scientific reports

Check for updates

OPEN

A comparative study of AI and human programming on environmental sustainability

Nolan H. Woo

Despite rising concerns over AI's environmental impact, a recent article claimed human writers emit 130 to 1,500 times more greenhouse gases than AI. While the study utilized life cycle assessment methodology, it overlooked a critical factor: output quality. Unlike AI, which instantly generates content, human writers work for months, producing superior results. To provide a more objective comparison, we analyzed the environmental impacts of human and AI programmers generating functionally equivalent code. Using the USA Computing Olympiad database, we developed infrastructure to evaluate multiple GPT-based models. To our knowledge, this is the first study to correctness-control and quantitatively assess AI's environmental impact in code generation. To address AI's inaccuracies, we built a multi-round correction process to iteratively fix responses. We calculated AI emissions from usage and embodied impacts, while human emissions were estimated using average computing power consumption. Our case study results show that smaller models can match the environmental impact of human programmers when they succeed, though they often fail. However, standard, widely-used models are far more environmentally strenuous. For example, GPT-4 emitted between 5 and 19 times more CO_2eq than humans, underscoring a much greater trade-off between efficiency and environmental cost than previously understood.

Keywords Environmental Impact, Large Language Model, Computational Sustainability, AI-Assisted Programming

As AI models continue to advance, there is a growing belief that increasing a model's scale leads to higher performance[1]. However, this improvement comes at a cost: larger models require significantly more computing resources. As a result, the community has begun raising concerns about the environmental impact of both training and serving these large models[2,3].

In contrast, a recent article reported that AI systems produce less carbon dioxide emissions than humans when performing tasks like writing and illustrating[4]. Although AI services consume significantly more power per user request, fulfilling these requests only requires a short duration of time. On the other hand, humans take longer to finish the same tasks, which leads to increased energy consumption, such as through prolonged use of a laptop. As a result, the article claims that an AI writer would emit 130 to 1,500 times less carbon dioxide than a human writer.

Unfortunately, the article did not compare the quality of the output delivered by AI and human creators, making a fair, apples-to-apples comparison impossible. To avoid subjectivity in evaluating writing or illustration quality, this study objectively quantifies and compares the environmental impacts of AI and human work by measuring their performance on programming tasks, where quality is precisely quantifiable.

Large language model (LLM)-based programming, in particular, is a rapidly evolving field, and its environmental impact is likely to change just as quickly. In addition to recent coding assistant tools such as GitHub Copilot[5] and Cursor[6], new concepts such as vibe coding[7] or self-correcting multi-step agent[8] have emerged. In this development practice, individuals without technical backgrounds can write code using natural language, often just their voice, and interact with LLM tools repeatedly until functionally satisfactory code is produced. This approach relies on LLMs even more heavily than traditional AI-assisted coding, where a human programmer reviews, understands, and debugs the generated code when it produces incorrect outputs. If this practice becomes mainstream, people will likely rely on LLMs even more aggressively for programming tasks. More importantly, the productivity gains could significantly increase the total amount of software being produced, making the environmental impact of LLMs an even more critical concern. This growing usage, combined with the lack of objective comparisons between humans and AI, highlights the need for research like ours to better understand the true environmental cost of LLM-based programming.

Saratoga High School, Saratoga, USA. email: nolan.h.woo@gmail.com

Research gap and contribution While recent studies have estimated the environmental impact of LLMs in general-purpose or other tasks[2–4,9–11], none have quantitatively compared AI and human programmers on the same functionally equivalent coding problems with correctness explicitly measured. Prior work has also not examined how iterative error correction influences emissions. Our study addresses these gaps by (i) selecting programming problems with clear correctness criteria, (ii) using a multi-round correction process to simulate AI debugging behavior, and (iii) applying life cycle assessment methods to both AI and human programming scenarios. To our knowledge, this is the first ever correctness-controlled study to conduct an environmental comparison between AI and human programmers.

Design and methods

USACO problems

To objectively compare the quality of programming tasks done by AI and humans, we first needed to find a suitable set of programming tasks that both could tackle under comparable conditions. One approach would have been to gather a group of human programmers, assigning them a set of problems within a fixed time frame and measuring their energy consumption and environmental impact. However, conducting such a large-scale experiment would be costly and prone to sampling bias. Instead, we decided to leverage the historic database of the USA Computing Olympiad (USACO)[12]. USACO is a timed algorithmic programming contest held four times annually in the December, January, February, and US Open competitions. The USACO contest served as a strong evaluation benchmark because it maintains a historical database of problems and test suites, human participants are primarily focused on solving the problems without distractions, and the fixed time limits allow us to estimate energy consumption and its corresponding environmental impact. The competition is divided into four levels: Bronze, which assesses basic concepts like sorting and binary search; Silver, which tests foundational problem-solving strategies and data structures; Gold, which introduces more advanced algorithms such as shortest-path and dynamic programming; and Platinum, designed for top participants tackling sophisticated and open-ended challenges. Although USACO problems are competitive rather than real-world software tasks, they offer objective scoring that allows for reproducible comparisons.

Multi-round correction process

To measure the environmental impact of AI systems solving USACO problems, we built an infrastructure to ask the OpenAI API service[13] to solve a given problem with the Python API. Out of the six different language choices that USACO allows, we selected Python because generative pre-trained transformer (GPT) models are known to perform better for high-level dynamically typed languages[14]. In fact, we initially conducted our experiments using C++, but the quality of LLM-generated code was insufficient to support meaningful comparisons with human programmers. By switching to Python, we were able to identify at least one problem set suitable for reliable evaluation, as detailed in a later section.

Figure 1 shows the infrastructure used to evaluate the environmental impact of AI-generated code. USACO problems and corresponding test cases are first selected and fed into a pre-processing stage that formats the prompts for GPT-based models. In this study, we used four different models available at the time of evaluation, GPT-4o-mini, GPT-4o, GPT-4-turbo, and GPT-4. The GPT service is then called through the OpenAI API. The returned code is executed and validated against USACO test cases.

Please note that an open-source model would have been more transparent from the perspective of environmental impact modeling, but, as this study found, the accuracy of functional output turns out to be extremely important to reduce the environmental impact, and, as of today, commercial models significantly outperform open source models in terms of functional accuracy for coding tasks. Thus, we decided to use OpenAI models as OpenAI offers the wide spectrum of models according to the SWE-bench leaderboard[15], allowing us to perform trade-off studies detailed later.

