Evaluating the Environmental Impact of Using SLMs and Prompt Engineering for Code Generation

Md Afif Al Mamun, Sayan Nath, Novarun Deb, Gias Uddin

Presented By: Pragati Kumari

The International Conference on Evaluation and Assessment in Software Engineering (EASE)
Glasgow, United Kingdom · June 9–12, 2026

Background

- **Small Language Models (SLM)** are open-source language models, tens of billions parameters, typically small enough to run locally on a developer's laptop or workstations.
- **Usage on the rise:** local inference offers data privacy, no vendor lock-in, no API cost, offline use.
- Code generation is moving from cloud data centers to developer machines.

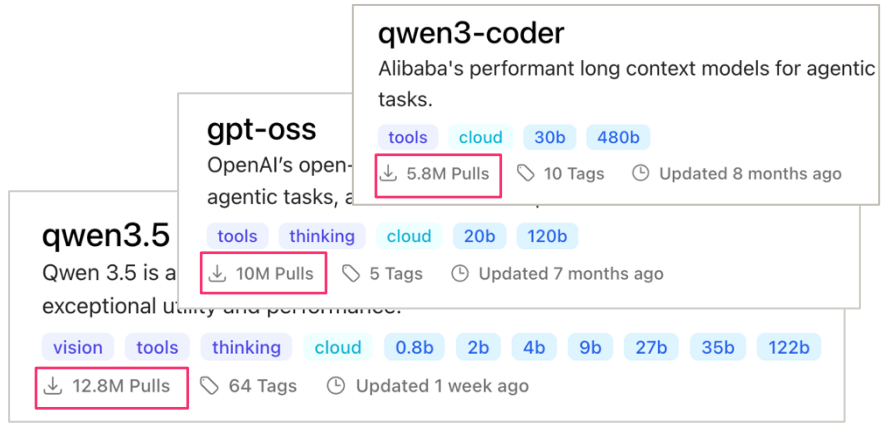


Figure: Open source SLMs from Ollama repository with millions of downloads.

Motivation

- **Inference dominates emissions:** 60% of a model's lifetime CO_2 comes from inference, not training. Code generation is one of the longest-output, most-repeated inference workloads[1].
- **Local SLMs decentralise the carbon bill:** trivial per query, but at millions of developers × thousands of calls/week, the cumulative footprint is substantial and invisible.
- **The blind spot:** many prompting strategies, yet without evidence, *"best prompt"* = highest Pass@1, silently burning carbon for marginal accuracy. *So, what's the cost?*

Research gap

No prior work systematically quantifies the sustainability trade-offs of prompting strategies for SLM-based code generation across heterogeneous hardware and electricity grids.

1. https://research.google/blog/good-news-about-the-carbon-footprint-of-machine-learning-training/

Contribution & Findings

UNIVERSITY OF CALGARY

01

First systematic study

Accuracy ↔ sustainability of 6 prompting strategies across 11 open-source SLMs (1B–34B) on HumanEval+ and MBPP+.

02

Sustainability decouples from accuracy

CoT achieves near-best Pass@1 while cutting CO_2 by up to **~80%** versus complex reasoning frameworks.

03

Grid > hardware

Regional grid carbon intensity is the dominant emissions driver; often outweighing hardware differences by an order of magnitude.

04

SCI Specification

CpCA metric

Carbon per Correct Answer (CpCA): an SCI-inspired metric that normalizes emissions against functional correctness.

Research Questions

UNIVERSITY OF CALGARY

RQ1

How do code generation accuracy and CO_2 emissions scale across SLM parameter sizes?

Do bigger models necessarily emit more? Are smaller models a viable greener choice?

RQ2

How do prompting strategies trade off between performance and sustainability metrics?

Is there any optimal prompting strategy?

RQ3

How do hardware architecture and Grid Carbon Intensity (GCI) shape inference emissions?

Same model, same prompt — different region: how much does that change the carbon bill?

