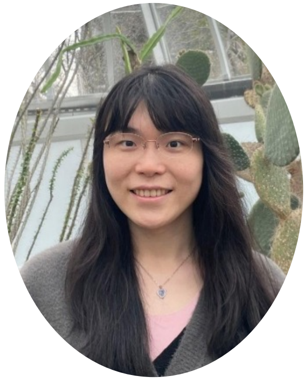
The IEEE/ACM International Conference on Software Engineering (ICSE 2025)

LiCoEval: Evaluating LLMs on License Compliance in Code Generation



Weiwei Xu
xuww@stu.pku.edu.cn
Peking University

Kai Gao
kai.gao@ustb.edu.cn
University of Science and Technology Beijing

Hao He
haohe@andrew.cmu.edu
Carnegie Mellon University

Minghui Zhou
zhmh@pku.edu.cn
Peking University

AI coding tools have been widely adopted but raised growing controversy about copyright

92% Developers

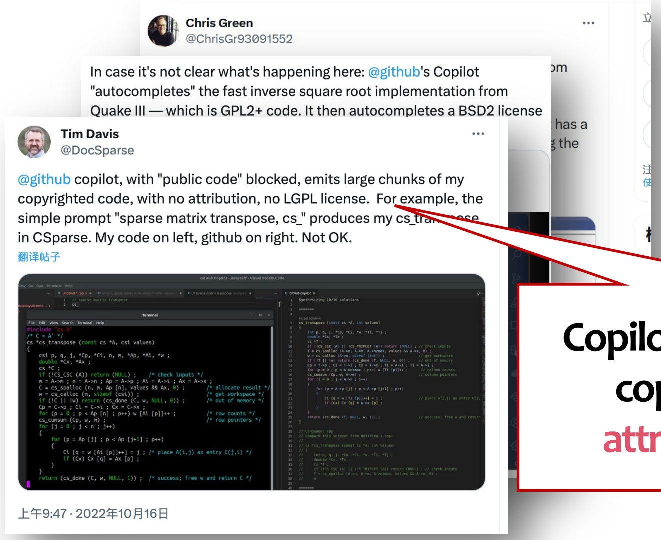
In U.S.

Using AI coding Tools

1/3 Projects

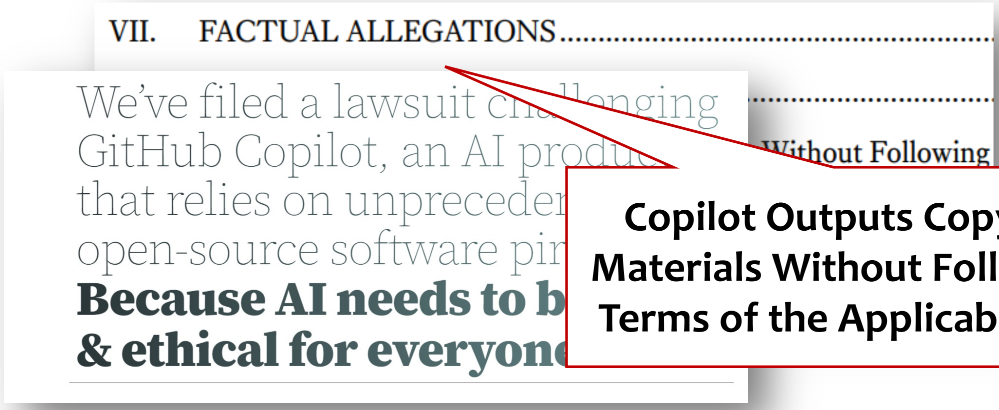
With at least one star

Using GitHub Copilot



Copilot emits large chunks of my copyrighted code, with no attribution, no LGPL license.