Unfortunately, because the outputted Python code is often incorrect, we built a multi-round correction process to iteratively correct erroneous responses[16]. Specifically, we categorized incorrect behaviors into three different issues and provided issue-specific feedback to the AI as shown in Table 1, followed by the prompt, "Can you review your code thoroughly and fix the code?" While USACO typically requires a Python program to execute within four seconds for each test case, achieving functional output from AI-generated code often proved difficult, as will be discussed later. Therefore, we extended the execution time for AI-generated problems to 100 seconds. Additionally, each USACO problem includes multiple test cases (usually between 10 and 20), and we allowed up to 100 rounds of iteration for a GPT model to produce correct outputs across all test cases. The number of LLM calls per task significantly depends on a given task and consequently affects energy consumption. For example, a recent article suggests a sophisticated decision tree to solve software engineering tasks, often calling LLMs 300 to 2,000 times[17]. Given that the number of calls needed will heavily depend on a target task, in this study, we decided to use 100 rounds following the observation made by Chen et al. (e.g., pass@100 metric)[18]. While the original proposal was more about statistical observations for the same prompt, we decided to follow their approach—which is heavily referenced by our research community—because a multi-step agentic execution flow is still an actively researched area and there is no single, well-established answer for the optimal number of LLM calls needed, as shown later from our experiments. Lastly, to account for the context length limitation of the models, we retained only the last ten rounds of conversations for each prompt. This threshold was sufficient to preserve relevant feedback while avoiding context overflow.

Environmental impact modeling

Once the program passes all test cases or reaches the iteration limit, the run is recorded. To estimate the energy consumption and environmental impact, we leveraged an open-source project, Ecologits 0.8.1, that employs life cycle assessment (LCA) methodology as defined by ISO 14044[19]. Ecologits accounts for the embodied

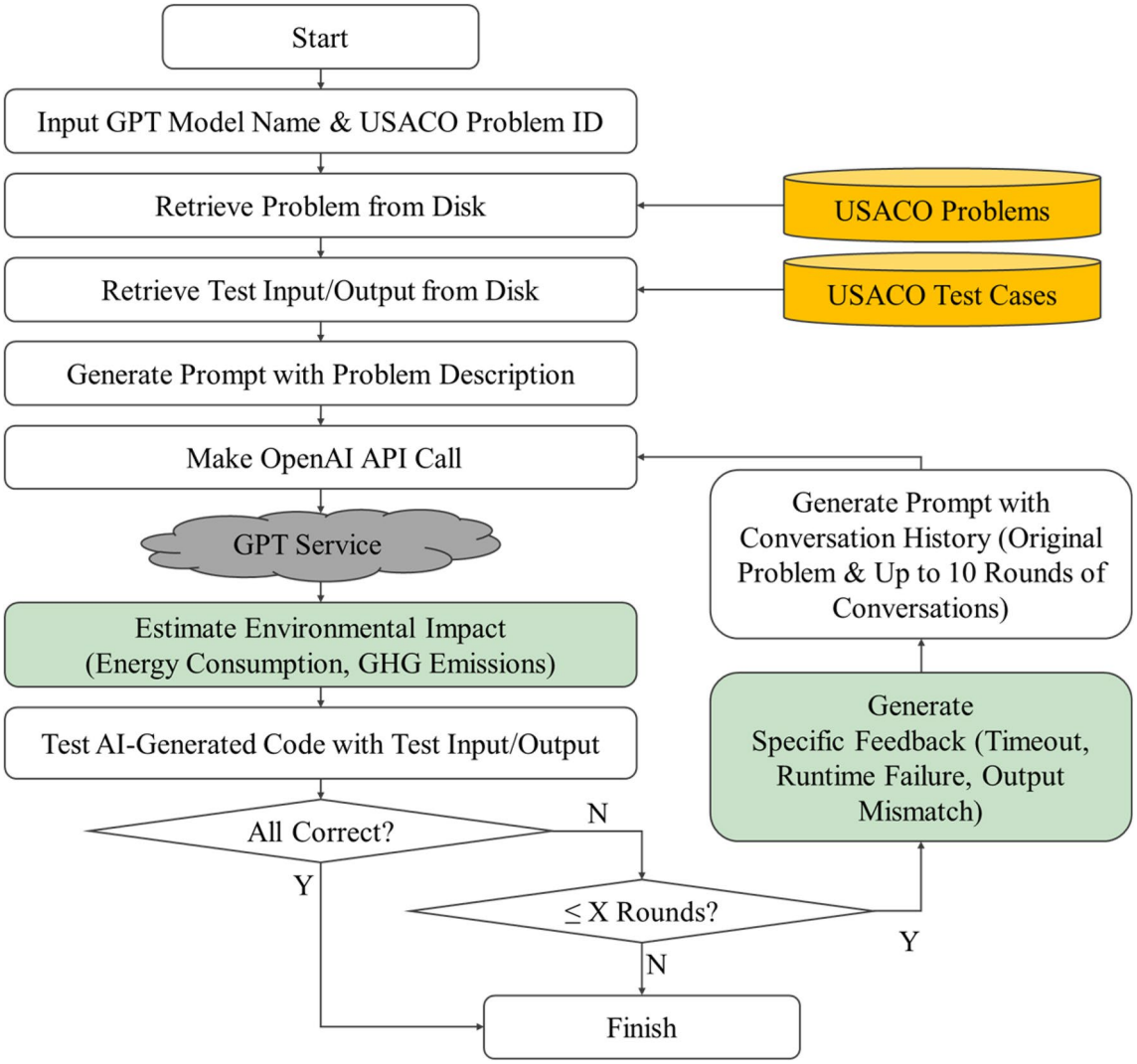


Fig. 1. Overall Workflows.

Types	Feedback
Timeout	"Your code took long and failed to finish within 100 seconds."
Runtime failure	"Your code failed at runtime. The first 3000 characters of the log are as follows:" + [The first 3000 characters of runtime log]
Output mismatch	"Result mismatched. The input was as follows:" + [input data] + "The expected output was as follows:" + [expected output] + "However, the actual output was as follows:" + [actual output]

Table 1. Incorrect behaviors and feedback.

ecological impacts and the usage impact of an AI inference service, but it excludes the impact of training: a limitation discussed later in this article. Specifically, the functional unit is one LLM inference request, while the system boundary adheres to the 'Cradle-to-Gate' scope[20] from the perspective of an LLM inference request. This includes both usage impacts and embodied impacts of the computing and cooling facilities of a datacenter. End-of-life impacts are excluded, and the allocation method is based on the time-weighted resource consumption of each request.

To estimate the environmental footprint of AI inference, Ecologits follows a two-part framework as shown in Table 2: usage impacts from energy consumption and embodied impacts from hardware production. This approach aligns with LCA standards and provides a holistic view of per-query emissions.

