Amortizing AI Training Carbon Footprint: Challenges, Limitations, and a Path Forward

NOMAN BASHIR[1], FABIO GRIMALDI[1], ARDESHIR RAIHANIAN MASHHADI[1], MICHAEL TAPTICH[2], KOMMY WELDEMARIAM[2], ARAVIND SRINIVASAN[2], RYAN BRADLEY[1], ALEXIS BATEMAN[1], [1]Amazon Web Services, [2]Amazon, Inc., USA

Allocating the one-time carbon cost of training a large AI model across the inference requests it serves is an open methodological problem with no standardized solution. The choices made in boundary definition, functional unit selection, and lifetime forecasting can alter reported per-request emissions by an order of magnitude, undermining any comparison across models. We decompose this problem into three challenges: defining the emission boundary (what counts as training carbon), selecting a functional unit (how to measure inference work), and forecasting lifetime usage (how many units the model will ultimately serve). The three differ in kind yet interact, since boundary and lifetime choices compound and the functional unit constrains forecasting complexity. Boundary scope and functional unit selection are addressable through improved measurement and standardized reporting; lifetime usage, by contrast, is not deterministic and for open-source models may never be precisely measurable. We outline a starting point to tackle each challenge, arguing that imperfect but consistent measurement today is preferable to waiting for a consensus that may never arrive.

1 Introduction

Training a frontier AI model can emit hundreds of tons of CO_2e in a single run [23, 30]. Once trained, that model may serve billions of inference requests over many months. As AI deployment scales [8, 25] and regulatory frameworks demand per-request carbon disclosure [11], organizations need a principled way to allocate this one-time training cost across the useful work the model performs. Our scope is precisely that allocation: distributing training carbon across inference requests over a model's operational lifetime. The stakes are practical: identical training runs reported under different methodological choices yield per-request figures that diverge by an order of magnitude, rendering cross-model comparison meaningless.

The problem has a natural analogue in hardware lifecycle assessment (LCA). Manufacturing emissions from a server are divided by its expected useful life, and the resulting per-unit embodied cost is added to operational emissions incurred during use. Standards such as ISO 14044 [18] and the GHG Protocol [34] codify boundary definitions, allocation rules, and reporting requirements for this process. (The GHG Protocol's corporate standard reports annual emissions without amortization; the product-level standard and ISO 14067 are the relevant frameworks for per-unit lifecycle allocation.)

Training carbon is structurally analogous: an upfront investment distributed across useful work. At its simplest:

$$\text{Amortized carbon per request} = \frac{\text{Total training emissions}}{\text{Total lifetime requests served}}$$

where the numerator captures the cost of producing the model and the denominator captures total useful work delivered. While hardware LCA converged on workable conventions despite significant heterogeneity [16, 18], the AI training case introduces dimensions of complexity that those conventions do not address.

First, the numerator is poorly defined. For hardware, the embodied carbon boundary is well understood: extraction, manufacturing, transport, and end-of-life, each with established protocols [18, 34]. For AI training, no consensus exists on what constitutes *training emissions*. The final run accounted for 37% of total project emissions for BLOOM [23], 50% for OLMo [28], and 4% for the Moshi speech model [21]. The remainder: R&D experiments, failed runs, hyperparameter tuning, evaluation. Reporting practices further diverge on idle server power (32% of BLOOM's energy beyond dynamic GPU draw [23]), infrastructure overhead, and embodied hardware carbon [15, 31]. The numerator is therefore not a fixed model property but a function of where the boundary is drawn.

Second, the denominator is difficult to define. Hardware embodied carbon is amortized over time (server-hours), which accumulates orthogonally to workload [15]. Training carbon must be amortized over units of work, and these units are heterogeneous: a single-token classification and a 4,000-token conversation both count as one "request," yet differ by an order of magnitude in compute cost [19]. No single unit (requests, tokens, FLOPs) captures resource consumption faithfully across modalities, making the denominator itself a consequential methodological choice.

Third, the denominator is difficult to forecast and may not be observable at all. Server lifetimes are stabilized by capital expenditure and physical installation friction [24]. Model replacement carries lower switching cost, producing more volatile lifetimes [4]: a model may be retired within weeks or persist for years depending on when a superior alternative emerges. A foundation model may be deployed by thousands of organizations with usage patterns invisible to the trainer [6]; for open-source releases, total global usage may never be measurable [33].

These differences indicate that ISO 14040/44 [17, 18] does not by itself resolve training amortization: the standards define the procedure but leave AI-specific allocation choices under-specified. We apply and extend LCA principles to this setting rather than proposing a method outside LCA. Recent work applies LCA-style thinking to AI systems [12, 23, 31] and tracks training emissions in real time [7, 20], but focuses on quantifying the numerator and treats amortization as straightforward division with an assumed denominator. The methodological questions we raise remain unaddressed as a coherent problem. We make three contributions:

(1) We decompose amortization into three distinct challenges: the emission boundary, the functional unit, and lifetime forecasting.
(2) We analyze interactions among the three and show how choices in one dimension constrain the others.
(3) We outline a practical starting point for each, arguing that consistent measurement with limitations is preferable to inaction.