Social Media Posts



Copilot Outputs Copyrighted Materials Without Following the Terms of the Applicable License

Lawsuit

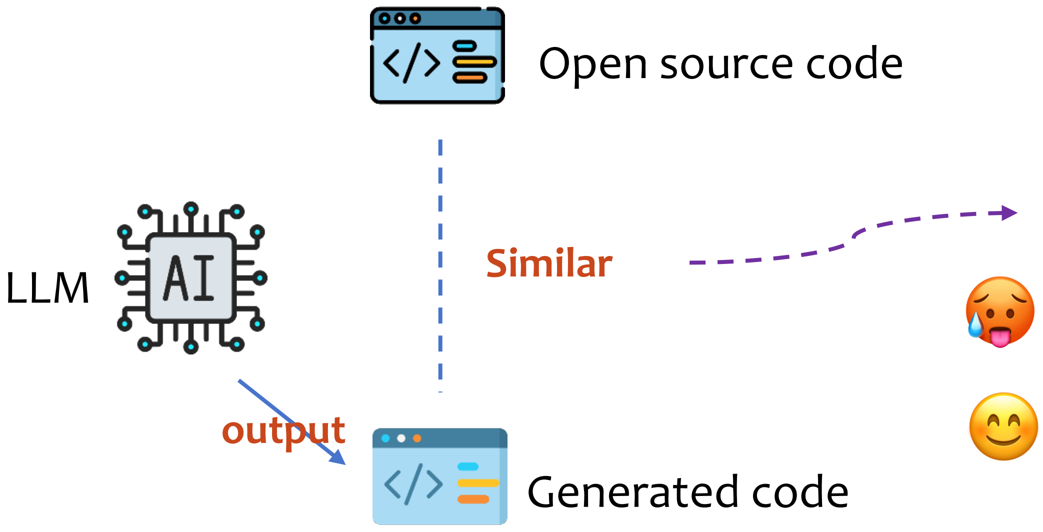
https://githubcopilotlitigation.com/

https://github.blog/news-insights/research/the-state-of-open-source-and-ai/

Evaluating LLM's license compliance in code generation is important but challenging

- Evaluate their ability to provide accurate license and copyright information during code generation
  - protect the IP rights of numerous open-source developers
  - shield users of such LLMs from unforeseen legal risks

- Challenge: Copying or coincidentally similar? 🤔



How to determine Copying or Not?

- 🥵 Copied from OSS ➡ entail issues of license compliance
- 😊 Independently generated and coincidentally similar

The key principle in law determining copyright infringement comes to the rescue

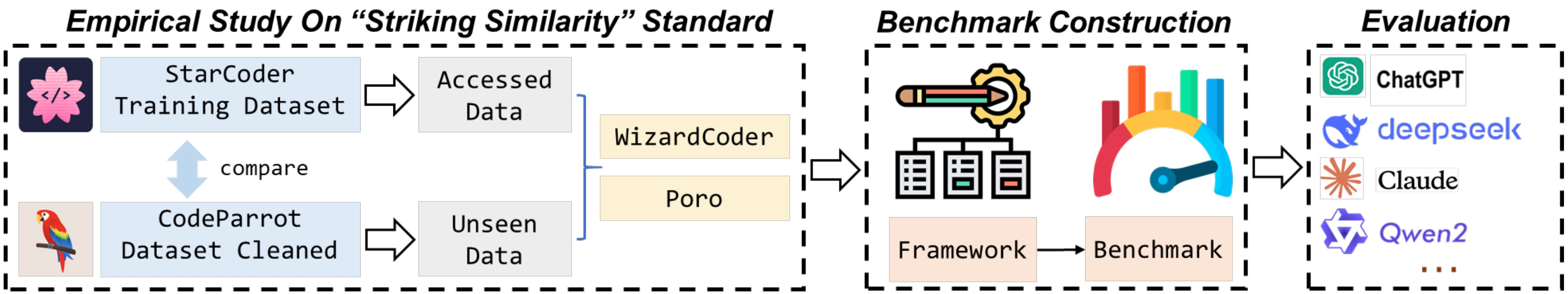
Courts have identified **two methods** to prove substantial copying of an original work[1].

- The plaintiff can choose to provide evidence showing that the defendant had "access" to the copyrighted work and that the two works are "substantially similar." ❌Because LLM's training data is usually undisclosed, "access" can be proven.