Usage impacts account for the operational energy required to perform a single inference. At the top level, this is scaled by the power usage effectiveness (PUE), a metric that reflects overall data center efficiency by incorporating both computing and overhead energy use (e.g., cooling and power delivery). The total energy

AI inference impact = Usage Impacts (energy consumption) + Embodied Impacts (hardware carbon footprint)	
Usage Impacts	$E_{\text{request}} = \text{PUE} \times (E_{\text{server}\setminus\text{GPU}} + \#\text{GPU} \times E_{\text{GPU}})$
	$E_{\text{GPU}} = T_{\text{out}} \times (\alpha \times P_{\text{active}} + \beta) \quad \rightarrow \quad$ from linear regression
	$E_{\text{server}\setminus\text{GPU}} = \Delta T \times W_{\text{server}} \times \frac{\text{GPU}}{\#\text{GPU}_{\text{installed}}}$
Embodied Impacts	$I^e_{\text{request}} = \frac{\Delta T}{\Delta L} \times I^e_{\text{server}}$
Total Impact	$I_{\text{request}} = E_{\text{request}} \times F_{\text{em}} + I^e_{\text{request}}$

Table 2. AI environmental impact calculation.

consumption includes both GPU, denoted as $\#\text{GPU} \times E_{\text{GPU}}$, and non-GPU server components, denoted as $E_{\text{server}\setminus\text{GPU}}$.

GPU energy is modeled based on the number of output tokens produced, denoted as T_{out}, and the size of the model, represented by P_{active}, the number of active parameters typically measured in billions. These variables are used in a linear regression model trained on an open-source dataset of *openly reported GPU energy measurements from several LLMs*[19]. The coefficients α and β are empirically fitted against these measurements, which serve as the "ground truth labels" for the regression. Because the proprietary details of OpenAI's models like exact GPU counts, parallelism strategy, or active parameter sharing are undisclosed, the framework does not use internal OpenAI information. Instead, Ecologits relies on published benchmarks of similar-sized open models and extrapolates energy trends, based on externally observable properties of OpenAI's models like output token rate. This introduces uncertainty, which is reflected in the reported ranges. Further details on the derivation of α and β are provided in the Ecologits documentation[19].

Server-side energy consumption is based on inference latency, ΔT, which reflects the processing time required to generate a response, and the average server power draw during inference, denoted as W_{server}. This energy is scaled by the proportion of GPU usage relative to the total number of GPUs installed.

Embodied impacts represent the carbon emissions associated with manufacturing the server and GPU hardware. Since these emissions are incurred once at the time of production, they are amortized over the lifespan of the hardware. For inference-level accounting, this embodied footprint is allocated proportionally to each request based on ΔT, relative to the hardware's expected lifetime ΔL, using the total embodied impact I^e_{server} as a baseline.

Total impact is calculated by summing the usage emissions and the amortized embodied emissions. To translate energy use into CO_2-equivalent emissions, we apply an emission factor F_{em}, which by default reflects the global electricity mix. While many modern data centers use renewable energy or low-carbon grid mixes, we use the global average to maintain comparability with prior work[21]; future work could improve the analysis with carbon intensities specific to a center.

Ecologits reports a range of environmental impact estimates—minimum and maximum values—instead of a single point value to reflect the uncertainty and variability in key input factors. These include differences in data center efficiency (PUE), hardware configurations, model sparsity, inference optimizations, and the carbon intensity of electricity used during inference. By capturing this spread, the framework allows for a more transparent comparison across systems and deployment contexts, since real-world conditions often deviate from standardized benchmarks.

To better understand the environmental cost of machine inference, we compared it to the energy used by a human solving the same task. Here, we model the environmental impact of human programmers following the functional unit, the system boundary, and the allocation methods of Ecologits. Specifically, we model embodied impacts and usage impacts of a 13-inch laptop connected to a 27-inch monitor, excluding only their end-of-life impact. If a programmer spends four hours working through the problem set, they would consume roughly 0.24 kWh on the laptop (60 W)[22] and 0.06 kWh on the monitor (15.8 W during its 'on' mode)[23]. To use the same method as Ecologits for embodied impacts, we account for the reported manufacturing and transport impact of these devices, 88.1% of 265 $kgCO_2eq$ for the laptop[24] and 55.8% of 195 $kgCO_2eq$ for the monitor[25]. We then scale these numbers to the four-hour USACO time limit, assuming 3-year lifecycle for the devices, 200 working days per year, and an 8-hour working day, resulting in 0.19 $kgCO_2eq$ and 0.09 $kgCO_2eq$, respectively. Finally, we added the usage impact of 0.12 $kgCO_2eq$ derived from 0.30 kWh[26].

Previous work[4] suggests including average per capita CO_2eq emissions for human programmers, with individuals emitting about 0.53 kg per hour[28], totaling 2.12 kg over a four-hour period. However, baseline human emissions are unrelated to programming tasks and should not be considered when comparing AI-based and human-based programming. Humans will continue to emit CO_2 even while an AI programs, given that they are still alive. To ensure a fair comparison, only emissions directly tied to the programming process were included.

Evaluation results

With the infrastructure in place, we evaluated GPT models using four years' worth of data from the USACO database. Since human programmers tackle three problems over a span of four to five hours in each USACO contest (four hours for December, January, and February tests and five hours for US Open tests), our first step was to identify a contest where a GPT model could successfully solve all three problems. This allowed us to estimate the energy consumption of human programmers based on the contest's time constraints. We acknowledge that

USACO problems represent competitive programming challenges rather than the full spectrum of software engineering tasks; their primary advantage here is the objective scoring and time-bounded nature, which allows controlled comparisons.

Figure 2 summarizes the experiments that we performed using the GPT-4o model on USACO Bronze problems. Each year has four different contests (December, January, February, and US Open), each with three problems. In the figure, "O" denotes the successful execution of all test cases. On the other hand, "F," "T," and "X" represent experiments that failed on the last test case due to output mismatch, timeout, or runtime failure, respectively. The color of each round represents the number of test cases that succeeded before a test case failed to execute properly. Here, we performed only up to 25 rounds to find a problem that can be solved by all the LLMs for a large scale experimentation to be explained later.

As shown in the figure, in many cases, GPT-4o failed to generate the correct code, making it difficult to find a competition where it successfully solved all three problems—a task human programmers are given to complete within four hours. The sole exception was the February 2021 competition, where the model successfully solved all three problems after five, one, and one round of trials, respectively. One caveat to note is that the behavior of the model is not deterministic.