Author's Contact Information: Noman Bashir[1], Fabio Grimaldi[1], Ardeshir Raihanian Mashhadi[1], Michael Taptich[2], Kommy Weldemariam[2], Aravind Srinivasan[2], Ryan Bradley[1], Alexis Bateman[1], [1]Amazon Web Services, [2]Amazon, Inc., Seattle, WA, USA.

2 Emission Boundaries

The numerator in the amortization formula requires a clear definition of what counts as "training emissions." ISO 14044 [18] requires system boundaries to encompass all processes contributing significantly to environmental impact; any exclusion must fall below a relevance threshold (typically 1% of energy or mass) and be justified. For physical products, Product Category Rules (PCRs) under ISO 14025 [16] codify these boundaries by product type: they specify which lifecycle stages a valid Environmental Product Declaration (EPD) must include, so that organizations reporting for the same category use identical rules and their declarations are comparable. No equivalent exists for AI model training. Each organization draws its own boundary, producing emissions figures that cannot be meaningfully compared. We propose a boundary framework along four dimensions (Figure 1) and argue that the AI community needs category-level rules analogous to PCRs.

2.1 Boundary Dimensions

The first three dimensions (activity, power, lifecycle) determine *what* energy and material flows cross the system boundary. The fourth (geography) determines *how* those flows convert to carbon emissions; in LCA terminology it belongs to impact assessment (LCIA) rather than the lifecycle inventory (LCI), but we include it as it produces up to 7.5× variation in reported training emissions.

The *activity boundary* determines which development phases contribute to the numerator. The final training run is the minimum scope: it directly produces the artifact. Training energy for the final run alone spans roughly two orders of magnitude across recent frontier language models, from under 500 MWh for BLOOM to over 200,000 MWh for the largest reported runs [10]. BLOOM's final run accounted for 37% of total project emissions [23]; OLMo's for approximately 50% [28]; and the Moshi speech model, built from scratch at a new lab, just 4% [21]. The multiplier scales with architecture novelty: a well-understood architecture requires fewer preliminary experiments, while a novel one demands extensive exploration. Under ISO 14044's criteria, R&D activities consuming 50–96% of total energy exceed any reasonable exclusion threshold.

The *power boundary* determines which energy draws are included within a given activity. Dynamic GPU power is the commonly reported figure, yet BLOOM's data reveals it represented just 54.5% of total system consumption [23]: idle servers added 32%, shared infrastructure 13.5%. Reporting only dynamic GPU power understates actual energy use by nearly half. Most published training emission figures report only dynamic GPU energy, sometimes with a PUE multiplier [35]; these should be viewed as lower bounds.

The *lifecycle boundary* determines whether hardware manufacturing emissions are included. An operational-only boundary corresponds to gate-to-gate in LCA terminology. Embodied carbon from GPU and server manufacturing adds 20–30% to lifecycle emissions [15]; independent lifecycle accountings place the embodied share at approximately 14.5% of total emissions [27]. The correct treatment of embodied carbon depends on what decision the number serves. For reporting and disclosure, under the current guidelines, the boundary should include embodied carbon because it represents part of the true lifecycle cost of producing the model. For real-time operational scheduling, embodied emissions are sunk costs: routing decisions cannot influence them, and including them in optimization metrics conflates avoidable with unavoidable costs, producing suboptimal outcomes [4]. In this paper, training amortization is an accounting exercise, and embodied carbon is included in the numerator. Once included, the embodied figure follows a provenance ladder: most reporters rely on public, secondary hardware LCA, while hyperscalers maintain primary, infrastructure-specific models. Neither is comparable across organizations until a hardware PCR standardizes how embodied carbon is computed.

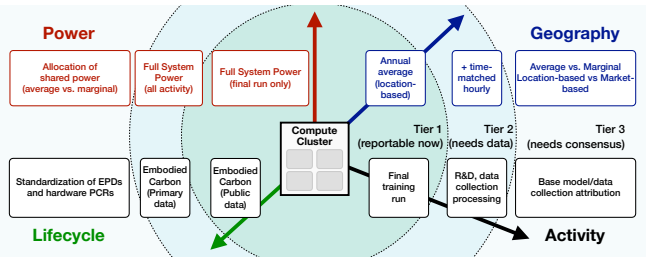

Fig. 1. Boundary framework for training emission accounting. Each dimension (activity, power, lifecycle, geography) advances through three barriers: Tier 1, reportable now; Tier 2, needs organizational data; Tier 3, needs cross-organizational consensus. Tier 3 is the agenda for a category rule.