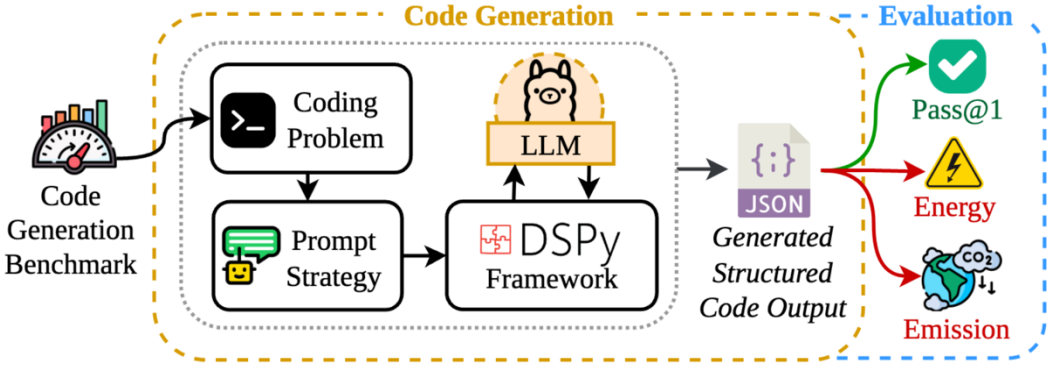
RQ4

What correlations exist between token usage, inference time and computational cost?

Are tokens a good proxy for energy? Or is wall-clock time the real predictor?

Experimental Setup

- Coding Benchmarks:
  - HumanEval+ (164 problems)
  - MBPP+ (150 problems)
- Hardware / regions:
  - Alberta, Canada (Xeon Silver + A100)
  - Ontario, Canada (Xeon Gold + L40S)
- Carbon Emission Estimation:
  CO_2eq = Energy/problem × Regional Grid Carbon Intensity



Model Categories

UNIVERSITY OF CALGARY

Tiny *1B – 3B*	Small *7B – 16B*	Medium *20B – 34B*
LLaMa3.2:1B general purpose	**CodeGemma:7B** coder model	**GPT-OSS:20B** general purpose
Qwen3:1.7B general purpose	**Qwen2.5-Coder:7B** coder model	**Gemma3:27B** general purpose
StarCoder2:3B coder model	**Phi-4:14B** general purpose	**Qwen3-Coder:30B** coder model
	DeepSeek-Coder-V2:16B coder model	**CodeLlama:34B** coder model

Prompting Strategies

UNIVERSITY OF CALGARY

1. **Direct**
   Problem in, code out. Single pass, no intermediate reasoning.
   ↓ low reasoning overhead

2. **Chain-of-Thought**
   Step-by-step reasoning before final code (CoT).
   ↓ low reasoning overhead

3. **Program-of-Thought**
   Decompose + syntactically verify generated code (PoT).
   → mid reasoning overhead

4. **Self-Consistency**
   Sample multiple CoT solutions, pick the consensus.
   → mid reasoning overhead

5. **Least-to-Most**
   Decompose into sub-problems, solve sequentially with state.
   ↑ high reasoning overhead

6. **ReAct**
   Iterative reason → act → observe loop, syntax-validation tool.
   ↑ high reasoning overhead

Evaluation Metrics

Standard metrics

Pass@1
Fraction of problems solved correctly on first attempt.

Avg. CO_2eq (kg)
Energy × regional grid intensity.

Avg. Energy (kWh)
Aggregated CPU + GPU + RAM power, integrated over inference.

Avg. Inference Time (s)
Wall-clock per problem.

Avg. Total Tokens
Prompt + completion tokens per problem.