- On the other hand, when access cannot be proven, the plaintiff can provide evidence demonstrating that the works in question have **"striking similarity" (sufficient to rule out the possibility of independent creation).** ✅

[1] U. S. C. for the Ninth Circuit. Copying—access and substantial similarity. [Online]. Available: https://www.ce9.uscourts. gov/jury-instructions/node/274

Our Solution: evaluating license compliance of LLMs based on Striking Similarities



Overview of this study

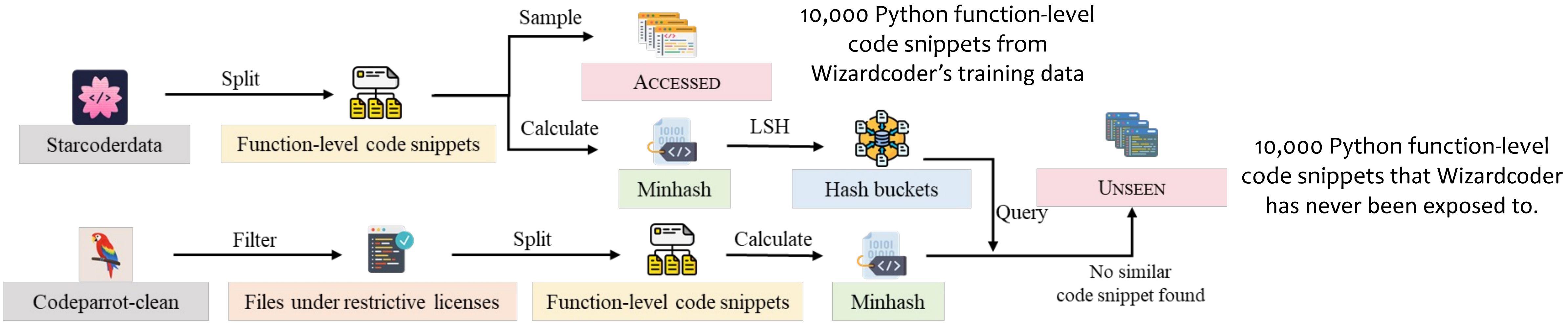
Establishing Striking Similarity Standard by Comparing LLM's Performance on Generating Accessed and Unseen Data

Where might the reasonable standard of striking similarity lie in the context of code generation by LLMs?

Model for analysis: WizardCoder-15B-V1.0 → 😊**All training data is public.**

Experiment setup:

- construct two distinct groups of code samples, UNSEEN and ACCESSED, to simulate two different scenarios, i.e., independent creation and non-independent creation.



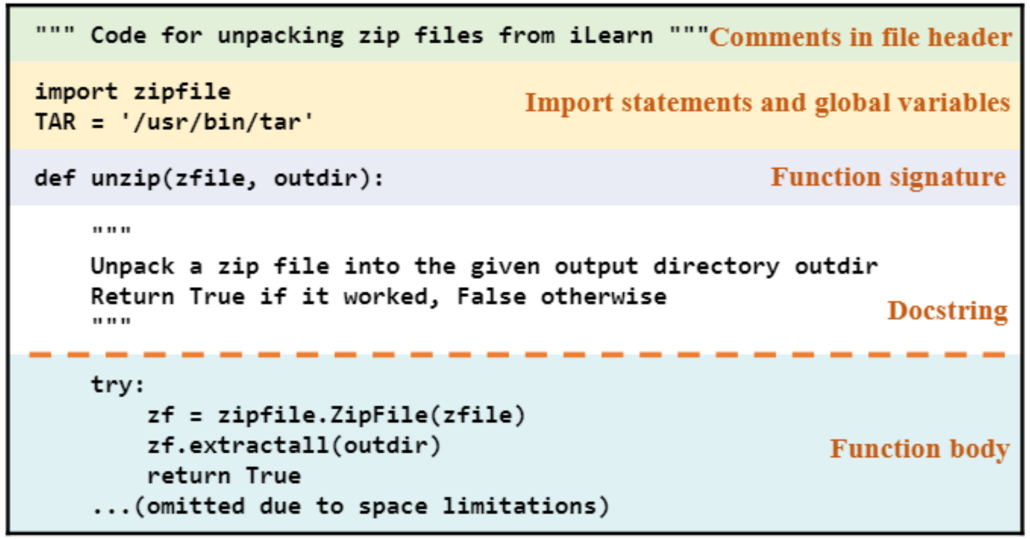
Our goal is to observe potential differences in similarity when the model generates code for these two distinct groups.