The *geography boundary* captures the carbon intensity of the electricity consumed. BLOOM trained on a 57 gCO_2eq/kWh grid in France and emitted 25 tCO_2eq; GPT-3 trained on a 429 gCO_2eq/kWh grid and emitted 502 tonnes [23, 30]. The 7.5× difference in grid intensity accounts for the largest share of this gap. The regional composition of chip supply adds a second-order effect: where wafers are fabricated and assembled shifts embodied emissions, since fab energy mixes vary by location. Organizations should report energy consumption and carbon emissions separately, enabling readers to distinguish efficiency improvements from grid-mix advantages.

2.2 Toward Product Category Rules for AI Training

A full PCR for AI model training is premature: the community lacks consensus on scope, and the empirical base is too thin to mandate a single methodology. However, the absence of any shared rules produces the incomparability problem described above. We outline a tiered reporting structure that parallels the progressive conformity model used in EPD programs, where organizations begin with simplified LCA and advance toward full PCR compliance as data and methods mature. The tiers are defined by the kind of barrier that separates a practitioner from wider coverage and comparability, and all dimensions advance through the same three barriers (Figure 1).

Tier 1 (reportable now). Each dimension at its most readily available scope: the final training run (activity), full-system power for that run including dynamic, idle, and PUE overhead (power), embodied carbon from public secondary-data estimates (lifecycle), and annual-average location-based grid intensity (geography). For cloud users, this data is already available through provider sustainability consoles (e.g., AWS Sustainability Console [2], Google Cloud Carbon Footprint [13], Azure Emissions Impact Dashboard [26])[1]. For on-premises operators, it requires system-level power monitoring,

[1] In GHGP, Scope 1 covers direct emissions from owned sources, Scope 2 the emissions from purchased electricity, and Scope 3 indirect value-chain emissions such as hardware manufacturing. This sense of "scope" is distinct from the boundary scope above.

regional grid-intensity data, and embodied-carbon estimates from published hardware LCA studies [31]. All exclusions beyond this scope must be documented with their estimated magnitude.

Tier 2 (needs data). Each dimension extended with data an organization can collect but most do not yet instrument: R&D and data-processing energy beyond the final run (activity), full-system power measured across that entire activity rather than the final run alone (power), embodied carbon from primary, infrastructure-specific LCA rather than secondary estimates (lifecycle), and time-matched hourly grid intensity (geography). The activity extension is the R&D multiplier, the ratio of total project GPU-hours to final-run GPU-hours, obtainable by aggregating across all logged experiments in tools such as MLflow or Weights & Biases. Empirical data places this multiplier at 2–3× for well-understood architectures and up to 25× for novel model types [21, 23]. These extensions require internal process and supplier disclosure, but no industry-wide agreement.

Tier 3 (needs consensus). Each dimension reaches a point where no amount of private data resolves it, because the open question is which method is correct: how to attribute base-model training and shared data to downstream fine-tuners and deployers (activity), how to allocate shared cluster power across concurrent tenants (power), how to standardize the computation of hardware embodied carbon through EPDs and PCRs (lifecycle), and whether to apply average or marginal and location-based or market-based emission factors (geography). Tier 3 is thus the agenda for a category rule: the choices that must be settled collectively before figures become reportable and comparable across organizations. It also encompasses open-source models where inference is decentralized and the aggregate denominator is unobservable, so proxy-based estimation or declared scope limitations become necessary.

This structure separates three obstacles. Tier 1 is limited by whether an organization chooses to report; Tier 2 by data within reach of an organization and its suppliers; Tier 3 by the absence of shared conventions, the frontier of the problem.

2.3 Challenges and Open Problems

Even with the tiered structure above, three boundary problems remain unresolved and require further community attention.

Double-counting across organizational boundaries. When a foundation model is fine-tuned by multiple downstream organizations, each may independently account for the base model's full training emissions [5, 33]. If every deployer of Llama 3 reports the full training cost, the same emissions are counted thousands of times in aggregate. If no deployer claims them, the emissions disappear from all reporting. This is an allocation problem for a shared upstream process, and ISO 14044 provides three procedures for such cases, none of which map cleanly to the foundation model setting. Physical causality (allocate by mass or energy flow) would require measuring each deployer's share of total inference volume, which is unobservable for open-source models. Economic allocation (allocate by revenue share) would require revenue data that organizations will not disclose. System expansion (expand the boundary to avoid allocation entirely) fails because downstream products are not substitutes for each other, making the expanded boundary ill-defined. The GHG Protocol's guidance on joint venture allocation (equity share or control approach) offers a potential analogue, though its applicability to decentralized open-source deployment remains untested. This is the same structure as Scope 3 supply-chain accounting for intermediate products, where an upstream producer publishes a product carbon footprint that downstream buyers inherit. Emerging data-space initiatives such as Catena-X provide a transport mechanism for such inherited footprints, and analogous publication of model training footprints could let downstream deployers account for inherited cost without each re-claiming the full total. Candidate approaches include: (a) only the model trainer reports training emissions, with downstream deployers reporting only their fine-tuning and inference costs; (b) the trainer publishes a per-download or per-API-call allocation factor; or (c) each deployer reports a declared fraction with explicit methodology. None of these has been adopted as a convention, and the problem remains open.