NEW METRIC

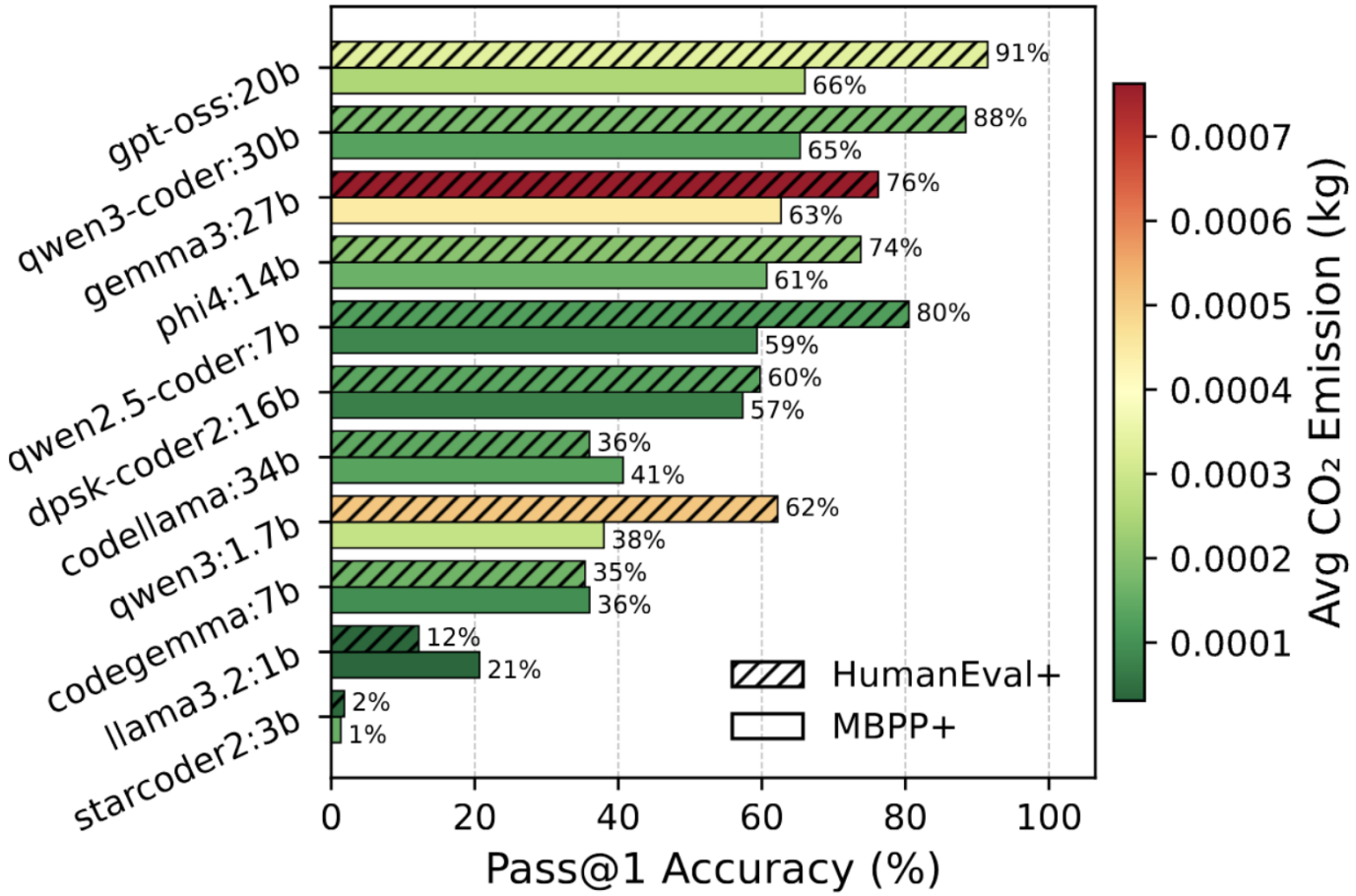
Carbon per Correct Answer (CpCA)

$$\text{CpCA} = \text{Avg. } CO_2 \;/\; \text{Pass@1}$$

↓ *lower is better*

Inspired by the Software Carbon Intensity (SCI) specification, CpCA normalises emissions against the functional unit that developers actually care about — a correct solution.

RQ1 – Accuracy-Emissions Relationship



Key findings

Bigger ≠ better.

GPT-OSS:20B beats Qwen3-Coder:30B on both datasets.

Reasoning wins over size

Qwen3:1.7B beats CodeLlama:34B (~20× larger) on HumanEval+.

CO_2 weakly tied to size.

Per-query emissions span 3e-5 – 7e-4 kg, driven by efficiency, not parameters.

