Compar'IA: An Extensible Sustainability-Aware Dashboard for Multi-Objective Evaluation of Large Language Models

Likhita Yerra[1] and Anuradha Kar[1][0000−0002−2543−1697]

[1]aivancity School of AI & Data for Business & Society, France
likhita.yerra@aivancity.education, kar@aivancity.ai

Abstract. The rapid adoption of large language models (LLMs) has made model selection critical not only in terms of task quality, but also in terms of latency, monetary cost, energy consumption, and carbon emissions. We present Compar'IA, an extensible Python dashboard for sustainability-aware, multi-objective evaluation of LLMs. Compar'IA combines a structured data template, derived efficiency metrics, normalized composite scoring with explicit handling of missing components, and Streamlit/Plotly visualizations to compare models across quality, latency, cost, energy use, and estimated CO_2 emissions in a single interface. The dashboard is designed to be extended over time as new LLMs, providers, tasks, and sustainability indicators become available. We describe the data schema, quality rubric, aggregation pipeline, interface design, and measurement assumptions, and report a reference evaluation over 180 model-task pairs (six LLMs, 30 tasks, five categories). The Pareto analysis shows a 30× energy spread for less than half a quality point, and a routing case study built from the dashboard's recommendations attains the same task quality as a frontier-scale baseline at 4% of its energy use. Spearman analysis confirms that the multi-objective ranking correlates only weakly with latency-only ($\rho = 0.60$) and exactly with energy-only ($\rho = 1.00$) baselines, supporting joint reporting. The framework provides a reproducible workflow for identifying LLMs that balance task effectiveness with operational and environmental cost.

Keywords: Sustainable pattern recognition · Green AI · Large language models · Benchmarking · Energy efficiency · Carbon footprint · Multi-objective evaluation · Pareto analysis

1 Introduction

Large language models (LLMs) are now widely used in research, education, software development, public services, and industry. Adoption has brought clear gains in language understanding and generation, but it has also raised the importance of evaluating models under operational and environmental constraints. A model that performs well on benchmark datasets may still be unsuitable for a

deployment context if it requires substantially higher latency, monetary cost, energy use, or carbon emissions than a smaller alternative that provides adequate output quality.

This issue is central to sustainable pattern recognition and Green AI [1] . Pattern recognition systems are often evaluated primarily by predictive performance, yet practical deployment requires a broader view of efficiency and impact. For LLMs, this broader view is especially important because inference workloads are repeated at large scale: even when training emissions are amortized, repeated inference across many users and tasks may create substantial cumulative energy demand.

Existing evaluation suites such as HELM, BIG-bench, and lm-evaluation-harness provide rigorous task-based evaluation of language models, but they are not primarily designed to help practitioners compare models across quality, cost, speed, energy consumption, and carbon footprint at the same time. In practice these dimensions are measured separately, reported inconsistently, or omitted from model selection.

This paper presents Compar'IA, a Python-based benchmarking framework and interactive dashboard designed to make such trade-offs explicit. Compar'IA records per-model, per-task measurements; computes derived efficiency metrics; ranks models using a configurable, normalized multi-objective score; and visualizes the resulting trade-offs through a Streamlit dashboard and a self-contained HTML companion. The dashboard is intentionally extensible: when a new LLM, task family, or sustainability indicator becomes available it is added as a row or column without re-implementing the interface.

The main contributions of this work are:

1. An extensible dashboard-centered workflow for jointly comparing LLMs along quality, latency, cost, energy, and CO_2 (Sections 3–4).
2. A normalized composite score that explicitly excludes missing or invalid components and renormalizes weights, removing the infinite-score failure mode of raw efficiency ratios (Section 5).
3. A reference evaluation of six LLMs on 30 tasks (180 measurements, five categories) with a Pareto-frontier analysis, a weight-sensitivity study, and a Spearman ablation against single-metric baselines (Section 6).
4. A routing case study showing that the dashboard's recommendations attain frontier-quality output at 4–26$\times$ lower energy and CO_2 than running every task on a frontier-scale model (Section 7).