Temporal boundary: when does training end? Production models rarely have a clean boundary between training and deployment. Many undergo continuous fine-tuning, reinforcement learning from human feedback (RLHF), safety training, or periodic refreshes that consume additional compute. The boundary question is whether each update constitutes a new model (with separate amortization) or an extension of the original (with update costs added to the numerator and amortized over the remaining lifetime). Treating each update as a new model provides clean accounting boundaries but produces discontinuities in reported per-request emissions. Accumulating update costs into the original numerator provides continuity but requires tracking cumulative training cost over time and periodically recalculating the amortization rate. No standard addresses this question, and organizations must establish internal policies for consistency.

Geographic comparability. Training in a low-carbon region reduces the numerator without reducing energy consumption. From a carbon accounting perspective, this is correct: carbon metrics reflect actual atmospheric impact, and location choice is a legitimate lever for reducing it. A precedent exists in international shipping, where emissions are reported separately from national inventories because geographic attribution is complex and the activity spans jurisdictions. The tension arises when comparing models across organizations. Two identical training runs (same architecture, same data, same GPU-hours) can produce very different reported emissions depending solely on grid placement. The regional composition of chip supply adds a second-order effect: where wafers are fabricated and assembled shifts embodied emissions, since fab energy mixes vary by location. This matters far less than operational grid intensity, but practitioners comparing models built on different hardware supply chains should be aware of it. This complicates benchmarking and procurement decisions where the goal is to compare computational efficiency rather than grid carbon intensity. Reporting energy consumption alongside carbon emissions allows readers to separate computational efficiency from grid carbon intensity, but no current standard requires this dual reporting.

The boundary framework defines the numerator of the amortization formula; the next two sections address the denominator by specifying how to measure the functional unit (Section 3) and how to estimate the lifetime over which that unit accumulates.

3 Functional Units

In LCA, the functional unit defines the quantified performance of a product system that serves as the reference for all subsequent calculations [18]. Its purpose is to enable informed decision-making: by expressing environmental impact per unit of function delivered, stakeholders can compare alternatives and identify improvement opportunities. ISO 14044 requires the functional unit to be clearly defined, measurable, and consistent with the goal and scope of the study. For comparative assertions (e.g., "model A has lower per-request emissions than model B"), the standard further requires that the functional units be equivalent: they must represent the same function delivered to the same quality standard.

In product LCA, functional unit selection is among the most debated methodological choices. Transport studies illustrate the difficulty: "per vehicle-km," "per passenger-km," and "per tonne-km" each capture a different aspect of the service delivered, and the choice determines which products appear more efficient [18]. The same tension applies to AI inference. A functional unit based on requests treats all queries as equivalent. A unit based on tokens reflects output volume but not computational intensity. A unit based on device-seconds reflects resource consumption but is opaque to users. Each captures a different dimension of the service, and no single unit satisfies all three desirable properties simultaneously: measurability, fairness of allocation, and cross-model comparability. For this reason, different modalities may require different functional units (tokens for LLMs, time or pixel-count for vision, audio-seconds for speech), with compute-weighted normalization for cross-modal comparison. The unit is therefore not a neutral accounting detail but a decision lever derived from the goal and scope of the study: it determines which model or deployment appears preferable, and thus which choices the analysis steers stakeholders toward.

3.1 Candidate Units

Table 1 summarizes the candidate functional units. We evaluate each against the criteria that ISO 14044 implies for a well-defined functional unit: it must be measurable with existing instrumentation, it must correlate with the resource consumption being allocated (so that allocation is fair), and it must generalize across the product category (so that comparisons are meaningful).

Per-request accounting is the simplest unit and is already tracked for billing and SLA monitoring in most serving systems. Its weakness is uniform treatment: a single-token classification and a 4,000-token multi-turn conversation both count as one request, yet differ by 6× within a single model [19] and over 25× across task types [22], so it overcharges simple queries and undercharges complex ones.

Token-based accounting gives finer granularity for LLMs and aligns with existing API billing, but has two failure modes. First, tokens are a property of text-based LLMs and are undefined for image, video, audio, or multimodal models, so a token-based unit cannot serve as a cross-modality baseline. Second, token count does not capture reasoning cost: a 50-token chain-of-thought step can consume more resources than a 500-token simple completion, a disconnect that will grow as reasoning models become prevalent.