Specialization ≠ winning.

Code-specialised models **do not** consistently beat general-purpose ones.

RQ2 – Impact of Prompting Strategies

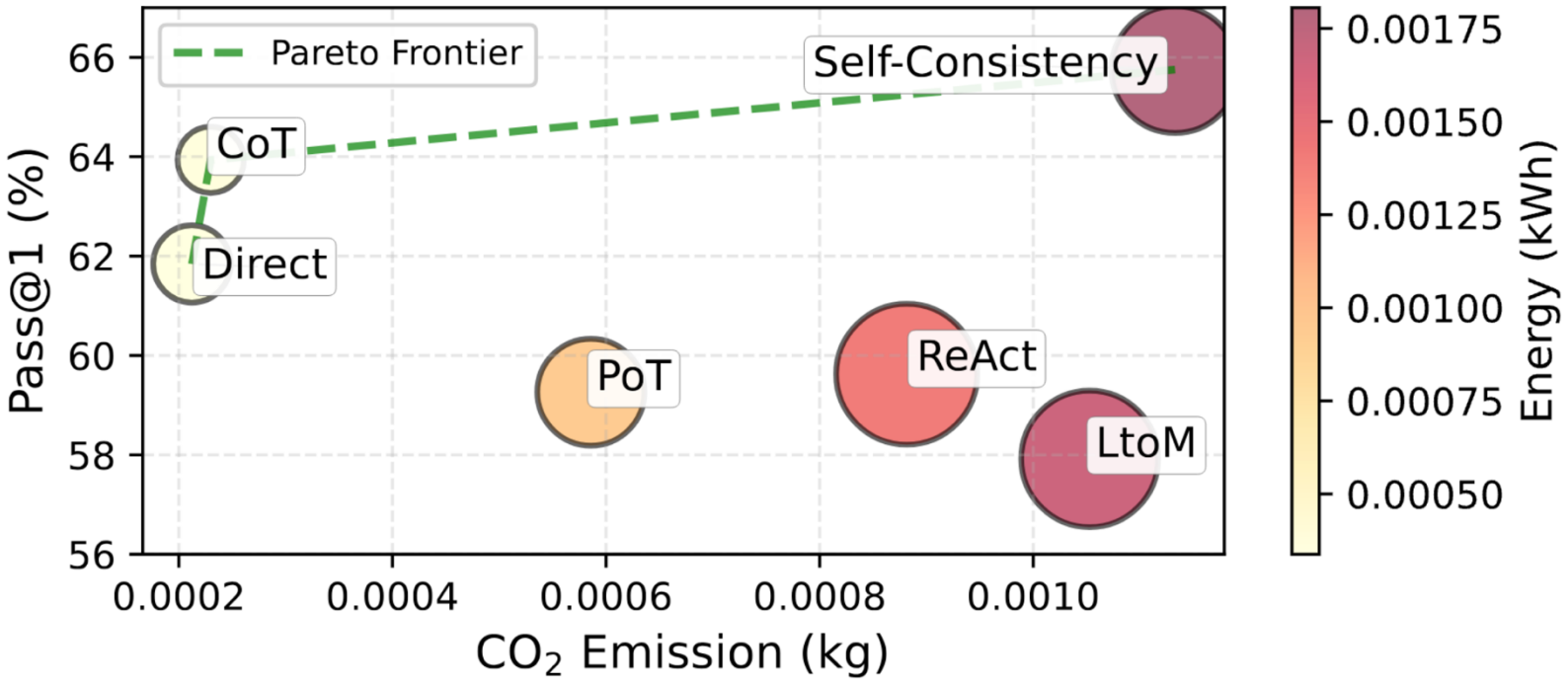


Figure: Bubble Chart for Pass@1 vs Sustainability Metrics. Bubble size represents token count. Results are reported as the average of both HumanEval+ and MBPP+ datasets

Key findings

Chain-of-Thought is the sweet spot.

–2.7% Pass@1 vs. Self-Consistency, **–80%** energy and CO_2.

Multi-sample strategies (Self-Consistency, ReAct, LtoM) burn 4–6× the carbon for marginal accuracy with SLMs.