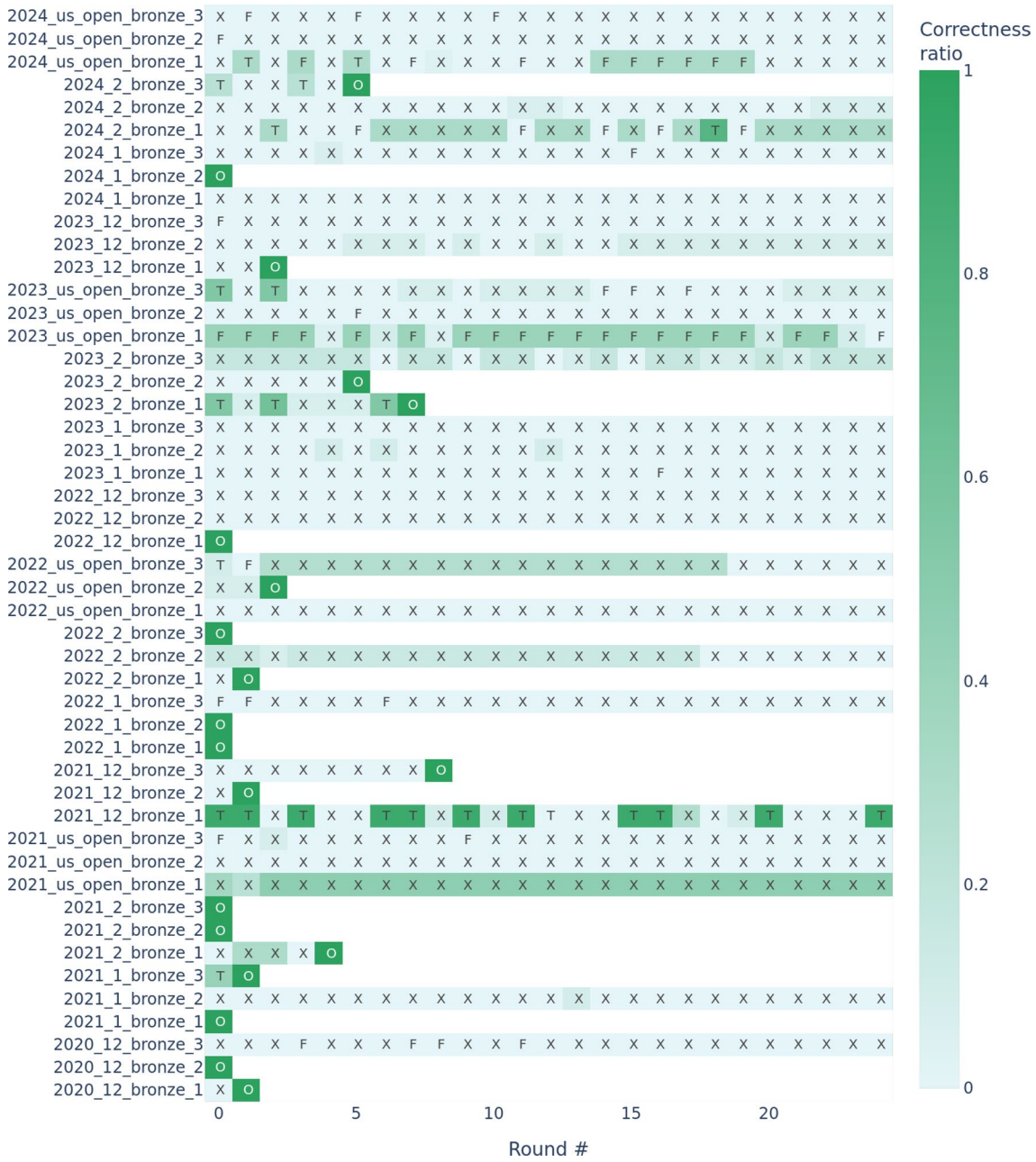


Fig. 2. GPT-4o experimentation for bronze problems.

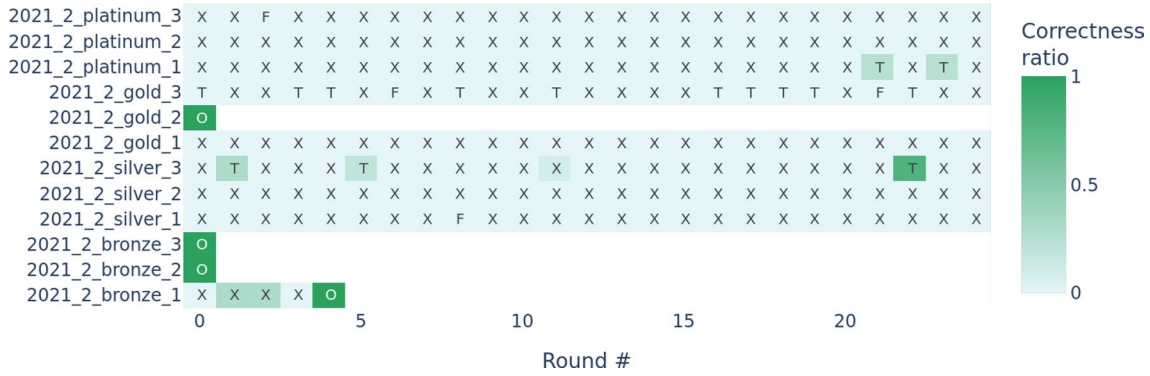


Fig. 3. GPT-4o experimentation for four different levels.

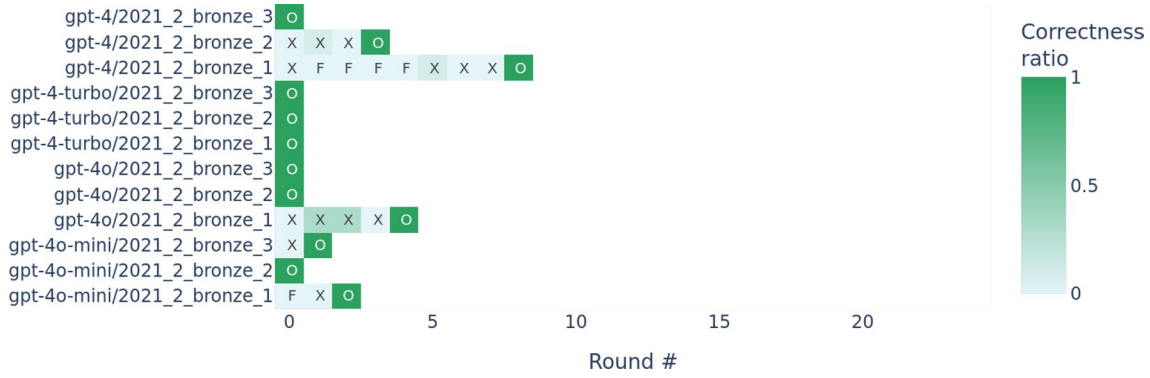


Fig. 4. Experimentation with different GPT models.

Model	Per 1M Input Tokens	Per 1M Output Tokens
GPT-4o-mini	$0.15	$0.60
GPT-4o	$5.00	$15.00
GPT-4-turbo	$10.00	$30.00
GPT-4	$30.00	$60.00

Table 3. The API price of different models.