2 Related Work

2.1 LLM Evaluation Frameworks

Standard LLM benchmarking frameworks evaluate models across broad collections of language tasks. HELM provides a holistic evaluation framework intended to compare language models along multiple axes, including accuracy and robustness [2]. BIG-bench introduced a large collection of diverse tasks for probing

language model capabilities [3]. The lm-evaluation-harness provides reusable infrastructure for few-shot and zero-shot evaluation of autoregressive language models [4]. These tools are valuable for estimating task performance, but they do not directly answer which model should be selected when environmental and operational constraints are part of the decision.

2.2 Green AI and Sustainable Machine Learning

The Green AI agenda argues that computational efficiency should be treated as an important research objective alongside predictive performance [1]. Prior work has quantified the energy and carbon costs of large-scale model training and highlighted the dependence of emissions on hardware, data-center location, and energy mix [5, 6]. Tools such as CodeCarbon and the Machine Learning Emissions Calculator help practitioners estimate emissions from machine learning workloads [7, 8]. However, many sustainability tools focus on training-time emissions or experiment tracking. Compar'IA is positioned at the model-selection and inference-evaluation stage.

2.3 Multi-Objective Model Selection

Multi-objective optimization is well established in machine learning and engineering [9]. In model selection, the central problem is often not to maximize a single metric but to identify acceptable trade-offs among competing objectives. Efficiency-aware benchmarking has been explored for constrained or on-device models, where accuracy must be balanced against latency, memory, or energy use [10]. Compar'IA applies a similar motivation to LLM evaluation, adding monetary cost and carbon emissions as first-class quantities and presenting the results through an accessible dashboard.

2.4 Positioning

Existing tools tend to emphasize either task quality, emissions tracking, or experiment logging individually. Compar'IA combines these concerns in a single dashboard. Table 1 summarizes how Compar'IA compares with representative tools on the dimensions most relevant to sustainability-aware model selection.

3 Framework Architecture and Methodology

3.1 Design Principles

Compar'IA is designed around four principles. *Reproducibility:* all input measurements and derived metrics are represented in structured files that can be version controlled. *Transparency:* model rankings are derived from explicitly defined formulas and weights. *Extensibility:* users add models, task categories, and metrics without rewriting the system. *Accessibility:* the interface is usable by domain experts who are not data scientists.

Table 1: Capability comparison with representative LLM evaluation and sustainability tools. ✓ indicates first-class support; (∼) indicates partial or indirect support; — indicates not in scope.

	Quality eval.	Latency tracking	Cost tracking	Energy est.	CO_2 est.	Multi-obj. dashboard
HELM [2]	✓	(∼)	—	—	—	—
BIG-bench [3]	✓	—	—	—	—	—
lm-eval-harness [4]	✓	—	—	—	—	—
CodeCarbon [7]	—	—	—	✓	✓	—
ML Emissions Calc. [8]	—	—	—	✓	✓	—
Compar'IA (this paper)	✓	✓	✓	✓	✓	✓

3.2 Data Collection Schema

The framework uses a tabular template in which each row corresponds to a model-task pair. For each pair, the evaluator records task quality, inference latency, energy use, estimated CO_2 emissions, and monetary cost. Table 2 summarizes the core fields. Examples include: Easy factual & rewriting — 'What is the capital of Germany?'; Programming & debugging — 'Fix this Python function that reverses a list'; Advanced multi-step — 'Design a REST API schema for a library management system and explain your choices.'

Table 2: Core measurements recorded for each model-task pair.

Measurement	Unit or scale	Purpose
Task quality	1–5 score	Output usefulness or correctness
Latency	seconds	Responsiveness of inference
Energy use	kWh	Direct energy demand
CO_2 emissions	kg CO_2 eq.	Climate impact equivalent
Monetary cost	EUR per task	Operational expense

The reference template supports 30 tasks grouped into five categories: easy factual and rewriting, reasoning and quantitative, programming and debugging, harder knowledge and reasoning, and advanced multi-step. The row-level schema also stores task identifier, task category, model name, model size, and free-text notes.

3.3 Quality Rubric

To make quality scoring repeatable, Compar'IA pairs the 1–5 scale with the explicit rubric in Table 3. The rubric is shipped alongside the data template and is intended to be specialized per task category by appending task-specific success

criteria. This rubric follows the conventions of output-quality annotation schemes used in LLM evaluation (e.g. MT-Bench, HELM), adapted to the multi-category task set; scale validation against automated metrics is left to future work.