Constructing prompts using function headers

Model for analysis: WizardCoder-15B-V1.0

Experiment setup:

- construct prompts using the UNSEEN and ACCESSED groups, then instruct WizardCoder to complete the code snippets

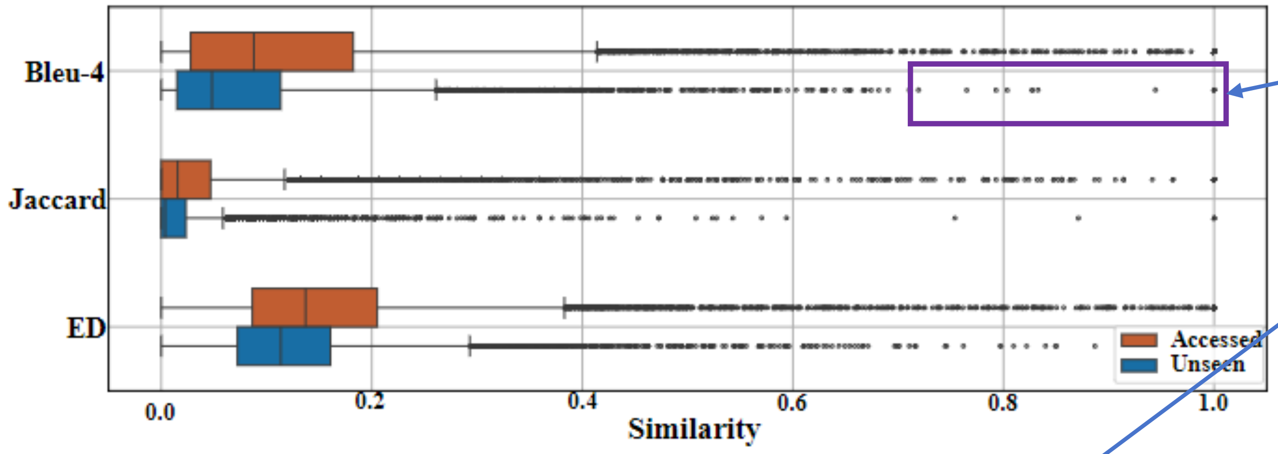


Structure of function-level code snippet

prompt

```
Complete the following Python function:
""" Code for unpacking zip files from iLearn """
import zipfile
TAR = '/usr/bin/tar'
def unzip(zfile, outdir):
    """
    Unpack a zip file into the given output directory outdir
    Return True if it worked, False otherwise
    """
```

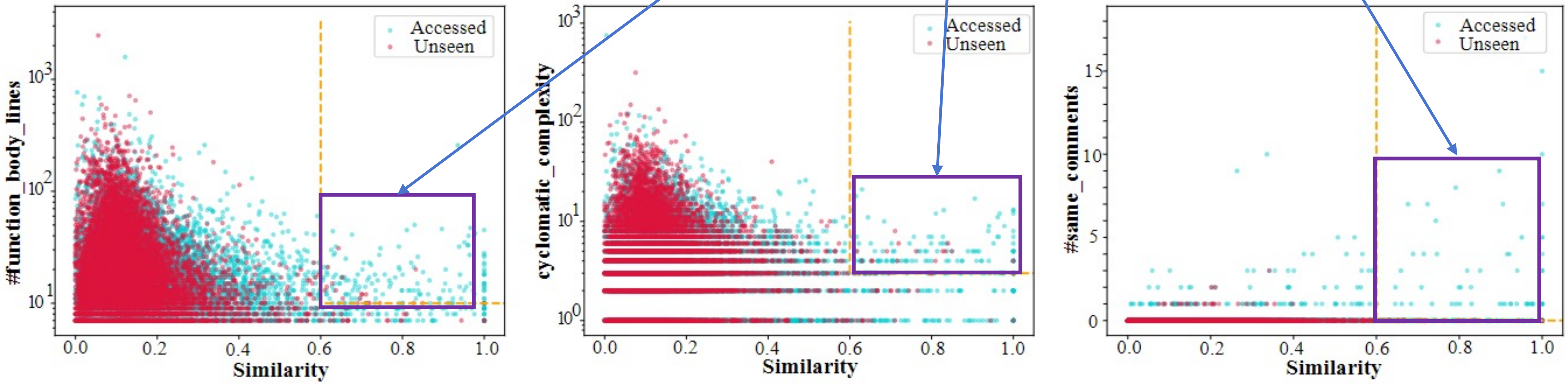
Different performances of WizardCoder for two groups



Text similarity alone cannot determine non-independent creation in LLM-generated code.