Next, to collect a wider set of experimental data for our environmental impact study, we tested the GPT-4o model against the four different levels of the 2021 February competition, as shown in Fig. 3. Although it successfully solved the first trial for one Gold problem, GPT-4o overall failed to fully solve all three problems at any other level. Again, this showcases the difficulty in finding a representative test set that we can reliably use to compare the environmental impact of AI and human programmers.

Thus, we shifted our focus to how various GPT models tackle the same Bronze problems. As shown in Fig. 4, we performed identical experiments using different models: GPT-4o-mini, GPT-4o, GPT-4-turbo, and GPT-4. Their API prices, which are believed to correlate with the size of each model, are detailed in Table 3.

For this set of experiments, all four models were able to correctly solve the same set of problems in most trials. Figure 5 summarizes the results. Because model behavior is not deterministic, we ran each model on the same set of problems 10 times, allowing up to 100 rounds of the correction process per problem. Each row of plots represents the results of one model across three problems, while each column represents the results of one problem across the four models. In each subplot, a bar shows the energy consumption of each round in one of the 10 experiments. For example, GPT-4o solved the first problem after one round in the first experiment (Exp 0), but failed to solve the same problem after 100 rounds in the third experiment (Exp 2). Overall, energy consumption depends strongly on both the model used and the number of correction rounds required to produce correct results.

Another noteworthy observation that we can make is the environmental impact of successful and failed runs. For example, out of 10 experiments that we did with GPT-4o to solve the `21_2_bronze_1` question, failed runs (`Exp 2, 4, and 9`) consumed 8.16 times more energy on average than successful runs (`Exp 0, 1,`



Fig. 5. Energy consumption of different GPT models (the average of min and max energy estimation for each round).

`3, 5, 6, 7, and 8`). If we exclude `Exp 7`, which solved the problem after 77 rounds, the ratio becomes 53 times. This again shows that failed runs are very expensive, and human programmers need good judgment on the capability of LLM services.

One interesting question is whether the energy estimation from Ecologits can accurately represent diverse service scenarios in which service operators perform various optimizations for large-scale serving through dynamic batching of user requests. Batching helps reduce energy consumption by amortizing the weight loading overhead of LLMs for the feedforward network of a transformer block, but it does not help to reduce the overhead of self-attention computation because context is unique to each user's request[29]. Furthermore, self-attention computation overhead scales quadratically with the sequence length. Figure 6 shows the input and output sequence length of our experiments. As shown in the figure, due to the long failure logs used for debugging for multi-round correction, the input context length is very long. A recent study[30] shows that, when the input sequence length is long, the energy consumption is dominated by prefill computation, and the energy consumption of the prefill computation is not sensitive to the size of batch, when the input sequence length is long. Thus, we believe that per-query estimates from Ecologits should be a reasonable order-of-magnitude approximation for the multi-round correction process that we evaluated, even for a large-scale service. Also, as previously mentioned, we had to restrict the failure log to the last ten rounds of conversations due to the current context length limitation of today's LLM services. Given that LLM service providers are increasingly supporting longer context lengths, we believe that the energy consumption of future coding agents will not be significantly amortized through batching.

Using these four models on the same problem set, we illustrated the energy consumption associated with AI-based programming in Fig. 7. Please note that this is the average of 10 experiments. Unfortunately, the same model could sometimes solve all three problems, yet other times fail to produce correct output even after 100 rounds, highlighting the difficulty of achieving reproducible and deterministic measurements.

As mentioned in the previous section, if a human programmer solves this set using their computer for four hours, they will consume 0.3 kWh for a laptop and a monitor. We note again that these values are representative averages; real-world consumption can vary significantly depending on hardware efficiency, system setups, and power management settings. On the other hand, Ecologits estimated that the most energy-efficient model, GPT-4o-mini, consumed between 0.13 kWh (minimum) and 0.40 kWh (maximum). In contrast, the least efficient model, GPT-4, consumed between 3.41 kWh (minimum) and 13.24 kWh (maximum). These values represent the averaged ranges across 10 experiments.

Furthermore, we evaluated the greenhouse gas emissions, as shown in Fig. 7. The most environmentally friendly scenario, GPT-4o-mini, emits between 0.08 kg and 0.24 kg of CO_2. In comparison, the worst-case scenario, GPT-4, emits between 2.03 kg and 7.83 kg of CO_2.



Fig. 6. Input / output sequence length distribution.

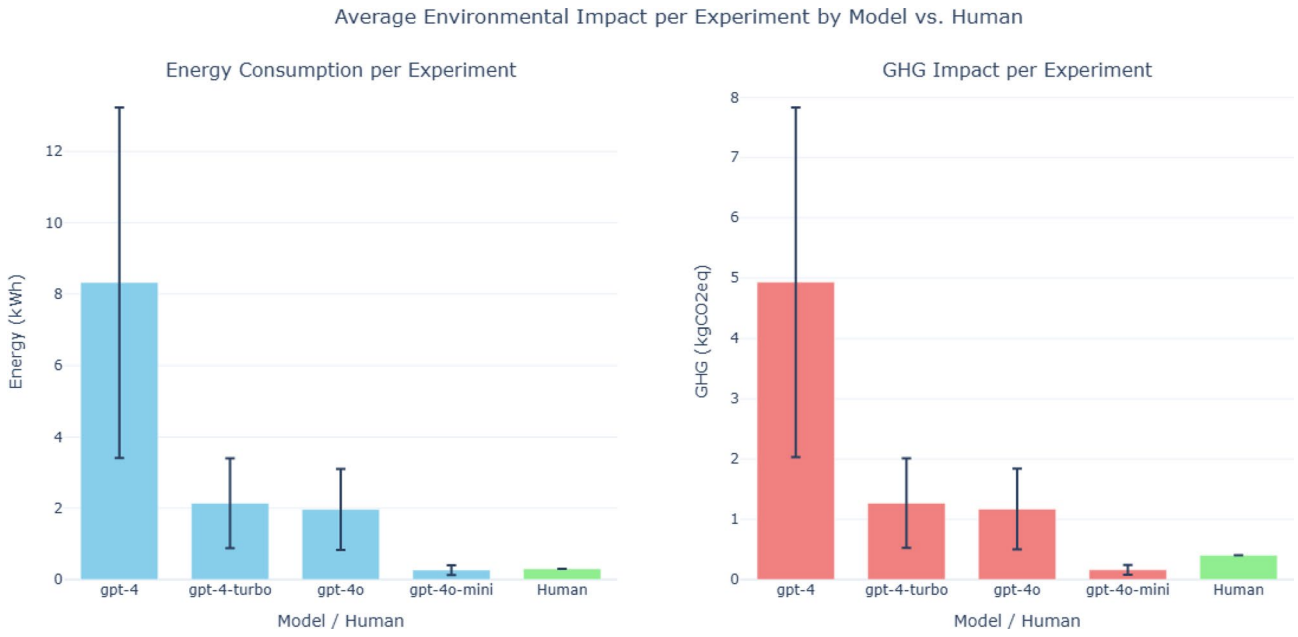


Fig. 7. The estimated energy consumption and greenhouse gas emission (average of 10 experiments, including an experiment where a model failed to produce correct output even after 100 rounds of trials).