Compute-weighted units (device-seconds, FLOPs) correlate most strongly with resource use and generalize across modalities, since a device-second, the compute time on whatever processor serves the request (GPU, CPU, or other accelerator), measures the same quantity for text, image, or speech workloads. The costs are instrumentation, since per-request compute profiling is not always available, and opacity: end users reason in requests or tokens, not device-seconds, which complicates reporting.

Table 1. Candidate functional units evaluated against LCA criteria.

Unit	Measurability	Compute correlation	Cross-modal	Failure mode
Per request	High	Weak	Yes	6–25× complexity variation
Per output token	High	Moderate	No	Reasoning-heavy short outputs
Per input/output token	High	Modest–strong	No	Same; input adds noise
Per device-sec	Moderate	Strong	Yes	Opaque; hard to forecast
Per value added	Low	Variable	Yes	Subjective; incomparable

Value-based accounting (per dollar, per task, or per business outcome) parallels economic allocation in ISO 14044, distributing shared cost by the value of co-products. However, value is subjective and organization-specific, which makes comparison infeasible and introduces a circularity: a model's apparent carbon efficiency would depend on how profitably it is deployed rather than how efficiently it uses resources. Value-based units are therefore useful for internal strategic analysis but unsuitable for external disclosure.

3.2 The Fairness-Tractability Tradeoff

The candidate units reveal a basic tension. Units with stronger correlation to resource consumption produce fairer allocation (complex requests bear proportionally more training cost), but they are harder to forecast over a model's lifetime.

If the functional unit is requests, the lifetime forecast requires predicting total requests served. Request volume is the most commonly tracked metric and the easiest to forecast from historical adoption data. If the unit is tokens, the forecast additionally requires predicting average output length, which depends on use-case mix and may shift over time as the model is adopted for new applications. If the unit is device-seconds, the forecast requires predicting both volume and complexity distribution, a harder problem. This tradeoff is not unique to AI. In transport LCA, "per passenger-km" is fairer than "per vehicle-km" (it accounts for occupancy), but it requires forecasting ridership patterns rather than simply counting trips. The AI case is more extreme because the variance in per-unit resource usage (6–25×) exceeds what is typical in physical product systems.

The interaction also runs in the other direction: the choice of functional unit affects the sensitivity of the amortized figure to forecasting error. A per-request unit with a 20% forecasting error produces a 20% error in the amortized figure. A per-device-second unit with the same 20% volume error may produce a larger or smaller error depending on the complexity distribution.

3.3 A Starting Point

No single unit resolves the fairness-tractability tradeoff for all contexts. We recommend a primary unit selected for the deployment context, with a secondary unit reported for comparability.

For LLM-only deployments, token-based accounting (per output token or per I/O token pair) provides the best balance. It correlates moderately with compute, aligns with existing billing infrastructure, and is forecastable from usage data that organizations already collect.

Its failure mode (reasoning cost) is bounded and documentable. For multi-modal deployments or cross-organization comparison, compute-weighted units (per device-second) provide a modality-agnostic baseline. They require more instrumentation but produce fairer allocation across heterogeneous workloads.

In both cases, organizations should report the chosen unit, its rationale, and the known failure modes, following the transparency principle required for functional unit choice in comparative studies. Concretely, we suggest reporting three fields: the primary unit and its measured per-unit value, a secondary compute-weighted unit reported for cross-model comparison, and the instrumentation source for each. The minimum instrumentation is per-request token counts from the serving layer, which are already collected for billing, plus periodic compute-time sampling over a representative subset of requests. Adoption of compute-weighted units at scale is typically constrained not by measurement capability but by organizational readiness and operational supportability, which therefore belong in any assessment of how a unit choice will scale across the industry.

4 Lifetime Forecasting

The denominator in the amortization formula requires estimating total functional units over a model's operational lifetime. Unlike boundary definition and functional unit selection, which are measurement and standardization problems with tractable solutions, lifetime forecasting is a prediction problem under uncertainty at its core. No methodology can eliminate this uncertainty; it can only manage it. In LCA, this challenge is recognized as the problem of prospective (or ex-ante) assessment: estimating lifecycle impacts for a product whose use phase has not yet occurred [18]. The standard response is scenario analysis and sensitivity testing, not point estimation. We argue that the same principle applies here. We also note that the GHG Protocol does not prescribe amortization; its corporate standard reports annual emissions and does not allocate one-time training cost across future use. The challenge addressed here therefore fills a gap where current protocols are silent rather than restating an established procedure. The stakes are decision-relevant: lifetime estimates feed directly into the per-request figures that guide model selection, procurement, and disclosure.