For complex functions in the UNSEEN group, the similarity is typically lower.

LLMs can memorize and reproduce comments from training data.



(a) (b) (c)

A simple but effective standard for “Striking Similarity”

Striking Similarity

- number of function body lines > 10
- cyclomatic complexity > 3
- text similarity > 0.6
- number of identical comments > 0

Copyright laws only protect expression (i.e., the specific expression of code), not ideas.

Evaluation

10,000
UNSEEN_Eval

10,000
ACCESSED_Eval

Wizardcoder

Poro-34b-chat

Trained on

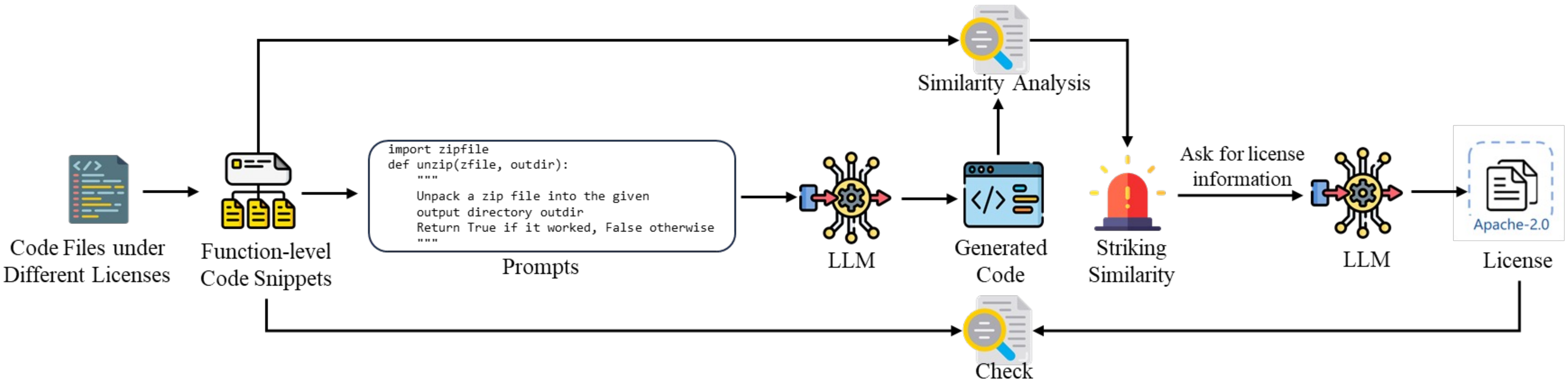
Striking Similarity

33 cases

33/33 from ACCESSED_Eval

32/33 excluded independent creation (expert evaluation)

The evaluation framework inspired by the empirical study



Overview of the evaluation framework

Constructing benchmark based on World of Code

</> WoC

700,867,958 unique Python file blobs

filter

PY PY PY

Blobs containing explicit license information in their file header comment

sort

PY

sorted by the number of projects containing the blob in descending order

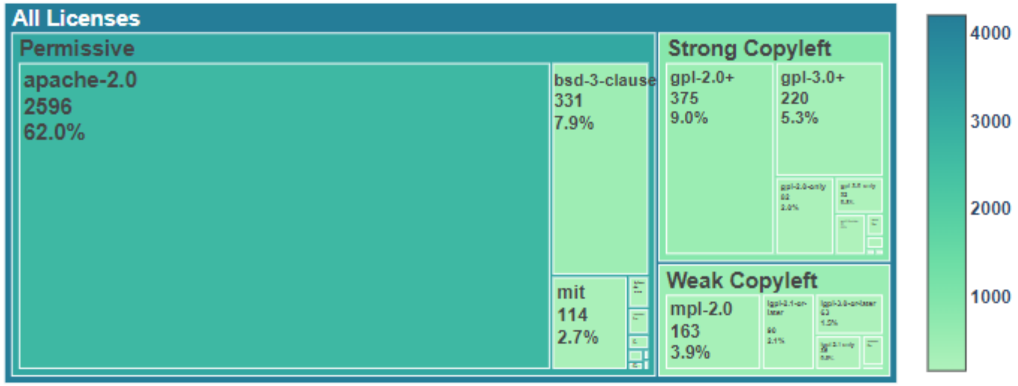
segment

top 10,000 functions meet the preconditions of our striking similarity

blobs widely acknowledged by the opens ource community in terms of their copyright information