Overall, our results show that the most environmentally friendly model, GPT-4o-mini, emits between 20% and 59% of the greenhouse gas emissions of human programmers. On the other hand, the least environmentally friendly model, GPT-4, emits 5 to 19 times more greenhouse gas than human programmers. We should keep in mind that these results stem from a perfect scenario where AI programmers are able to fully solve the problem, which is fairly unlikely according to our trials. For more complex or open-ended problems, we would expect a greater number of iterations and therefore higher AI emissions. Conversely, tasks suited to LLM strengths may result in lower impacts.

It should be noted that, for other USACO competitions, the GPT model ran for significantly more rounds due to its inaccurate output, as shown in Fig. 2. Consequently, its environmental impact tends to be far greater than that of its human counterpart, often failing to produce correct results, as illustrated in the figure.

Finally, Fig. 8 shows the relationship between API price, the number of inference calls, and their corresponding greenhouse gas emissions. As shown in the chart, the number of inference calls needed to finish a given task has a higher correlation to the amount of greenhouse gas emission than the API price itself. This suggests that

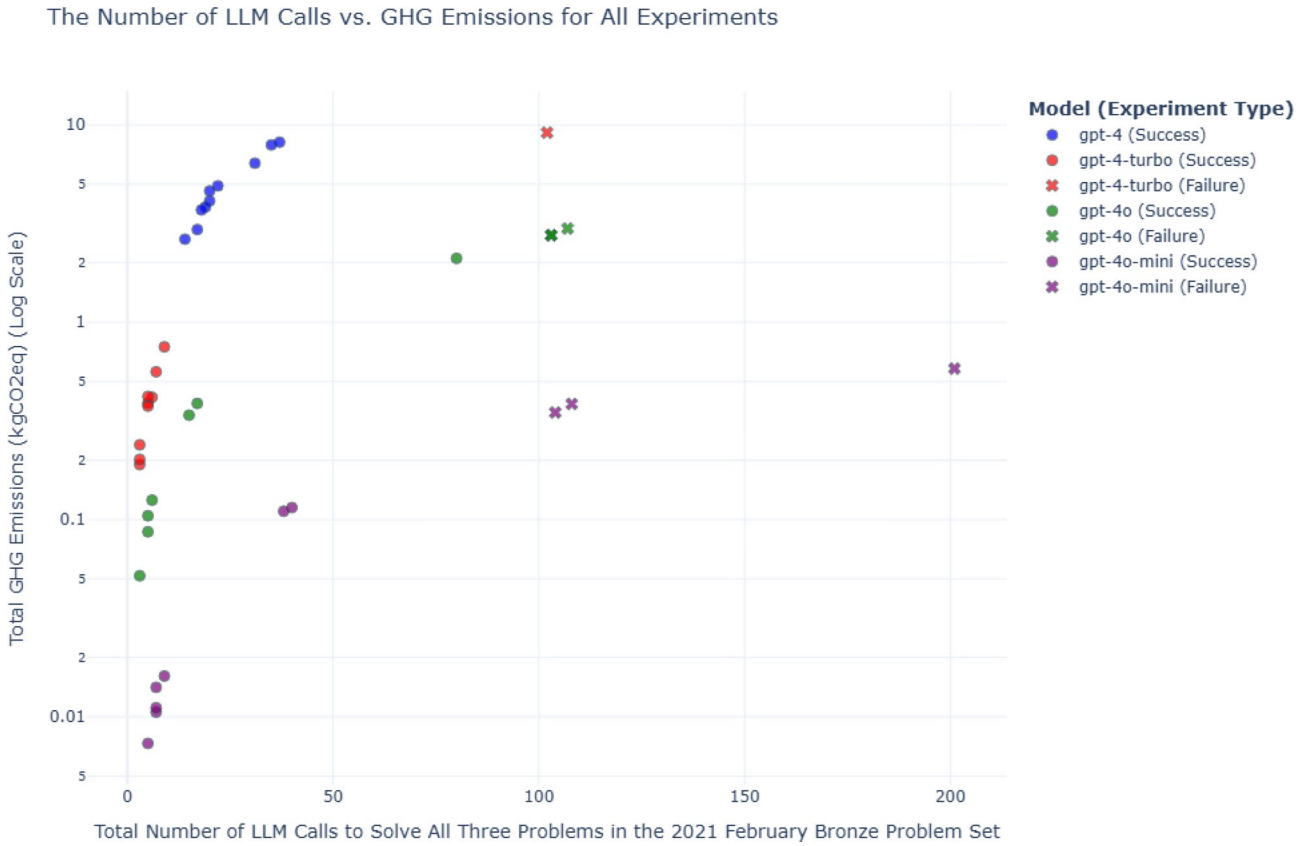


Fig. 8. The relationship among API price, the number of inference calls, and greenhouse gas emissions.

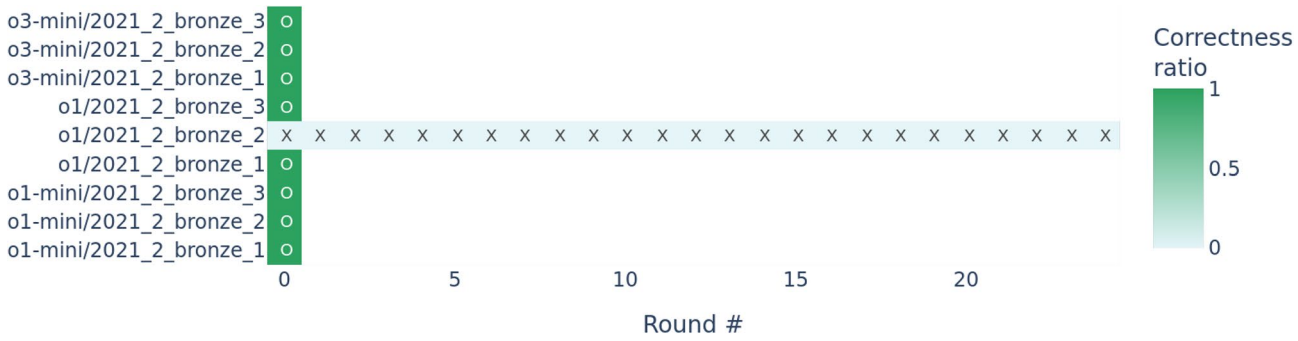


Fig. 9. Experimentation with reasoning models.

using neither the cheapest model nor the most expensive model is an environmentally better choice. Rather, we need to find the appropriate model for a given task so that the number of trials needed to achieve the task can be minimized. It also suggests that failed experiments, even after 100 trials, are environmentally unfriendly. For example, failed trials with GPT-4o-mini often emit more GHG than successful executions of GPT-4o or GPT-4-turbo. This further implies that harder problems, which we were unable to use in our experiments, may have even more significant environmental impacts.