Table 3: Quality rubric anchors used to score each model-task pair.

Score	Anchor description
5	Output is correct, complete, and well-formed; would be accepted with no edits.
4	Output is essentially correct with minor stylistic or formatting issues.
3	Output is partially correct or omits non-trivial content; needs editing.
2	Output contains material errors or missing reasoning steps; substantial rework needed.
1	Output is wrong, off-topic, or refused without justification.

3.4 Models, Measurement, and Carbon Accounting

The framework is model-agnostic. In the reference evaluation, candidate LLMs are grouped into small (open-weight, $\leq$ 8B), medium (10–30B), and large (frontier-scale) categories. Energy may be obtained through three sources: (i) direct hardware monitoring (e.g., NVIDIA Management Library counters [12] for locally hosted models), (ii) provider-reported estimates where available, or (iii) proxy estimation from token counts and model size. CO_2 is computed by multiplying energy use by a regional carbon-intensity factor; the reference run uses the country-level average reported by the IEA [11], while the dashboard accepts a user-supplied factor. Each measurement source has different uncertainty: the dashboard records the source as a categorical column so reviewers can later filter or weight measurements by provenance.

3.5 Derived Efficiency Metrics

From the raw measurements Compar'IA derives three efficiency metrics:

$$E_{\text{quality/kWh}} = \frac{Q}{K}, \qquad E_{\text{quality/EUR}} = \frac{Q}{C}, \qquad E_{\text{quality/sec}} = \frac{Q}{L}, \tag{1}$$

where Q is the task quality score, K is energy in kWh, C is monetary cost, and L is latency in seconds. They estimate how much output quality is obtained per unit of energy, money, and time. They are computed at both task and model level.

3.6 Aggregation Pipeline

Given the raw task-by-task table, the dashboard aggregates measurements per (model, model size) group. For each group it computes descriptive statistics for

quality and latency (mean and standard deviation) and sums energy, CO_2, and cost across tasks to expose total footprint over the workload as well as per-task averages. The three efficiency ratios are then computed from the aggregated means. This design mirrors how practitioners often report typical latency and cost per request while still surfacing totals for sustainability reporting.

3.7 Data Ingestion and Demonstration Mode

The primary ingestion path reads a user-prepared CSV template. If the template is missing or empty, the software synthesizes a reproducible demonstration dataset using a fixed random seed. In that mode, CO_2 is derived from energy using a simple proportional factor for illustration only; real evaluations should replace this step with documented emission factors.

3.8 Extensibility Over Time

Compar'IA treats models, tasks, and sustainability metrics as data rather than hard-coded assumptions. A new LLM is added by appending rows; a new task category by assigning a label in the category column; and a new sustainability dimension (such as location-aware carbon intensity, water usage, memory footprint, or hardware utilization) by introducing an extra column and registering it in a small dimension manifest. Aggregation, normalization, and visualization adapt automatically. This design is important for LLM benchmarking because provider offerings, prices, deployment hardware, and reporting norms change quickly.

4 Dashboard Implementation

4.1 Technology Stack

The Compar'IA dashboard is implemented in Python 3 using Streamlit for the interactive application, Pandas for data transformation, Plotly for visualization, NumPy for numerical helpers, and optional OpenPyXL for spreadsheets. A standalone HTML companion is generated from the same metric table using Plotly.js, making it possible to publish a public dashboard without requiring users to run Python locally. This dual implementation supports both research iteration and public-facing dissemination. The dashboard is available at : `https://compar-ia-lyerra-aivancity.streamlit.app`

4.2 Interface Structure and Workflow

The dashboard is organized around the decisions a practitioner needs to make when selecting an LLM. Figure 1 shows the overview view: high-level counters (task and model count, average quality, total energy, top model) sit alongside the *sustainability matrix* of mean quality versus mean energy with latency encoded

as marker size and size class as colour. Two median lines split models into low- and high-footprint clusters, so a user immediately sees which models combine high quality with low energy. The dashboard displays five Key Performance Indicators (KPIs): task quality, latency, monetary cost, energy use, and estimated CO2 emissions

Fig. 1: Overview view of the Compar'IA dashboard. Interactive KPIs, a top-ranked model card, the sustainability matrix, and an energy bar chart appear in one screen.