4.1 The Uncertainty Landscape

Production model lifetimes vary widely. Table 2 summarizes publicly observable deployment timelines for several frontier models. These figures represent lower bounds: the model marked "ongoing" has not yet been deprecated, and its actual retirement date may extend further. Usage volumes are equally volatile. Traffic to ChatGPT increased 55% following the release of GPT-4o, retroactively shifting per-request training attribution by 40% [32]. A model that achieves unexpected adoption improves its amortization ratio; one that is quickly superseded allocates its training cost across fewer requests.

Three factors drive this uncertainty. First, model retirement is driven by external capability benchmarks rather than internal degradation. A model does not wear out; it becomes obsolete when a superior alternative emerges, a phenomenon well studied in the functional obsolescence literature [3]. The timing of competitor releases is not deterministic and cannot be predicted at training time. Second, usage volume depends on adoption dynamics that are unknown when training begins: market conditions, pricing decisions, API availability, and downstream application development all influence total lifetime requests. Third, for open-source models, the training organization has no visibility into total global usage. If Llama 3 is deployed by thousands of organizations independently, no single entity can observe the aggregate denominator [33].

Table 2. Observed deployment timelines for selected production language models. Lifetimes are measured from public release to deprecation or successor release. Models marked ongoing have not yet been deprecated.

Model	Release	Deployment duration
GPT-3.5 Turbo	Mar 2023	∼16 months
Claude 2	Jul 2023	∼13 months
Llama 2	Jul 2023	∼14 months
GPT-4	Mar 2023	∼25 months [9]
Llama 3	Apr 2024	13+ months (ongoing)

None of these is a measurement gap. Even with perfect instrumentation, the denominator cannot be known at training time, because it depends on future events that have not yet occurred, so the uncertainty can be managed but not eliminated. This also sets the relative importance of each challenge by deployment context: at high adoption (billions of requests) the per-request training cost is small regardless of boundary choice, so lifetime forecasting error dominates; at low adoption (millions of requests) boundary definition becomes the dominant source of variation. The other cross-challenge interactions are examined in Section 5.

4.2 A Starting Point

Given irreducible uncertainty, we recommend a hybrid approach with periodic reconciliation. This parallels how prospective LCA handles uncertain use-phase parameters: establish a baseline scenario, document assumptions, and update as data becomes available.

Initial estimate. At training time, establish a conservative lifetime estimate based on observed retirement patterns for comparable models. Current data (Table 2) suggests deployment durations of 13–25 months for frontier language models. Usage volume should be estimated from capacity planning projections, with the estimate documented alongside its assumptions and confidence level. Given the lifetime assumptions, the amortization rate is estimable but not deterministic. Since observed retirement diverges from initial assumptions in practice, the rate must be treated as provisional and revised, rather than reported as a fixed property of the model.

Periodic true-up. As the model operates, actual usage data accumulates and the forecast can be refined. The reconciliation frequency should match the organization's reporting cadence (quarterly or annually). Each update revises the denominator forecast based on observed adoption trends, producing an updated amortization rate applied prospectively to subsequent requests.

Final reconciliation. At model retirement, a final reconciliation using actual lifetime usage data produces the definitive per-request figure. Previously published provisional figures should be updated in retrospective disclosures. If lifetime was overestimated and the model is retired with training carbon still unallocated, the residual must be assigned rather than discarded: the final reconciliation redistributes the unallocated remainder across the realized request volume, which raises the definitive per-request figure. This prevents optimistic initial forecasts from causing systematic under-reporting.

Open-source and decentralized models. For models where usage is not directly observable, the hybrid approach must rely on proxy

estimation: download counts, API call sampling, or community surveys. These proxies provide order-of-magnitude bounds rather than precise measurement. Organizations should document the proxy methodology and acknowledge the resulting uncertainty explicitly, rather than reporting a point estimate without qualification.

5 Interactions and Implications

The three challenges we identify are not independent. Choices in one dimension propagate to others, and the resulting interactions determine the full range of variation in reported emissions.

The boundary–lifetime interaction is the most consequential. A wide boundary (one that adds R&D and embodied carbon to the final run) produces a large numerator; if the model also has low lifetime usage, the per-request figure is high on both accounts. Conversely, a narrow boundary combined with optimistic lifetime assumptions produces a low figure that may later require upward revision. The practical implication is that boundary and lifetime choices must be made jointly: an organization reporting at Tier 2 (with R&D multiplier) should pair it with a conservative lifetime estimate to avoid compounding optimistic assumptions.

The functional unit–lifetime interaction determines forecasting difficulty. Units with stronger compute correlation require more complex forecasts; an organization choosing per-device-second accounting commits itself to forecasting not just total request volume but also the complexity distribution over the model's lifetime.

These interactions have implications for standards development. Emerging frameworks such as the Green Software Foundation's SCI for AI specification [14] define a formula structure (emissions per functional unit per unit time) but leave boundary definition and lifetime estimation to the implementer. We now quantify how the interactions compound when all vary simultaneously.