Discussion

Reasoning models and agents

In addition to traditional LLM models, we also attempted to evaluate recent reasoning models. As shown in Fig. 9, the reasoning models have done a better job in terms of their functional accuracy compared to the traditional large language models, although they are not perfect either and can often be costly.

As is widely known, these reasoning models are based on test-time compute, a process that involves making significantly more LLM calls for a single query through various strategies such as repeated sampling, self-correction, or tree search[31]. Such expensive operations are reflected in their API pricing as shown in Table 4.

Model	Per 1M Input Tokens	Per 1M Output Tokens
o1-mini	$1.10	$4.40
o1	$15.00	$60.00
o3-mini	$1.10	$4.40

Table 4. The API price of reasoning models.

Because one API call can trigger an unspecified number of parallel or sequential executions of LLMs of unknown size, precise environmental impact estimates are challenging. In practice, however, the strong correlation we observed between total inference calls and emissions (Fig. 8) implies that reasoning models are likely to have higher footprints. Therefore, Fig. 8 serves as a qualitative pointer underscoring the critical need for transparency in these new architectures.

Furthermore, emerging agentic AI services also call LLMs multiple times for a single query, often assigning different roles to the model throughout the process[32]. Like reasoning models, these agentic systems obscure the structure of underlying LLM calls, which makes estimation of environmental impact uncertain. As the industry increasingly adopts multi-call architectures for user queries, there is an urgent need that we prioritize the AI service development in an environmentally sustainable manner.

Limitations

While we attempt to objectively compare the environmental impact of AI against that of humans, there exist several caveats that readers should be aware of. First, we have performed our study using competitive programming tasks, which may not reflect typical day-to-day programming challenges. Such tasks are narrowly focused on algorithmic correctness and do not capture aspects like collaboration with others, which could affect real-world energy use. Correctness is not the only important factor in software development; readability, extensibility, and modularity are also crucial to ensure that code can be understood, reused, and extended by others. Second, the success rate of LLM-based programming was not high enough for us to conduct a study across a wide set of test suites. Instead, we had to assume that, for unsuccessful executions, LLMs would waste significantly more energy. Third, due to the nature of decoding strategies in LLMs[33], outputs are not always deterministic, making experiments difficult to reproduce consistently. Running a sufficiently large number of trials to stabilize results statistically is both costly and environmentally unfriendly. Note that our multi-round correction process already runs an LLM up to 100 times for a given programming task. Fourth, the environmental impact reported by Ecologits is an estimation rather than a direct measurement. Ideally, each provider would publish the environmental impact of their LLM services. Fifth, training cost was not included in our estimates because it is unclear how to quantify it—training often involves many failed trials, and it is difficult to determine how those costs are amortized without knowing the exact usage volume of the deployed models. Also, our estimates assume a future scenario where LLM inference is widespread and training costs can be amortized. Sixth, estimating the environmental impact of proprietary AI models is not straightforward. As our study suggests, the quality of an AI model strongly affects the number of inferences required in the multi-round correction process. For this reason, using production-quality models was unavoidable in our evaluation. However, this raises questions about the validity of estimating the impact of proprietary systems. Ecologits estimates model size indirectly, using observed execution performance such as output token rates and comparing them against open-source models. While useful, this approach may not fully capture real usage, particularly at large scales where many requests are batched together. A more recent study[34] attempts to account for such scenarios, but it also faces limitations because it relies on restricted information from an AI workload analysis service[35], such as fixed token counts. We also tried to estimate energy consumption using the method suggested in that study, building an interpolation function across dynamically changing token counts. However, due to the limited number of data points, we found that this approach could not reliably estimate the energy consumption of GPT-4, like in the original work. As a community, we still lack a complete understanding of how to measure these impacts accurately. We exclude office energy, commuting, daily necessities like food, and personal device embodied impacts for humans, just as we exclude provider offices for AI. These are organizational overheads, not task-marginal impacts. In vibe coding, humans are still present in the work environment, so their normal lifestyle emissions continue regardless of whether AI writes the code, and should not be allocated to the programming task.

Suggestion

Given that our study found model selection to be critical in minimizing environmental impact, we propose that API providers leverage their operational data to develop routing mechanisms that optimize for both performance and sustainability. Since these services handle requests from a wide range of users and already monitor quality metrics, we believe they are well-positioned to analyze this data and build algorithms that can select the most appropriate LLM for a given task. Such routing could minimize the number of iterations needed to correct the code, which our results show is the largest driver of emissions. This change would prioritize both efficiency and environmental sustainability.

Conclusion

This study is, to our knowledge, the first-ever correctness-controlled analysis to quantitatively compare the environmental impacts of a human programmer and an AI programming service. Our findings reveal that AI models emit up to 19 times more CO_2eq than human programmers, demonstrating a far more severe trade-

off between computational efficiency and environmental impact than previously understood. By focusing on competitive programming tasks with objective correctness metrics, we address the research gap left by prior work, which lacked direct, apples-to-apples comparisons of AI and human coding emissions. This disparity underscores the urgent need for strategic regulation and optimization of hybrid intelligence.

Additionally, we found that the imperfect output of AI services often requires numerous iterations of queries, significantly increasing emissions compared to those generated by a human programmer. This supports our hypothesis that the output quality must be considered when comparing the environmental footprint of AI versus humans.

Furthermore, while model size influences emissions, our findings indicate that greenhouse gas output is more closely tied to the number of LLM inference calls. With smaller models lacking accuracy and larger ones being carbon-intensive, selecting the appropriate model for each task is crucial to minimizing environmental impact. Future work should expand this comparison to more diverse programming contexts and include real-world software development workflows. Ultimately, these findings signal an urgent need for industry-wide accountability in ensuring sustainable AI deployment.

Data availability

Source code and test data available at https://github.com/JelloCello30/humanvsaicoding.