RQ3 – Influence of Hardware and Region

Machine 1 — Alberta, Canada

Xeon Silver 4316 + NVIDIA A100 40GB · 64 GB RAM

Avg. Energy (kWh)	0.000297
Estimated GCI	≈ 0.627 $kgCO_2/kWh$ (coal + gas)
Avg. CO_2 (kg)	0.000186
Pass@1 (MBPP+)	58.66 %

Machine 2 — Ontario, Canada

Xeon Gold 6442Y + NVIDIA L40S 48GB · 250 GB RAM

Avg. Energy (kWh)	0.000634 ▲ +113.5%
Estimated GCI	≈ 0.065 $kgCO_2/kWh$ (nuclear + hydro)
Avg. CO_2 (kg)	0.000041 ▼ −78.0%
Pass@1 (MBPP+)	58.25 % (−0.71 %)

Key findings

GCI controls the CO_2 emission profile

Grid carbon intensity is a first-order determinant of inference-time emissions — often an order of magnitude beyond hardware-level differences.

Observation

If Machine 2 (Ontario) had run under Alberta's electricity mix, its CO_2 would jump more than **2× Machine 1**

RQ4 – Token Usage and Sustainability Factors

	Pass@1	CO_2	Tokens	Time	Energy
Pass@1	—	+0.04	-0.18	-0.02	-0.11
CO_2	+0.04	—	+0.46	+0.72	+0.33
Tokens	-0.18	+0.46	—	+0.68	+0.63
Time	-0.02	+0.72	+0.68	—	+0.87
Energy	-0.11	+0.33	+0.63	+0.87	—

Pearson correlation matrix (MBPP+, merged hardware).

Key findings

Inference time is highly correlated with energy consumption and CO_2 emission.

#Tokens only moderately relates to emission.

Pass@1 has no meaningful correlation with any sustainability metric.

Discussion – Carbon per Correct Answer (C_PC_A)

Strategy	Machine 1 (AB)	Machine 2 (ON)
Direct Prompting	0.000341	0.000092
CoT	**0.000370**	**0.000084**
PoT	0.001079	0.000240
LtoM	0.001773	0.000383
Self-Consistency	0.001752	0.000444
ReAct	0.001794	0.000679

Lower CpCA = greener per correct answer.

Stable winner

CoT and Direct dominate CpCA on both machines. Strategy ranking is stable across regions — even as absolute values shift with the grid.

5× penalty

Self-Consistency improves Pass@1 by only a few points but multiplies CpCA by **>5×** — a poor sustainability trade for SLMs on simple problems.

Software vs. infrastructure

Greener grids lower the absolute cost; smarter prompting lowers it everywhere.

Limitations – Threats to Validity

- System background noise can perturb fine-grained energy and time measurements.
- The results are based on only two Canadian regions; considering global regions may impact the order of the prompting strategies.
- Static GCI via CodeCarbon may overestimate emissions in rapidly decarbonizing grids.

Takeaway

UNIVERSITY OF CALGARY

Accuracy ⊥ sustainability

Pass@1 shows **no meaningful** correlation with energy or CO_2. Practitioners should optimize for accuracy-per-watt, not accuracy alone.

CoT > heavyweight reasoning

Complicated reasoning steps may not improve performance in SLMs.

Grid > hardware

Regional GCI drives an order-of-magnitude swing in emissions — **model & prompt choices are nonetheless the most actionable lever.**

Greener code generation is a prompting decision before it is a hardware or regional decision.

Conclusion

Code generation can be greener AND accurate, if we treat prompting as a sustainability decision.

For Developers

Treat prompting as a sustainability choice; default to Chain-of-Thought unless multi-sample reasoning is needed

For Tool Builders

Surface per-prompt energy & CO_2 in IDEs and coding assistants.

Researchers

Pair code-gen benchmarks with energy and carbon reporting when possible, not just Pass@1.

Replication Package

github.com/afifaniks/slm-codegen-sustainability

Contact

afif.mamun@ucalgary.ca

Thank for listening, questions?