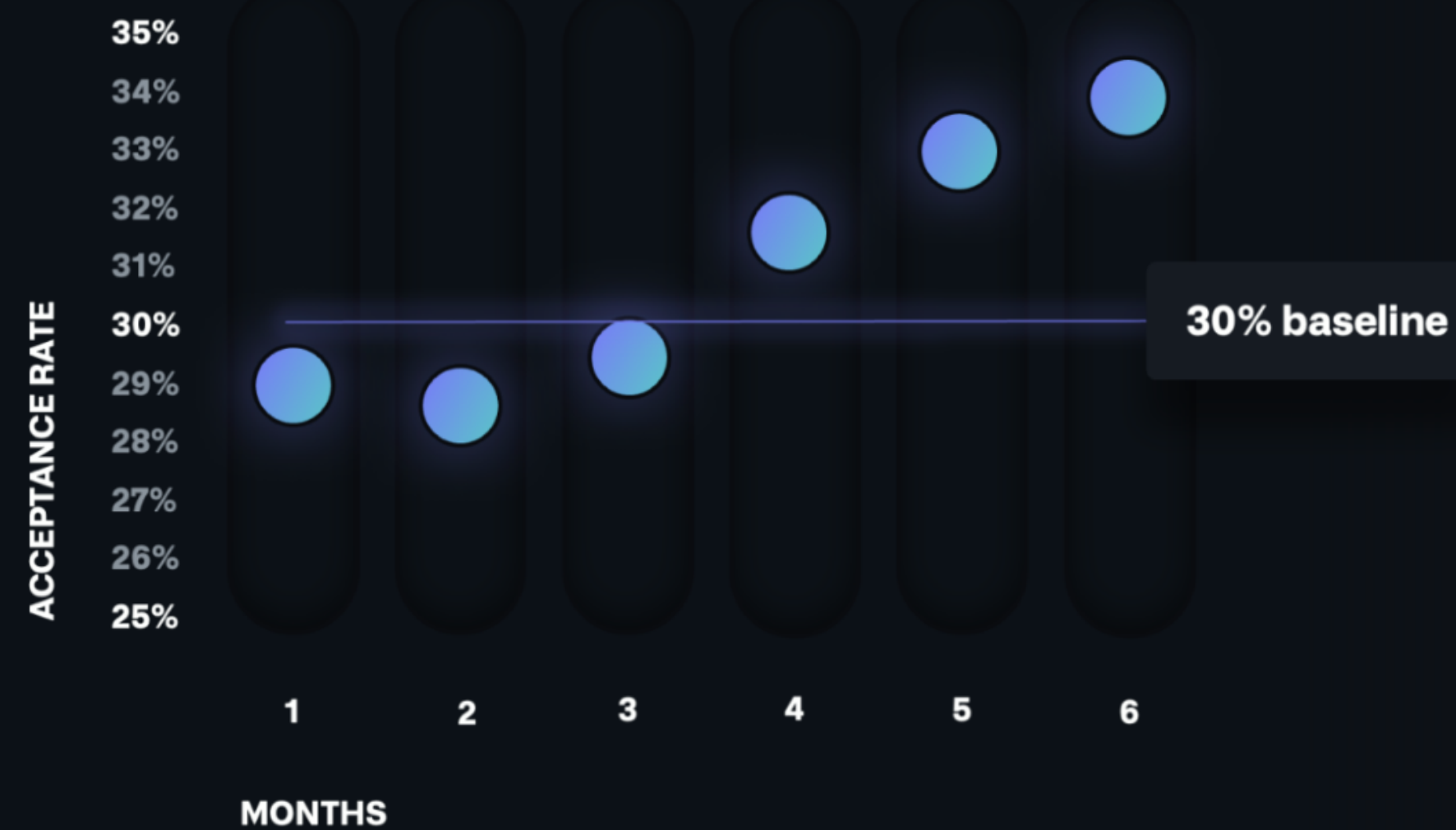
 **Yuxiang Wei**, 2nd Year PhD Student, University of Illinois at Urbana-Champaign

Jan 23, 2024

The image displays a variety of AI models, including:

- CodeT5
- AlphaCode
- CodeLlama
- ChatGPT
- Codex
- CodeGen2
- StarCoder
- CodeGen
- Phind
- InCoder
- MagiCoder
- GPT-4
- PaLM
- Phi
- Gemini
- WizardCoder
- DeepSeek-Coder
- CodeT5+
- ...

Copilot's impact increases over time



"GitHub Copilot has been activated by more than **one million developers** and adopted by over **20,000 organizations**. It has generated over **three billion accepted lines of code**, and is the world's most widely adopted AI developer tool."

<https://github.blog/2023-06-27-the-economic-impact-of-the-ai-powered-developer-lifecycle-and-lessons-from-github-copilot/>

How are foundation code models built?

StarCoder (15.5B)

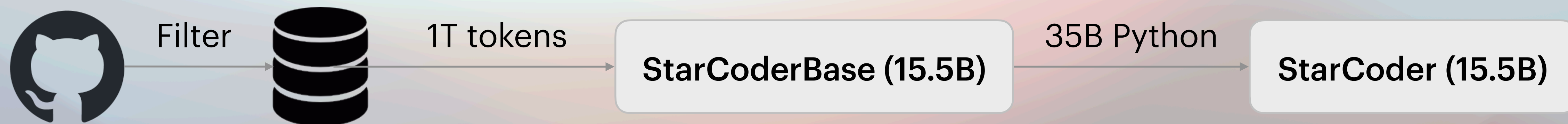
Collect massive code from GitHub: The Stack

6TB permissive code data

Filtering and cleaning: starcoderdata

783GB of code in 86 programming languages

StarCoderBase (1T training tokens); StarCoder (+35B Python tokens)