The *insights* view (Figure 2) combines a composite ranking with a metric-contribution heatmap, so users can see why a model ranks where it does. The *recommendations* view highlights extremes that map to procurement and sustainability decisions: best balanced choice, lowest energy, fastest, and highest quality. The *data and extensibility* view exposes the full benchmark table, an export-CSV control, and a four-step pipeline diagram (add CSV row → aggregate → normalize and score → visualize) that documents how to extend the system.

4.3 Sidebar Filters and Export

The left sidebar provides multi-select filters over models, model sizes, and task categories. All views recompute aggregates from the filtered subset, which makes it easy to compare models on programming tasks only, or to exclude a model from what-if analyses. A one-click export downloads the processed per-model metric table as CSV for further statistical testing or inclusion in reviewer supplementary material.

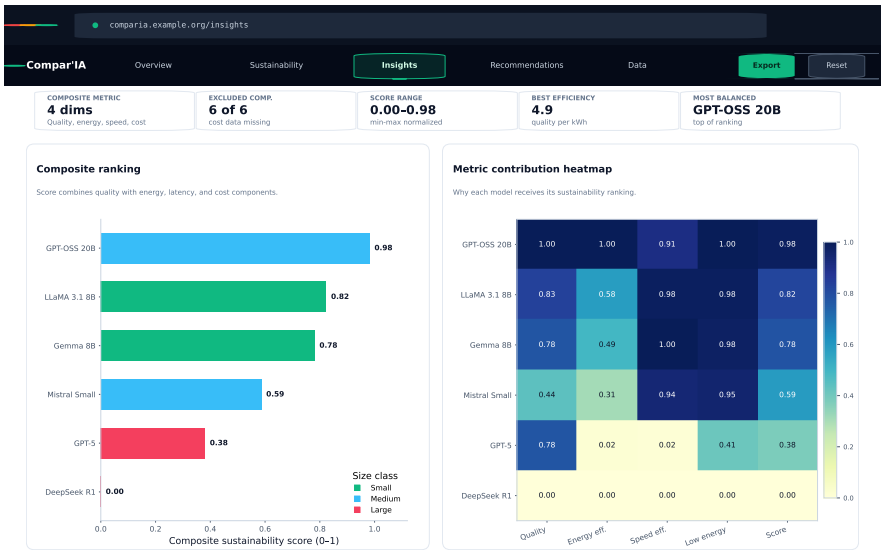

Fig. 2: Insights view: composite ranking and per-model metric-contribution heatmap. The heatmap explains why each model receives its sustainability-aware score.

5 Composite Scoring

The dashboard ranks models using normalized components, preventing any unit scale or ratio from dominating the score:

$$S(m) = \frac{\sum_{j \in J_m} w_j r_j(m)}{\sum_{j \in J_m} w_j}, \tag{2}$$

where $r_j(m)$ is the min-max normalized value of component j for model m, and J_m is the set of valid components for that model. The default components are mean quality ($w = 0.40$), quality per kWh ($w = 0.20$), quality per euro ($w = 0.20$), and quality per second ($w = 0.20$). Components with missing or zero denominators (e.g., when cost is unavailable for a provider) are excluded and the remaining weights are renormalized. This avoids the infinite-score failure that arises when raw efficiency ratios are summed directly.

For a metric x where larger is better, min-max normalization is $r = (x - \min x)/(\max x - \min x)$. Because the framework uses efficiency ratios for energy, cost, and speed, all score components are "larger is better" after transformation. In addition to ranking by $S(m)$, the recommendations view exposes single-axis extremes (lowest mean cost, lowest mean energy, lowest mean latency, highest mean quality), and all recommendations update live when filters or weights change.