5.1 The Arithmetic of Divergence

Consider BLOOM, the only frontier LLM with a fully decomposed emissions disclosure [23]. Under the narrowest common boundary (dynamic GPU energy for the final run), BLOOM emitted 24.7 tCO_2e. Adding idle and infrastructure power raises this to 39.3 t; including embodied hardware brings the total to 50.5 t; widening to the full development project (R&D, failed runs, evaluation) produces 123.8 t, of which the final run is 37%. One model, one grid, one year: the boundary alone moves the numerator fivefold. This multiplier is not a constant; it was approximately 2× for OLMo [28] and 25× for Moshi [21], scaling with architecture novelty.

The denominator compounds the divergence. Within a single model and task, per-request energy varies 6–25× depending on complexity and measurement protocol (Table 1). Across task types the spread exceeds three orders of magnitude: image generation draws roughly 1,000× more inference energy than text classification [22]. Deployment lifetime adds a further factor: observed durations range from 13 to 25 months across frontier models (Table 2), approximately 2×, and usage volume varies by orders of magnitude.

These factors enter the amortization formula multiplicatively. Boundary contributes ≈5×, functional unit 6–25× (same model, same task), and lifetime ≈2×. Their product places two methodologically defensible reports of the same model more than two orders of magnitude apart in the per-request training charge, without either report being incorrect. The divergence is not measurement noise; it is the arithmetic consequence of unstated choices.

Table 3. Training's share of per-request lifecycle emissions (f) as a function of daily request volume and emission boundary.

Volume (requests/day)	Tier 1 (narrow)	Tier 2 (wide)
10 million	94%	99%
100 million	63%	89%
2.5 billion	6%	25%
10 billion	2%	8%

5.2 The Share That Disappears

This compounding has a practical corollary: it determines how much of a model's lifecycle emissions vanish from reporting when training is omitted entirely. The excluded fraction follows directly from the amortization definition. Let T denote total training carbon (Section 2), N the total lifetime requests (volume × deployment duration, Section 4), and c_{op} the operational inference carbon per request (Section 3). The amortized training charge per request is $a = T/N$, and training's share of per-request lifecycle emissions is:

$$f = \frac{a}{a + c_{\text{op}}} = \frac{T}{T + N \cdot c_{\text{op}}} \tag{1}$$

Every input to Equation 1 depends on the challenges we decompose: T varies with boundary choice (≈5×), c_{op} varies with functional unit (6–25×), and N varies with lifetime and volume.

Table 3 evaluates Eq. 1 across plausible volume and boundary combinations for a frontier model with training energy of approximately 45,000 MWh [10] and per-request inference cost consistent with Section 3. As shown, the excluded share is therefore not a fixed percentage; it is itself a function of the same unstandardized choices.

6 Conclusion

Training carbon amortization is not one problem but three challenges that differ in kind: a boundary definition solvable through PCRs, a functional unit requiring context-dependent measurement design, and an irreducible lifetime forecast managed only through periodic reconciliation. Together they place honest reports of the same model more than two orders of magnitude apart, excluding 6% to 94% of lifecycle emissions depending on volume and boundary. The path forward requires coordinated action: standards bodies should develop a PCR for AI model training, as already underway for data center equipment [29]; cloud providers should expose per-workload Scope 3 data; and model trainers should publish disclosures with explicit boundaries, R&D multipliers, and lifetime assumptions. Inherited-emissions methodologies are emerging [1], but without comparable harmonization the same formula will keep producing incomparable figures. Consistent measurement with stated limitations is nonetheless preferable to inaction, and we invite the AI and LCA communities to develop such a rule with us.

Acknowledgments

We thank the anonymous HotCarbon reviewers, as well as Marco Masciola, Vinny Ferrero, Patrick Ford, and Zane Rankin for their valuable feedback and suggestions. We also note that Aravind Srinivasan's contribution to this publication was not part of his University of Maryland duties or responsibilities.

References

[1] Amazon Web Services. 2025. How AWS Estimates Embodied Emissions of IT Hardware: The Science and Technology Behind the Latest Customer Carbon Footprint Methodology. AWS Infrastructure and Sustainability Blog. https://aws.amazon.com/blogs/infrastructure-sustainability/how-aws-estimates-embodied-emissions-of-it-hardware-the-science-and-technology-behind-the-latest-customer-carbon-footprint-methodology/ Accessed: 2026-06-27.

[2] Amazon Web Services. 2026. AWS Sustainability Console: Carbon Footprint Reporting. https://aws.amazon.com/sustainability/tools/console/ Accessed: 2026-04-21.