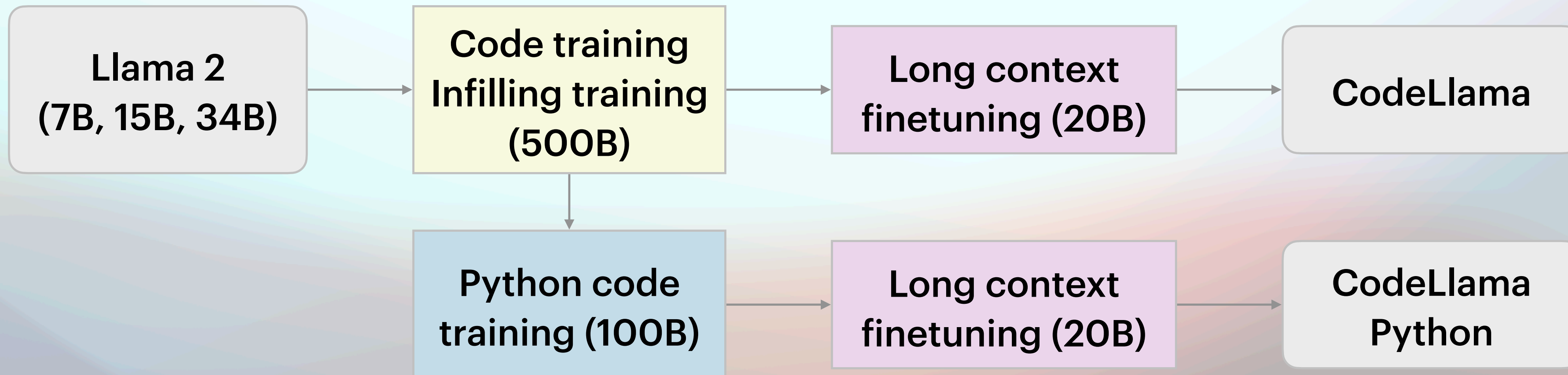


CodeLlama (7B, 15B, 34B)

Continue pretraining Llama 2 on code and natural language + code data

CodeLlama (500B tokens)

CodeLlama-Python (+100B Python tokens)



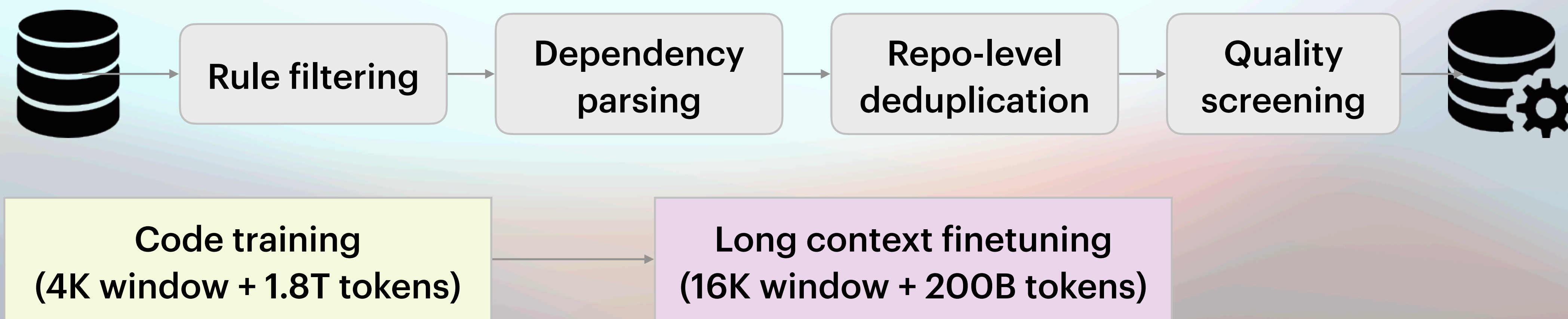
DeepSeek-Coder (1.3B, 6.7B, 33B)

Adopts Llama 2 architecture

Same training data as StarCoder, but

Concatenates dependent files to form a single training example

Trained on 2T tokens



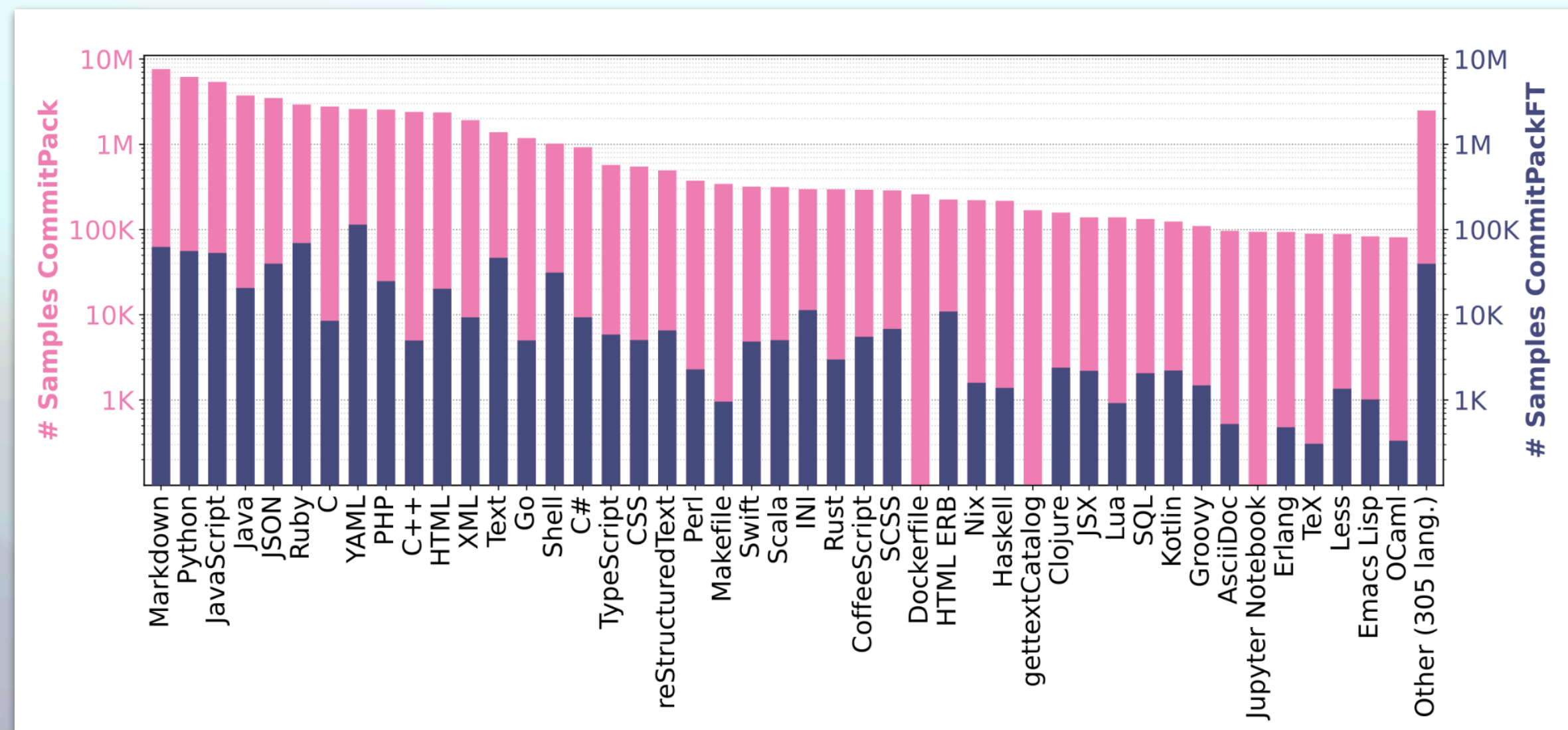
Instruction tuning boosts LLM performance