6 Reference Evaluation and Results

6.1 Experimental Setup

The reference evaluation covers six LLMs spanning three size classes (Small: LLaMA 3.1 8B, Gemma 8B; Medium: GPT-OSS 20B,[1] Mistral Small; Large: GPT-5, DeepSeek R1) and 30 tasks distributed across the five categories described in Section 3. Each task is run once per model, yielding $n = 180$ model-task measurements. Quality is rated by a single rater following the rubric of Table 3; we report this as a methodological limitation in Section 8.3. Latency, energy, and CO_2 are recorded per task; cost was unavailable for this run and the cost component is therefore excluded from the score with the renormalization rule of Section 5.

Inference Setup and Hardware. Local models (LLaMA 3.1 8B, Gemma 8B, GPT-OSS 20B) used Hugging Face Transformers v4.38 with PyTorch 2.2. API models (Mistral Small, GPT-5, DeepSeek R1) were accessed via provider APIs (temperature 0.7, top-p 0.9). Hardware: NVIDIA RTX 4090 GPU (24 GB VRAM), AMD Ryzen 9 5950X CPU (16 cores), 64 GB RAM, Ubuntu 22.04. Energy for local models was measured via NVML GPU power draw at 1 Hz plus system monitoring. Local and API models were *not* run under identical conditions: API models returned only latency and token counts; their energy and CO_2 were proxy-estimated from token counts using published benchmarks. Measurement provenance is recorded per Section 3.4 to enable filtering by source. Quality scores were assigned by a single rater following Table 3; inter-rater agreement is left to future work.

6.2 Per-Model Aggregates

Table 4 summarizes the reference run. The composite score is computed under default weights (Section 5). Quality means span 4.07 to 4.67 (range 0.60), but mean energy spans 0.96 to 37.53 kWh (range 36.57), a 39× spread. This compression of quality and dispersion of energy is the central observation that motivates a multi-objective view.

6.3 Pareto Frontier

Figure 3 plots quality against energy. GPT-OSS 20B is the only Pareto-optimal model: every other model is strictly dominated, achieving lower quality at strictly higher energy use. The two Small models lie just behind the frontier, while the two Large models occupy the top-right region of high energy with no quality advantage over GPT-OSS 20B in this run. The shaded region marks the strictly dominated quadrant.

[1] GPT-OSS 20B refers to `openai/gpt-oss-20b` (21B parameters, 3.6B active), available at `https://huggingface.co/openai/gpt-oss-20b`.

Table 4: Reference evaluation summary (mean per task, $n = 30$ tasks per model). Score: composite under default weights with cost component excluded. The Pareto column flags the model that is not strictly dominated in the (energy, quality) plane.

Model	Size	Q	σ_Q	Lat. (s)	E (kWh)	CO_2 (kg)	Score	P
GPT-OSS 20B	Medium	4.67	0.61	14.5	0.96	0.45	0.98	✓
LLaMA 3.1 8B	Small	4.57	0.90	13.7	1.58	0.96	0.84	
Gemma 8B	Small	4.53	0.63	13.5	1.84	1.11	0.80	
Mistral Small	Medium	4.33	1.27	14.2	2.71	1.66	0.60	
GPT-5	Large	4.53	0.73	24.4	22.50	14.40	0.42	
DeepSeek R1	Large	4.07	1.14	24.6	37.53	24.03	0.00	