Non-trivial

deduplicate



The distribution of license in LiCoEval

Retain a natural distribution of licenses

4,187 samples

Evaluating 14 LLMs on LicoEval

PERFORMANCE OF 14 LLMS ON LICOEVAL. ✓MEANS PUBLICLY AVAILABLE WEIGHTS AND × MEANS UNAVAILABLE WEIGHTS.

	Model	HumanEval	Weights	#striking_sim	Acc	#permissive	Acc_p	#copyleft	Acc_c
General LLM	GPT-3.5-Turbo	72.6	×	29 (0.69%)	0.72	26	0.81	3	0.0
	GPT-4-Turbo	85.4	×	25 (0.60%)	0.72	22	0.82	3	0.0
	GPT-4o	90.2	×	47 (1.12%)	0.74	41	0.85	6	0.0
	Gemini-1.5-Pro	71.9	×	41 (0.98%)	0.59	39	0.62	2	0.0
	Claude-3.5-Sonnet	92.0	×	84 (2.01%)	0.69	79	0.71	5	0.4
	Qwen2-7B-Instruct	79.9	✓	20 (0.48%)	0.95	20	0.95	0	-
	GLM-4-9B-Chat	71.8	✓	0 (0.0%)	-	-	-	-	-
	Llama-3-8B-Instruct	62.2	✓	1 (0.02%)	0.0	1	0.0	0	-
Code LLM	DeepSeek-Coder-V2	90.2	✓	37 (0.88%)	0.0	36	0.0	1	0.0
	CodeQwen1.5-7B-Chat	83.5	✓	17 (0.41%)	0.24	17	0.24	0	-
	StarCoder2-15B-Instruct	72.6	✓	13 (0.31%)	0.23	13	0.23	0	-
	Codestral-22B-v0.1	61.5	✓	91 (2.17%)	0.73	87	0.77	4	0.0
	CodeGemma-7B-IT	56.1	✓	3 (0.07%)	0.33	3	0.33	0	-
	WizardCoder-Python-13B	64.0	✓	27 (0.64%)	0.04	26	0.04	1	0.0

Discussion

Limitation:

- a “minimum” standard that emphasizes precision and interpretability
- may perform poorly on recall

Even with such a minimum standard, we are still able to obtain concerning results from state-of-the-art LLMs.

What’s more:

- Different prompts…
- Different scopes(File? Class?)
- Different languages

Discussion

Implications:

LLM providers:

- Data Cleaning and License Detection
- Enhancing License-Code Association
- Addressing Copyleft Information Suppression

Open-source communities:

- Adopting more explicit license declarations
- Developing guidelines for incorporating and attributing AI-generated code in projects
- Establishing clear policies on how their own code should be used in AI training process

LLM users:

- Be aware of risks
- Verify generated code

Legal professionals:

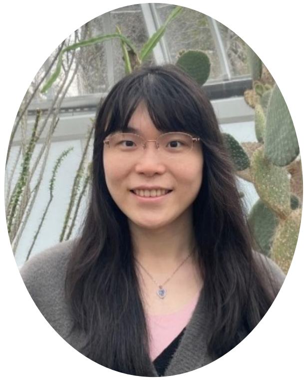
- It is feasible to characterize non-independent creation in LLM outputs using specific feature.

The IEEE/ACM International Conference on Software Engineering (ICSE 2025)

Thank you!

Paper

Code



Weiwei Xu
xuww@stu.pku.edu.cn
Peking University

Kai Gao
kai.gao@ustb.edu.cn
University of Science and Technology Beijing

Hao He
haohe@andrew.cmu.edu
Carnegie Mellon University

Minghui Zhou
zhmh@pku.edu.cn
Peking University

Acknowledgement: National Natural Science Foundation of China 62332001.