Collecting real-world instructions

OctoCoder

Uses GitHub commits to instruction-tune StarCoder

Achieves **46.2** HumanEval pass@1 (**13pp higher** than the base StarCoder)



Collecting closed-domain synthetic data

CodeLlama-Instruct

Applies Llama 2 to generate problems and CodeLlama for solutions

Modest Impact: **4pp** improvements over 7B and **2pp** over 13B base models (results for the 34B model not reported). **42.7** HumanEval pass@1 (13B)

```
[INST] Write 50 programming interview questions of easy and medium complexity. Provide questions on a diverse range of subjects, and make sure no two questions are alike. Make sure the problems can be solved with a single standalone Python function using standard libraries. [/INST]
```

1. Write a function that finds the maximum depth of list nesting in a given list.
 2. Given an integer array nums, rotate the array to the right by k steps, where k is non-negative.
 3. Write a function that gets the musical pitch that is closest to a given frequency in Hz. A pitch should consist of a letter, possibly a # sign, and an octave number.
 4. Write a function that removes any sequences of whitespace that are between numbers in an input string.
- ...

Collecting open-domain synthetic data

Leverages ChatGPT's few-shot capability to generate synthetic instruction-response pairs

```
You are asked to come up with a set of 20 diverse code generation task instructions. These task instructions will be given to a GPT model and we will evaluate the GPT model for completing the instructions.
```

```
Here are the requirements:
```

```
1. Try not to repeat the verb for each instruction to maximize diversity.
```

```
...
```

```
List of 20 tasks:
```

```
{seed_tasks}
```

SELF-INSTRUCT (Relies on 21 seed tasks as few-shot examples)

Collecting open-domain synthetic data

You are asked to come up with a set of 20 diverse code generation task instructions. These task instructions will be given to a GPT model and we will evaluate the GPT model for completing the instructions.

Here are the requirements:

1. Try not to repeat the verb for each instruction to maximize diversity.

...

List of 20 tasks:

`{seed_tasks}`

SELF-INSTRUCT (Relies on 21 seed tasks as few-shot examples)

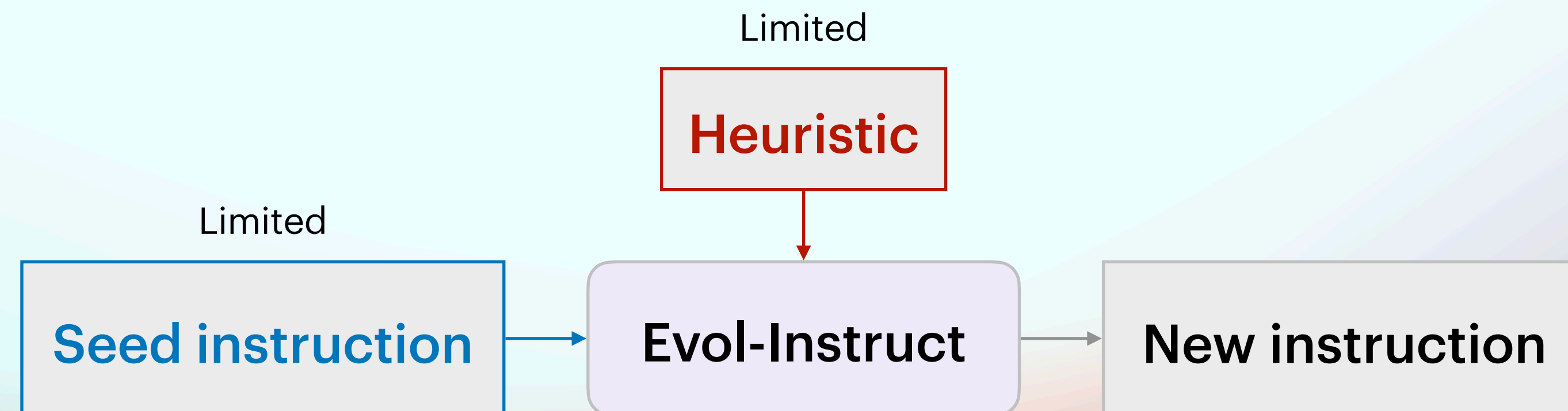
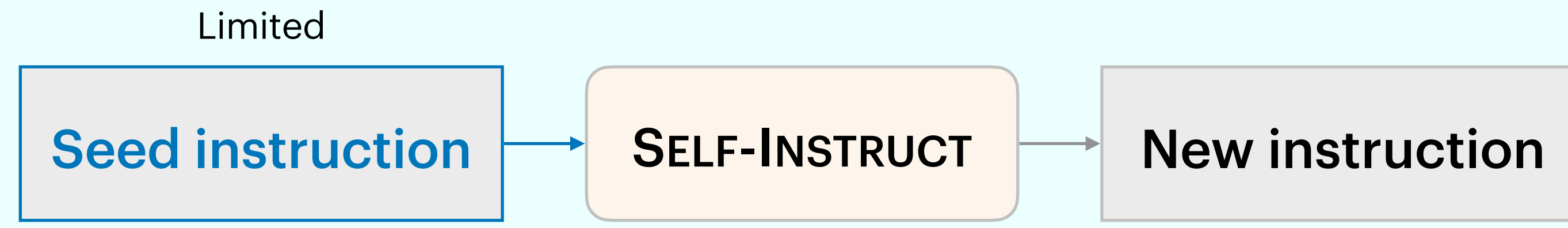
Please increase the difficulty of the given programming test question a bit.