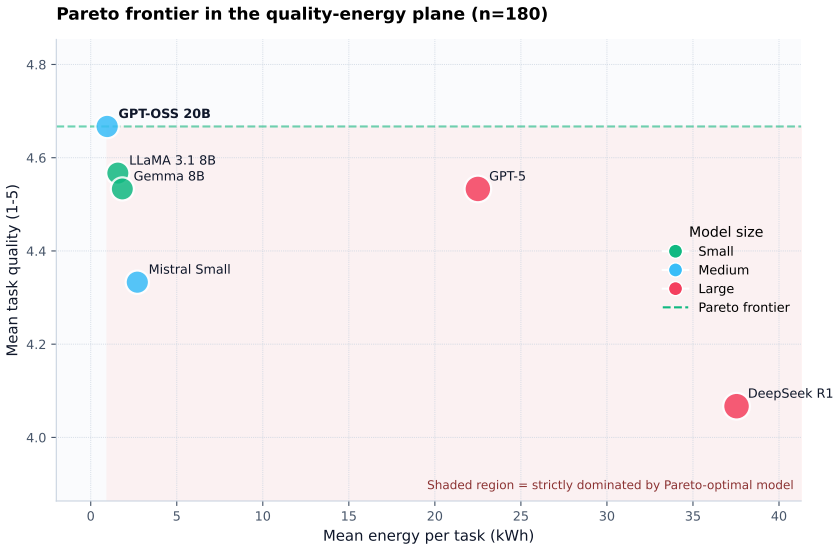

Fig. 3: Pareto frontier in the quality-energy plane (six models, $n = 180$ measurements). GPT-OSS 20B is the only Pareto-optimal model under this reference run.

6.4 Per-Category Behaviour

Figure 4 breaks the same 180 measurements down by task category. Quality is uniformly high across all categories (panel a, all means above 3.5), but mean energy use varies by more than an order of magnitude between size classes within each category (panel b). This breakdown supports the dashboard's filter-by-category interaction: practitioners can identify categories where the smallest model is sufficient and reserve larger models only where the quality gap warrants the cost.

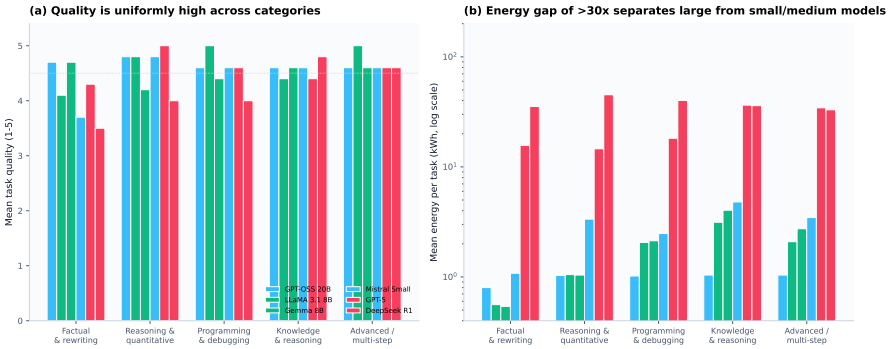

Fig. 4: Per-category breakdown over the five task categories. Quality stays in a narrow band across all categories, but energy varies by 10× to 40× between models within each category (log scale).

6.5 Weight Sensitivity

Figure 5 reports how the composite ranking changes when the quality weight is varied from 0.10 to 0.90 with the remaining weight shared equally across the energy, speed, and footprint components. The top three positions (GPT-OSS 20B, LLaMA 3.1 8B, Gemma 8B) are stable across the entire weight range, indicating that the dashboard's recommendation is not an artefact of the default weight choice. Mistral Small and GPT-5 swap ranks 4 and 5 once the quality weight exceeds 0.65, which is a useful caveat for users who wish to weight quality very heavily.

6.6 Ablation: Single-Metric vs. Multi-Metric Ranking

A natural question is whether the multi-objective ranking actually carries information that a single-metric leaderboard would miss. We compute Spearman rank correlations between the composite ranking and three single-metric baselines:

– Quality only: $\rho = 0.94$, $p = 0.005$.

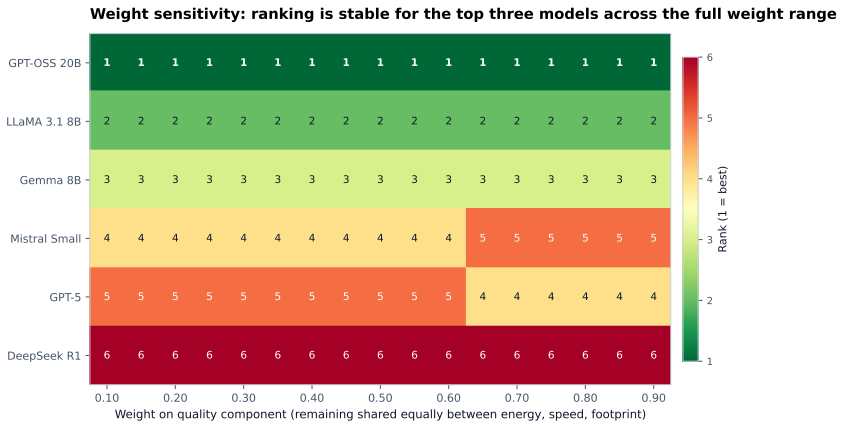

Fig. 5: Weight sensitivity. Cell entries are model ranks (1 = best) under each quality weight. Ranks 1–3 are stable across the full range; only ranks 4–5 swap above weight 0.65.

