

LLMs effectively spot misleading identifiers in student code but can be overly pedantic.



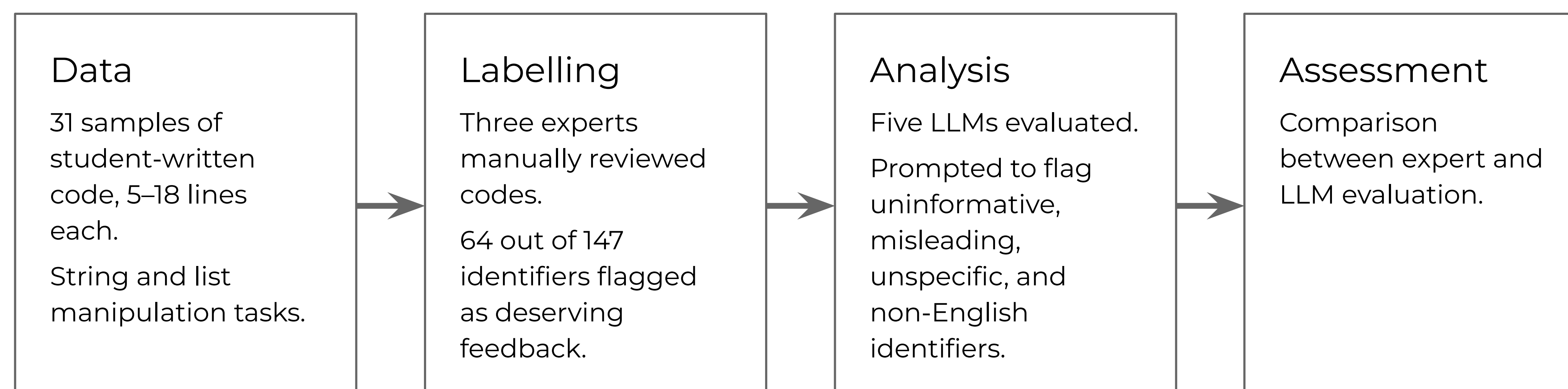
Scan for:
- the full extended abstract
- contact info
- more cat drawings

Motivation

Appropriate identifiers are critical for readable code, so programming students need feedback on their usage, ideally in a scalable way.

Manual code review is time-intensive and automated tools often miss context-dependent naming issues – creating a need for more sophisticated solutions.

Approach



```
def five_multiples(seznam): # non-English
    x = []                 # uninformative
    for i in seznam:      # misleading
        if i % 5 == 0:
            x.append(i)
    return x
```

	accuracy	precision	recall	f-score
ChatGPT-4o	72%	76%	94%	0.84
Claude	59%	61%	96%	0.74
Gemini	40%	52%	64%	0.57
ChatGPT-3.5	33%	47%	53%	0.50
ChatGPT-4	32%	38%	62%	0.47

Insights

- Best performing LLMs were too strict, flagging identifiers as misleading even when experts found them acceptable (e.g., suggesting `list_of_numbers` over `numbers`).
- Attempts to reduce the pedantry failed:
 - ◆ Tweaking the prompt didn't help.
 - ◆ Asking for severity score improved precision to 90% but reduced recall to 22%.
- ChatGPT-4o performed consistently across 27 tests, with less than 1% variation.

Future Work

Study how to make LLM feedback align with CS1 pedagogical practices, such as adapting feedback strictness based on student learning stage and course context.

Systematically explore robustness of poor names detection (more naming issues, more diverse codes).