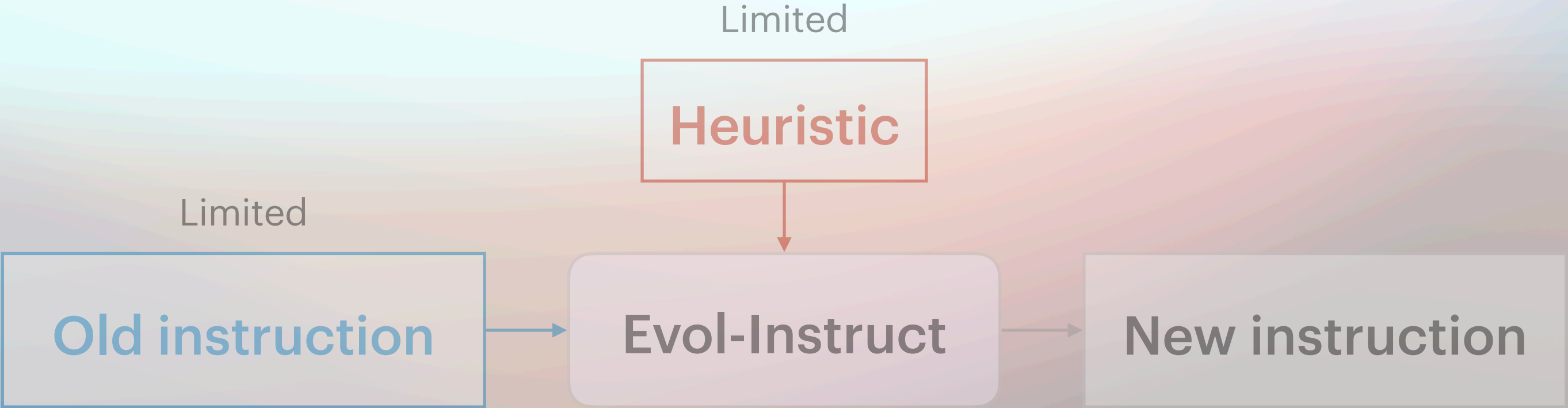
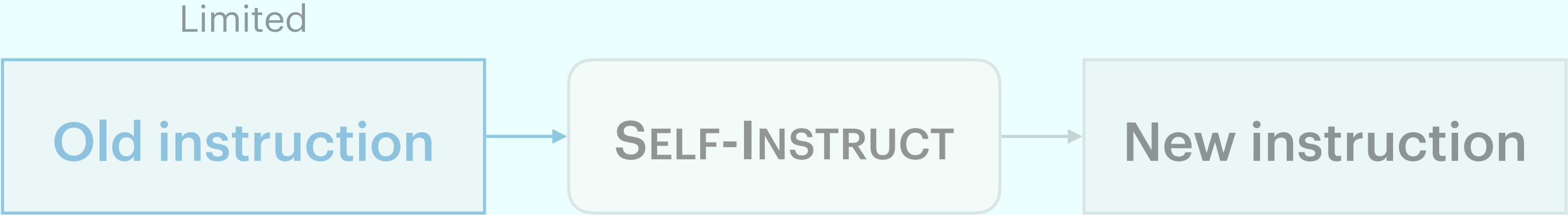
You can increase the difficulty using, but not limited to, the following methods:

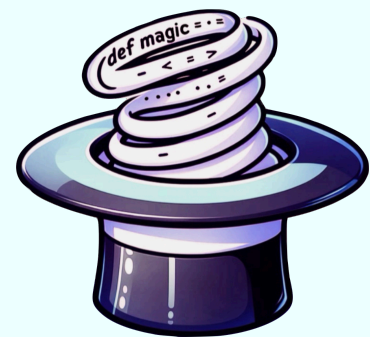
`{heuristic}`

`{seed_question}`

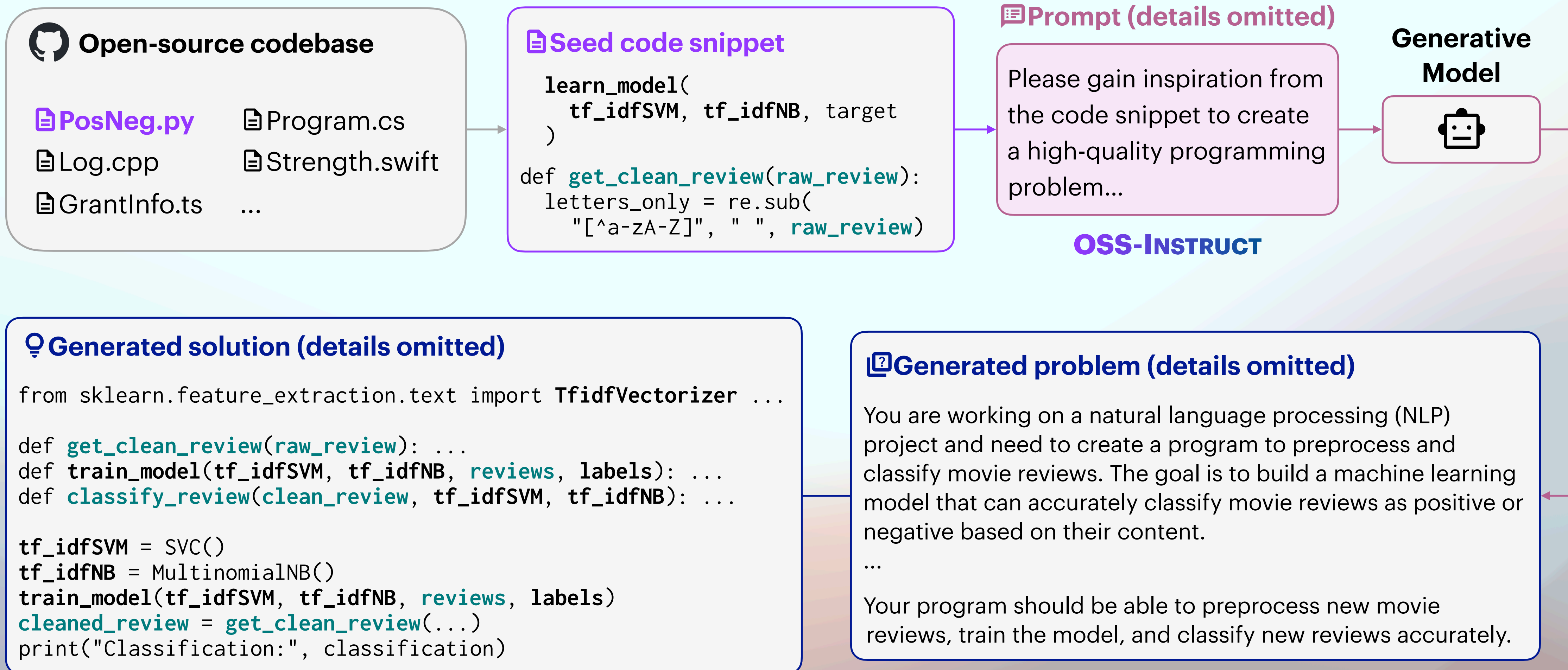
Evol-Instruct (Evolve seed questions using 5 heuristics)







Magocoder: Source Code Is All You Need



OSS-INSTRUCT: data collection






Codebase: StarCoder training data (starcoderdata)

783GB of code in 86 programming languages

Includes 54GB GitHub Issues + 13GB Jupyter notebooks

32GB of GitHub commits

Open-source codebase

 PosNeg.py  Program.cs
 Log.cpp  Strength.swift
 GrantInfo.ts ...

Seed code snippet

```
learn_model(  
    tf_idfSVM, tf_idfNB, target  
)  
  
def get_clean_review(raw_review):  
    letters_only = re.sub(  
        "[^a-zA-Z]", " ", raw_review)
```







OSS-INSTRUCT: data collection

Seed snippet collection

Randomly extract 1–15 consecutive lines from each document

80K in total (75K after cleaning): 40K from Python, and 5K from each of C++, Java, TypeScript, Shell, C#, Rust, PHP, and Swift respectively

Open-source codebase

 PosNeg.py  Program.cs
 Log.cpp  Strength.swift
 GrantInfo.ts ...

Seed code snippet

```
learn_model(  
    tf_idfSVM, tf_idfNB, target  
)  
  
def get_clean_review(raw_review):  
    letters_only = re.sub(  
        "[^a-zA-Z]", " ", raw_review)
```

OSS-INSTRUCT: prompt design

Ask ChatGPT (gpt-3.5-turbo) to get inspired from the code

You are exceptionally skilled at crafting high-quality programming problems and offering precise solutions.

Please gain inspiration from the following random code snippet to create a high-quality programming problem. Present your output in two distinct sections: **[Problem Description]** and **[Solution]**.

Code snippet for inspiration:

```
```\n{code}\n```
```

Guidelines for each section:

**[Problem Description]:** This should be **\*\*completely self-contained\*\***, providing all the contextual information one needs to understand and solve the problem. Assume common programming knowledge, but ensure that any specific context, variables, or code snippets pertinent to this problem are explicitly included.

**[Solution]:** Offer a comprehensive, **\*\*correct\*\*** solution that accurately addresses the [Problem Description] you provided.

## Seed: method definition

```
render() {
```

## Problem

Your task is to complete the `render` method to generate the rendered shape as a string...

## Code

```
class ShapeRenderer {
 constructor(vertices) {
 this.vertices = vertices;
 }
 render() {
 let renderedShape = "";
 for (let i = 0; i < this.vertices.length; i++) {
 const vertex = this.vertices[i];
 renderedShape += `(${vertex.x}, ${vertex.y})`;
 if (i < this.vertices.length - 1) {
 renderedShape += " - ";
 }
 }
 return renderedShape;
 }
}
```

## Seed: shell script

```
python3 makeErrorFile.py data/test_dataset_14 14
```

## Problem

Create a Python program that generates an **error file** based on a given **dataset**...

## Code

```
def generate_error_file(dataset_file, ...):
 error_lines = []
 with open(dataset_file, 'r') as file:
 for line in file:
 ...
 with open(error_file_name, 'w') as error_file:
 for error_line in error_lines:
 error_file.write(error_line + '\n')
if __name__ == "__main__":
 if len(sys.argv) != 3:
 print("Usage: ...")
 else:
 dataset_file = sys.argv[1]
 dataset_number = sys.argv[2]
 generate_error_file(...)
```

### 📄 Seed: library imports

```
import numpy as np
import gym_electric_motor as gem
import matplotlib.pyplot as plt
```

### 🔍 Problem

Create a **reinforcement learning agent** to control an electric motor using the OpenAI **Gym** environment...

### 🔗 Code

```
import numpy as np
import gym_electric_motor as gem
...
env = gem.make("DcSeriesCont-v1")
class DQNAgent:
 def __init__(self, state_dim, action_dim): ...
 def build_model(self): ...
 def act(self, state): ...
 def train(self, state, action, reward, ...): ...
...
for episode in range(episodes):
 state = env.reset()
 state = np.reshape(state, [1, state_dim])
 ...
```

### 📄 Seed: class signature