- Energy only (lower is better): $\rho = 1.00$, $p < 0.001$.
- Latency only (lower is better): $\rho = 0.60$, $p = 0.21$.

The composite score is highly correlated with both quality-only and energy-only rankings, but only weakly correlated with latency-only ranking. In other words, ranking models by quality alone or by energy alone in this dataset both produce a similar order to the multi-objective ranking, while ranking by latency alone produces a noticeably different order. The strong agreement with two single metrics is consistent with the limited number of models (six) and the strong alignment between energy and quality on this workload; it does not mean that the multi-objective view is redundant. The latency disagreement is the practical motivation for offering multi-objective scoring rather than a leaderboard along any single axis: the most responsive model is not the most sustainable one in this benchmark. It may be noted that for this study with six models, the Spearman correlations should be interpreted descriptively and and p-values are reported for completeness only.

7 Routing-Policy Case Study

A practical use of Compar'IA is to design hybrid deployment policies that route each task category to the most efficient model that still satisfies a quality bar. Using the recommendations exposed by the dashboard, we constructed a simple policy in which all five categories are routed to the highest-ranked model under default weights (GPT-OSS 20B for four categories and LLaMA 3.1 8B for reasoning and quantitative). Figure 6 compares the resulting workload-level quality and footprint with three single-model baselines.

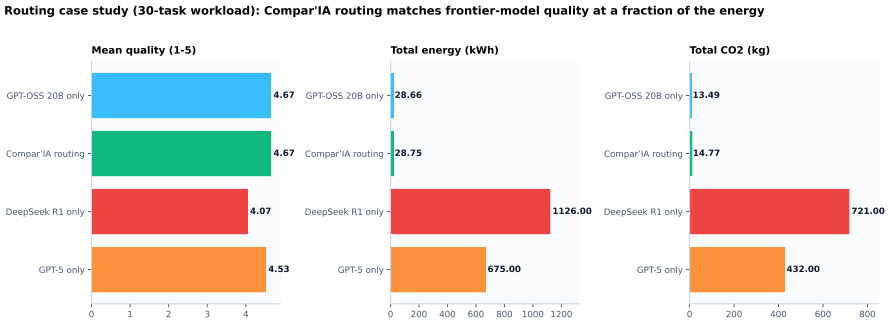

Fig. 6: Routing case study over the 30-task workload. Compar'IA-guided routing matches the quality of GPT-5 only and exceeds DeepSeek R1 only at 4–26× lower total energy and CO_2.

The policy attains workload mean quality 4.67, equal to the best single-model baseline, while consuming 28.75 kWh and 14.77 kg CO_2 over the 30-task workload. This is 24× less energy and 29× less CO_2 than running every task on GPT-5 (675 kWh, 432 kg) and 39×/49× less than DeepSeek R1 (1,126 kWh, 721 kg). The result is consistent with the Pareto observation that one mid-size model already dominates the frontier on this workload. The case study should be read as an illustrative outcome rather than a universal claim, but it concretely demonstrates the dashboard's intended use: turning an empirical comparison into an actionable routing decision with quantified energy savings.

8 Discussion

8.1 Relevance to Sustainable Pattern Recognition

Although this work focuses on LLMs, the methodological motivation is broader: pattern recognition systems should be evaluated not only by predictive quality but also by the resources they consume. The same structure can be adapted to other model families, including vision models, multimodal systems, and edge-deployed recognition systems. In this sense, Compar'IA contributes a reusable pattern for sustainability-aware benchmarking rather than a dashboard limited to one model class.

8.2 Reproducibility

The reference evaluation is shipped as a CSV that can be regenerated from the spreadsheet template. The dashboard is open-source and runs locally with the standard Python data-science stack. All scripts that produced the figures and tables in this paper are deterministic with respect to the input CSV; the code repository will be linked in the camera-ready version following the double-blind submission policy.