Received: 13 April 2025; Accepted: 14 October 2025
Published online: 10 November 2025

References

1. Kaplan, J. et al. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361 https://doi.org/10.48550/arXiv.2001.08361 (2020).
2. Luccioni, A. S., Viguier, S. & Ligozat, A.-L. Estimating the carbon footprint of BLOOM, a 176b parameter language model. *J. Mach. Learn. Res.* **24**(253), 1–15. https://doi.org/10.48550/arXiv.2211.02001 (2023).
3. Patterson, D. et al. Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350https://doi.org/10.48550/arXiv.2104.10350 (2021).
4. Tomlinson, B., Black, R. W., Patterson, D. J. & Torrance, A. W. The carbon emissions of writing and illustrating are lower for AI than for humans. *Sci. Rep.* **14**(1), 3732. https://doi.org/10.17605/OSF.IO/YHTMQ (2024).
5. Github Copilot - Your AI pair programmer. Accessed 13 Apr 2025. https://github.com/features/copilot (2025).
6. Cursor - The AI Code Editor. Accessed 13 Apr 2025. https://www.cursor.com/en (2025).
7. Vibe coding. Accessed 13 Apr 2025. https://en.wikipedia.org/wiki/Vibe_coding (2025).
8. Falkiewicz, P. Self-correcting code generation using multi-step agent. Accessed 27 Sept 2025. https://deepsense.ai/resource/self-correcting-code-generation-using-multi-step-agent/?hl=en-US (2025).
9. Fu, B., Salman, A., Álvarez-Otero, S., Sui, J. & Razzaq, M. G. A. The dynamic connectedness between environmental attention and green cryptocurrency: Evidence from the COVID-19 pandemic. *J. Organ. End User Comput.* **36**(1), 1–18. https://doi.org/10.4018/JOEUC.338215 (2024).
10. Xia, M. et al. From carbon capture to cash: Strategic environmental leadership, AI, and the performance of US firms. *J. Organ. End User Comput.* **36**(1), 1–24. https://doi.org/10.4018/JOEUC.359892 (2024).
11. Wang, Y., Wang, T., Wang, W. & Hou, Y. Collaborative application of deep learning models for enhanced accuracy and prediction in carbon neutrality anomaly detection. *J. Organ. End User Comput.* **36**(1), 1–25. https://doi.org/10.4018/JOEUC.340385 (2024).
12. USA Computing Olympiad. Accessed 15 Sept 2024. https://usaco.org/ (2024).
13. OpenAI. OpenAI developer platform. Accessed 15 Sept 2024. https://platform.openai.com/ (2024).
14. Buscemi, A. A comparative study of code generation using ChatGPT 3.5 across 10 programming languages. Accessed 15 Sept 2024. arXiv preprint arXiv:2308.04477https://doi.org/10.48550/arXiv.2308.04477 (2023).
15. SWE-bench Team. SWE-bench leaderboards. Accessed 27 Sept 2025. https://www.swebench.com/index.html (2025).
16. Liu, Z., Tang, Y., Luo, X., Zhou, Y. & Zhang, L. F. No need to lift a finger anymore? assessing the quality of code generation by ChatGPT. *IEEE Trans. Softw. Eng.* https://doi.org/10.48550/arXiv.2308.04838 (2024).
17. Aygün, E. et al. An AI system to help scientists write expert-level empirical software. arXiv preprint arXiv:2509.06503 (2025).
18. Chen, M. et al. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374https://doi.org/10.48550/arXiv.2107.03374 (2021).
19. Rincé, S. & Banse, A. Ecologits. Evaluating the environmental impacts of generative AI. *J. Open Source Softw.* **10**, 7471. https://doi.org/10.21105/joss.07471 (2025).
20. Ecochain Technologies. Cradle-to-Gate in LCA: What is it and how does it work? Accessed 27 Sept 2025. https://ecochain.com/blog/cradle-to-gate-in-lca/ (2025).
21. Ecologits. LLM inference - Ecologits. Accessed 27 Sept 2025. https://ecologits.ai/latest/methodology/llm_inference/#assumptions-and-limitations (2025).
22. Dell Inc. Dell XPS 13 laptop - Thin and Lightweight Laptop | DELL USA. Accessed 27 Sept 2025. https://www.dell.com/en-us/shop/dell-laptops/xps-13-laptop/spd/xps-13-9350-intel-laptop (2025).
23. Dell Inc. Dell pro 27 Inch Plus Monitor P2725h - FHD IPS display | DELL USA. Accessed 27 Sept 2025. https://www.dell.com/en-us/shop/dell-pro-27-plus-monitor-p2725h/apd/210-bmfj/monitors-monitor-accessories (2025).
24. Dell Inc. Product Carbon Footprint - XPS 13 9350. Accessed 27 Sept 2025. https://www.delltechnologies.com/asset/en-us/products/laptops-and-2-in-1s/technical-support/xps-13-9350-pcf-datasheet.pdf (2024).
25. Dell Inc. Life Cycle Assessment - P2725h. Accessed 27 Sept 2025. https://www.delltechnologies.com/asset/en-us/products/electronics-and-accessories/technical-support/p2725h-monitor-pcf-report.pdf (2024).
26. U.S. Environmental Protection Agency. Greenhouse gas equivalencies calculator. Accessed 27 Sept 2025. https://www.epa.gov/energy/greenhouse-gas-equivalencies-calculator (2024).
27. Sibelga. How much power does a computer use? and how much CO_2 does that represent? Accessed 15 Septt 2024. https://www.energuide.be/en/questions-answers/how-much-power-does-a-computer-use-and-how-much-co2-does-that-represent/54/ (2024).
28. Ritchie, H., Rosado, P. & Roser, M. CO_2 and greenhouse gas emissions. *Our world in data*. https://ourworldindata.org/co2-and-greenhouse-gas-emissions (2023).
29. Recasens, P. G. et al. Mind the memory gap: Unveiling GPU bottlenecks in large-batch lLM inference. arXiv preprint arXiv:2503.08311 (2025).
30. Fernandez, J. et al. Energy considerations of large language model inference and efficiency optimizations. arXiv preprint arXiv:2504.17674 (2025).

31. Li, Z.-Z. et al. From system 1 to system 2: A survey of reasoning large language models. arXiv preprint arXiv:2502.17419https://doi.org/10.48550/arXiv.2502.17419 (2025).
32. Shavit, Y. et al. *Practices for governing agentic AI systems* (Research Paper, OpenAI, 2023).
33. Holtzman, A., Buys, J., Du, L., Forbes, M. & Choi, Y. The curious case of neural text degeneration. arXiv preprint arXiv:1904.09751https://doi.org/10.48550/arXiv.1904.09751 (2019).
34. Jegham, N., Abdelatti, M., Elmoubarki, L. & Hendawi, A. How hungry is AI? benchmarking energy, water, and carbon footprint of LLM inference. arXiv preprint arXiv:2505.09598 (2025).
35. Artificial Analysis. Artificial analysis: AI model & API providers analysis. Accessed 15 Aug 2025. https://artificialanalysis.ai/ (2025).

Author contributions
N.W. performed the study, wrote the main manuscript, and prepared all the figures.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to N.H.W.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by-nc-nd/4.0/.

© The Author(s) 2025