```
@SpringBootApplication
@Import({ AxonConfig.class })
public class AxonbankApplication {
 public static void main(String[] args) {
```

### 🔍 Problem

Create a simple Java **Spring Boot application** for a **banking system**...

### 🔗 Code

```
import org.axonframework.commandhandling.CommandBus;
import org.axonframework.config.Configuration;
...
@SpringBootApplication
@Import({ AxonConfig.class })
public class AxonbankApplication {...}
public class BankAccount {...}
public class CreateAccountCommand {...}
public class DepositFundsCommand {...}
public class WithdrawFundsCommand {...}
public class AccountCreatedEvent {...}
public class FundsDepositedEvent {...}
public class FundsWithdrawnEvent {...}
```



### Seed: code statements

```
cutoff_range = np.ptp(cutoffs)
if cutoff_range == 0: cutoff_range = 1
cutoff_colors = plt.get_cmap('plasma')(
 (cutoffs - np.min(cutoffs)) / cutoff_range
)
```

### Problem

Implement a function that calculates the color values for a given set of **cutoff** values based on a specified **color map**...

### Code

```
import numpy as np
import matplotlib.pyplot as plt

def calculate_cutoff_colors(cutoffs, cmap_name):
 cutoff_range = np.ptp(cutoffs)
 if cutoff_range == 0:
 cutoff_range = 1
 cmap = plt.get_cmap(cmap_name)
 normalized_cutoffs = ...
 cutoff_colors = ...
 return cutoff_colors
```

### Seed: comments

```
Set degrees
```

### Problem

Implement a Python class that represents a **temperature in degrees**...

### Code