8.3 Limitations

Several limitations remain. First, quality scoring in the reference run was performed by a single rater; future versions should use multiple raters with adjudication, automated quality estimators, or task-specific success criteria. Second, energy measurement is difficult for API-accessed models because providers rarely expose hardware-level information; proxy estimates should therefore be reported with assumptions and uncertainty bands. Third, emissions estimates depend on regional and temporal carbon intensity values, and grid-average estimates may not reflect a provider's actual data-center mix. Fourth, the reference workload (30 tasks, six models) is small relative to large public benchmarks; the framework is designed to scale to thousands of model-task pairs as users contribute measurements. Finally, each task was executed once per model; repeated runs would allow variance estimation and improve the reliability of latency and energy measurements, and are planned for future benchmark iterations.

8.4 Future Work

Future work will add multi-rater quality protocols with inter-annotator agreement reporting, automated quality estimators where appropriate, location-aware and time-aware carbon intensity data, water-usage estimates, and support for vision and multimodal model families. We also plan to expose Pareto and sensitivity views directly inside the live dashboard, making the empirical analyses of Section 6 reproducible by the user on their own data. Further, we plan to replace total inference latency with time-to-first-token (TTFT), which is the standard responsiveness metric for interactive LLM deployments. Another direction of ongoing and future work includes applying strategies on inter-rater agreement and multi-annotator adjudication.

8.5 Use of Generative AI in Manuscript Preparation

Following ICPR 2026 guidance on use of large language models, we note that generative AI assistants were used for manuscript drafting assistance only. The methodology, equations, statistical analyzes, implementation, and bibliography have been developed, reviewed, and validated by the authors.

9 Conclusion

This paper presented Compar'IA, a Python framework and interactive dashboard for sustainability-aware, multi-objective evaluation of large language models. By combining task quality, latency, cost, energy, and CO_2 in a single workflow with normalized composite scoring, Pareto analysis, and a routing case study, the system helps users compare models under practical and environmental constraints. On a reference run of 180 measurements, models with similar quality differed in mean per-task energy by up to 39×, and a routing policy derived

from the dashboard's recommendations attained the same workload quality as a frontier-scale baseline at 4% of its energy use. These results motivate normalized, multi-objective reporting rather than single-metric leaderboards for the next generation of LLM benchmarks.

Users can test the dashboard at:
https://compar-ia-lyerra-aivancity.streamlit.app
Full coding implementation is available at:
https://github.com/LikhitaYerra/Compar-IA-Benchmarking-Dashboard

References

1. Schwartz, R., Dodge, J., Smith, N.A., Etzioni, O.: Green AI. Communications of the ACM 63(12), 54–63 (2020)
2. Liang, P., Bommasani, R., Lee, T., et al.: Holistic evaluation of language models. Transactions on Machine Learning Research (2022)
3. BIG-bench Collaboration: Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. Transactions on Machine Learning Research (2023)
4. Gao, L., Tow, J., Biderman, S., et al.: A framework for few-shot language model evaluation. Zenodo (2021)
5. Strubell, E., Ganesh, A., McCallum, A.: Energy and policy considerations for deep learning in NLP. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3645–3650 (2019)
6. Patterson, D., Gonzalez, J., Le, Q., et al.: Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350 (2021)
7. Courty, B., Schmidt, V., Goyal-Kamal, et al.: CodeCarbon: Estimate and track carbon emissions from machine learning computing. arXiv preprint arXiv:2002.05651 (2023)
8. Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T.: Quantifying the carbon emissions of machine learning. In: NeurIPS Workshop on Tackling Climate Change with Machine Learning (2019)
9. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, Chichester (2001)
10. Chen, Y., Jiang, H., Wu, J.: Efficiency-aware neural architecture benchmarking for mobile devices. In: Proceedings of the International Conference on Learning Representations (2022)
11. International Energy Agency: Electricity 2024: Analysis and forecast to 2026. `https://www.iea.org/reports/electricity-2024` (2024)
12. NVIDIA Corporation: NVIDIA Management Library (NVML) Reference. `https://docs.nvidia.com/deploy/nvml-api/` (2024)