```
class TemperatureConverter:
 def __init__(self): ...
 def set_celsius(self, degrees): ...
 def set_fahrenheit(self, degrees): ...
 def set_kelvin(self, degrees): ...
 def get_celsius(self): ...
 def get_fahrenheit(self): ...
 def get_kelvin(self): ...
 def convert_to(self, unit):
 if unit == 'C':
 return self.get_celsius()
 elif unit == 'F':
 return self.get_fahrenheit()
 elif unit == 'K':
 return self.get_kelvin()
 ...
```

# Training Magicoder and Magicoder-S

**Base models: CodeLlama-Python-7B and DeepSeek-Coder-6.7B-Base**

Base model + 75K OSS-INSTRUCT 📌 Magicoder

Generated by gpt-3.5-turbo-1106

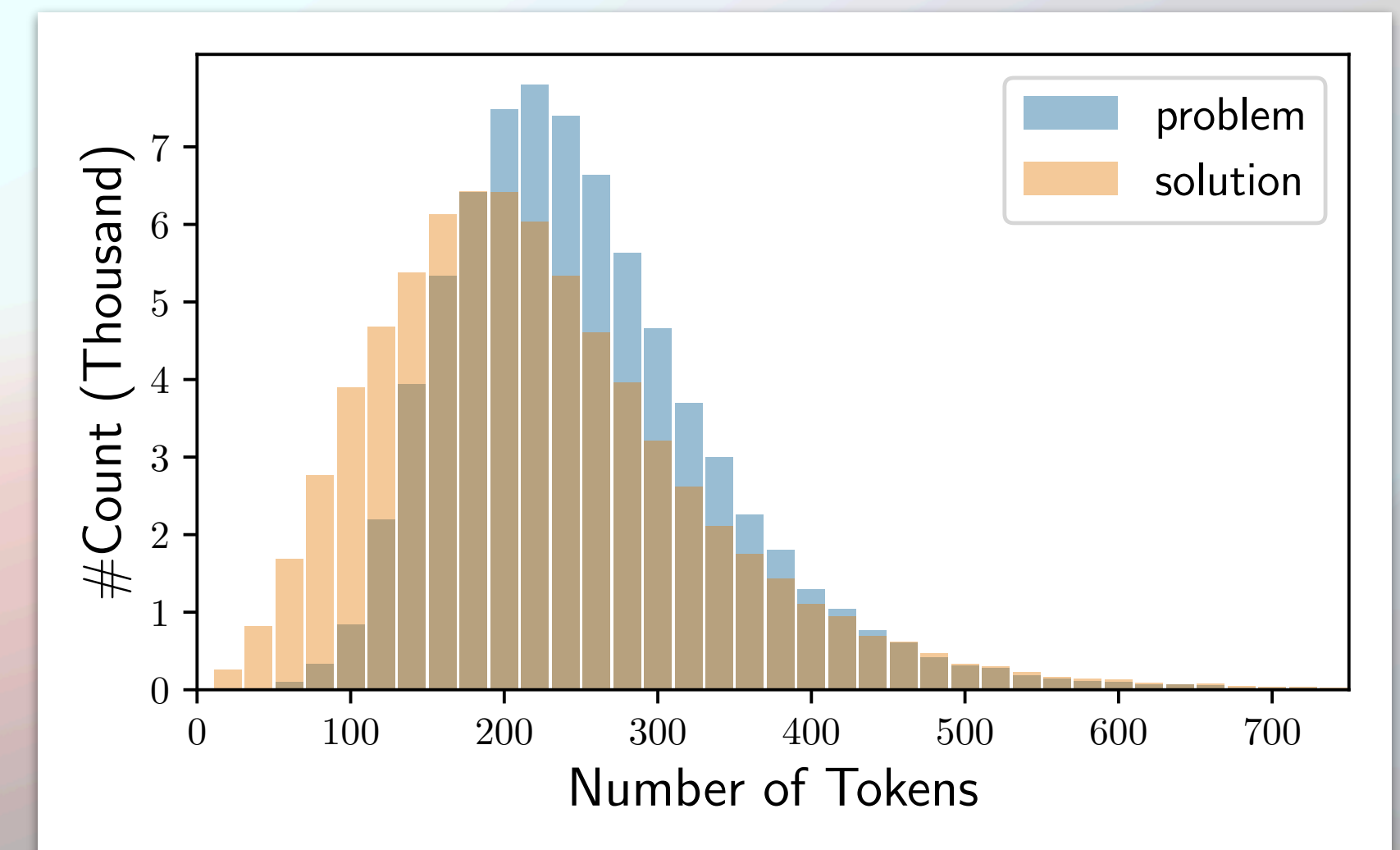
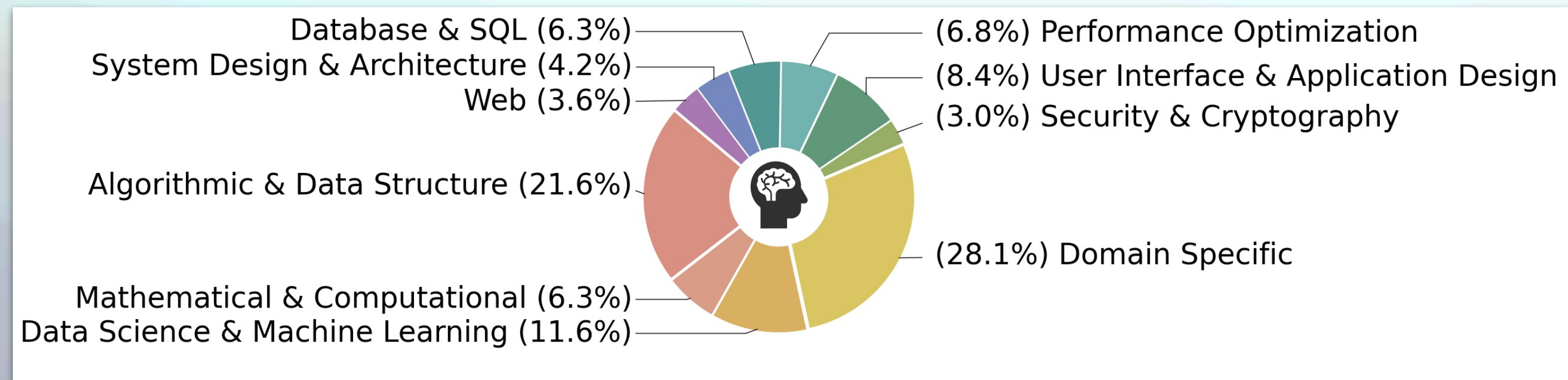
Magicoder + 110K Evol-Instruct 📌 Magicoder-S

Decontaminated from evol-codealpaca-v1, an open-source reproduction of Evol-Instruct generated using gpt4

# Overview of OSS-INSTRUCT dataset

**Problem types:** categorized using the SOTA instruction-tuned embedding model INSTRUCTOR with manually designed queries

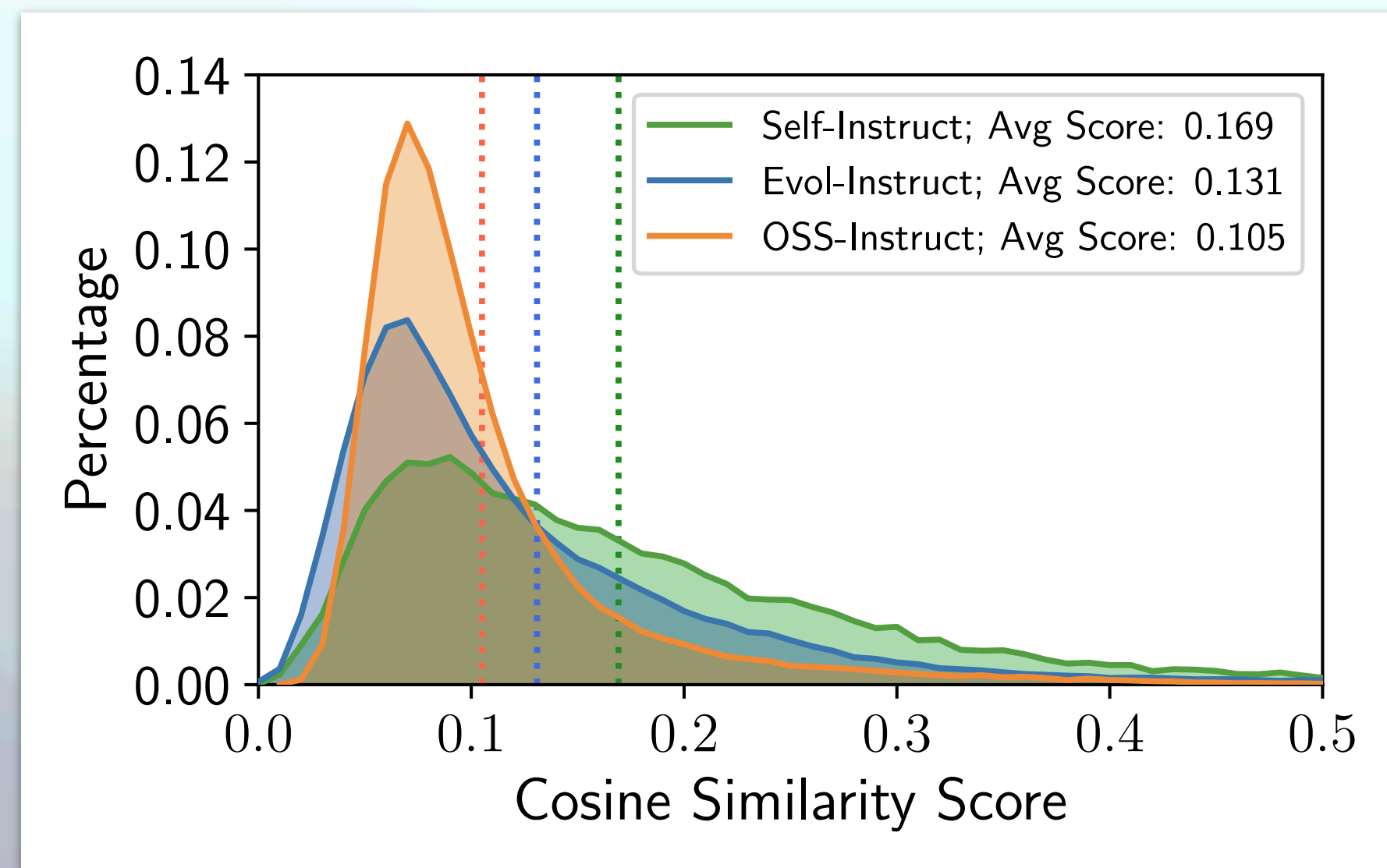
**Length distribution** of the generated problems and solutions



# Similarity with HumanEval

Measured using TF-IDF embedding

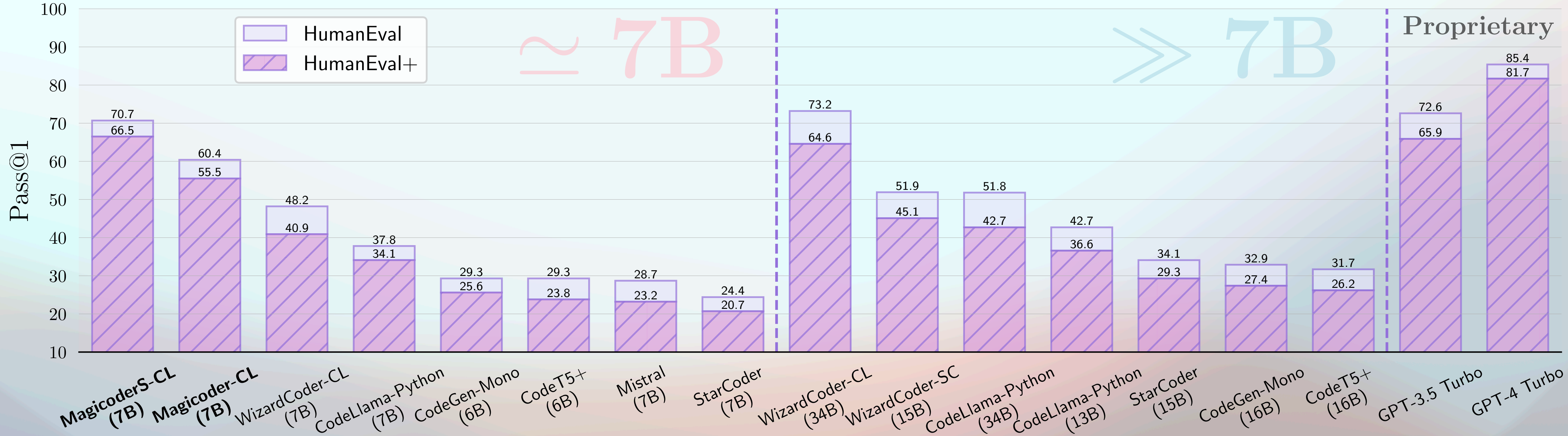
Similarity to HumanEval: OSS-INSTRUCT < Evol-Instruct < Self-Instruct





# Pass@1 results on HumanEval

## CodeLlama-Python 7B as the base model



# Pass@1 results on HumanEval

## DeepSeek-Coder 7B as the base model

Model (size)	#Training tokens	HumanEval+
DeepSeek-Coder Base (6.7B)	2T	39.6
DeepSeek-Coder Instruct (33B)	+2B	72.6
DeepSeek-Coder Instruct (6.7B)	+2B	70.1
<b>Magocoder-DS (6.7B)</b>	<b>+90M</b>	<b>60.4</b>
<b>Magocoder-S-DS (6.7B)</b>	<b>+240M</b>	<b>70.7</b>

More evaluation 🙌 EvalPlus Leaderboard (<https://evalplus.github.io>)

# Multi-lingual evaluation on MultiPL-E

Magocoder-CL significantly outperforms the base model

Magocoder-S-CL (**7B**) reaches WizardCoder-CL (**34B**)

Model (Size)	Java	JavaScript	C++	PHP	Swift	Rust
CodeLlama-Python (34B)	40.2	41.7	41.4	40.4	35.3	38.7
<b>WizardCoder-CL (34B)</b>	<b>44.9</b>	<b>55.3</b>	<b>47.2</b>	<b>47.2</b>	<b>44.3</b>	<b>46.2</b>
WizardCoder-SC (15B)	35.8	41.9	39.0	39.3	33.7	27.1
CodeLlama-Python (7B)	29.1	35.7	30.2	29.0	27.1	27.0
Magocoder-CL (7B)	36.4	45.9	36.5	39.5	33.4	30.6
<b>Magocoder-S-CL (7B)</b>	<b>42.9</b>	<b>57.5</b>	<b>44.4</b>	<b>47.6</b>	<b>44.1</b>	<b>40.3</b>

# Impact of the programming language distribution

Training on different languages boosts overall performance

Python data can boost non-Python performance and vice versa

Combining data from both sources achieves the best result

Model (7B)	Finetuning data	Python (HumanEval+)	Others (MultiPL-E)
CodeLlama-Python	-	34.1	29.6
Magocoder-CL	Python (43K)	47.6	32.7
Magocoder-CL	Others (32K)	44.5	38.3
Magocoder-CL	Both (75K)	55.5	37.8



# OSS-INSTRUCT vs. direct finetuning

Directly finetuning on open-source code may harm performance

75K comment-function pairs data following CodeSearchNet

Data factuality is essential to code instruction tuning

Finetuning Data	HumanEval+	MultiPL-E
Base model w/o finetuning	34.1	29.6
Comment-function pairs (75K)	34.1	24.1
OSS-INSTRUCT (75K)	55.5	37.8



# Summarizing Magicoder

Empower Code Generation with ✨ OSS-Instruct: inspiring LLMs with open-source code snippets for data generation

**1.7K stars** ★ on GitHub (was one of the GitHub trending projects and Hugging Face trending models)

**6K downloads** for model + data on Hugging Face

Try our [Gradio demo](#) for 🐍 game, Othello game, and more!