[3] Brian Bartels, Ulrich Erber, Peter Sandborn, and Michael Pecht. 2012. Strategies to the Prediction, Mitigation and Management of Product Obsolescence. *John Wiley & Sons* (2012). doi:10.1002/9781118275474

[4] Noman Bashir, Varun Gohil, Mohammad Shahrad, David Irwin, Anagha B. Subramanya, Elsa Olivetti, and Christina Delimitrou. 2024. The Sunk Carbon Fallacy: Rethinking Carbon Footprint Metrics for Effective Carbon-Aware Scheduling. In *ACM Symposium on Cloud Computing (SoCC)*. ACM. doi:10.1145/3698038.3698542

[5] Noman Bashir, David Irwin, Prashant Shenoy, and Abel Souza. 2023. Sustainable Computing - Without the Hot Air. *SIGENERGY Energy Inform. Rev.* 3, 3, 47–52. doi:10.1145/3630614.3630623

[6] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, et al. 2021. On the Opportunities and Risks of Foundation Models. *arXiv preprint arXiv:2108.07258* (2021). https://arxiv.org/abs/2108.07258

[7] Semen A. Budennyy, Vladimir D. Lazarev, Nikita N. Zakharenko, et al. 2022. eco2AI: Carbon Emissions Tracking of Machine Learning Models as the First Step Towards Sustainable AI. *Doklady Mathematics* (2022). doi:10.1134/S1064562422060230

[8] Andrew A. Chien, Udit Gupta, Shaolei Ren, Akshitha Sriraman, and Bill Tomlinson. 2026. Strategies and Design for Increasing AI Sustainability. *Nature Reviews Clean Technology* (2026). doi:10.1038/s44359-026-00195-w

[9] Benj Edwards. 2025. The AI That Sparked Tech Panic Heads to Retirement: GPT-4 Effective End of Life. Ars Technica. https://arstechnica.com/ai/2025/04/the-ai-that-sparked-tech-panic-and-scared-world-leaders-heads-to-retirement/ Accessed: 2026-06-27.

[10] Epoch AI. 2025. Data on Notable AI Models. https://epoch.ai/data/notable-ai-models Training compute, power draw, and training time for notable models. Accessed: 2026-06-27.

[11] European Parliament and Council. 2022. Directive (EU) 2022/2464 – Corporate Sustainability Reporting Directive (CSRD). https://eur-lex.europa.eu/eli/dir/2022/2464 Official Journal of the European Union, L 322/15.

[12] Sophia Falk, David Ekchajzer, Thibault Pirson, Etienne Lees-Perasso, Augustin Wattiez, Lisa Biberfreudenberger, Sasha Luccioni, and Aimee van Wynsberghe. 2025. More than Carbon: Cradle-to-Grave Environmental Impacts of GenAI Training on the Nvidia A100 GPU. *arXiv preprint arXiv:2509.00093* (2025). https://arxiv.org/abs/2509.00093

[13] Google Cloud. 2025. Carbon Footprint. https://cloud.google.com/carbon-footprint Accessed: 2026-04-21.

[14] Green Software Foundation. 2025. Software Carbon Intensity for AI (SCI for AI) Specification. https://greensoftware.foundation/standards/sci-ai/ Accessed: 2026-04-21.

[15] Udit Gupta, Young Geun Kim, Sylvia Lee, Jordan Tse, Hsien-Hsin S. Lee, Gu-Yeon Wei, David Brooks, and Carole-Jean Wu. 2021. Chasing Carbon: The Elusive Environmental Footprint of Computing. In *IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 854–867. doi:10.1109/HPCA51647.2021.00076

[16] International Organization for Standardization. 2006. ISO 14025:2006 – Environmental Labels and Declarations – Type III Environmental Declarations – Principles and Procedures. Geneva, Switzerland.

[17] International Organization for Standardization. 2006. ISO 14040:2006 – Environmental Management – Life Cycle Assessment – Principles and Framework. Geneva, Switzerland.

[18] International Organization for Standardization. 2006. ISO 14044:2006 – Environmental Management – Life Cycle Assessment – Requirements and Guidelines. Geneva, Switzerland.

[19] Nidhal Jegham, Marwan Abdelatti, Lassad Elmoubarki, and Abdeltawab Hendawi. 2025. How Hungry is AI? Benchmarking Energy, Water, and Carbon Footprint of LLM Inference. *arXiv preprint arXiv:2505.09598* (2025). https://arxiv.org/abs/2505.09598

[20] Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. 2019. Quantifying the Carbon Emissions of Machine Learning. *arXiv preprint arXiv:1910.09700* (2019). https://arxiv.org/abs/1910.09700

[21] Marta López-Rauhut, Loic Landrieu, Mathieu Aubry, and Anne-Laure Ligozat. 2026. Environmental Footprint of GenAI Research: Insights from the Moshi Foundation Model. *arXiv preprint arXiv:2604.11154* (2026). https://arxiv.org/abs/2604.11154

[22] Alexandra Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power Hungry Processing: Watts Driving the Cost of AI Deployment?. In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*. doi:10.1145/3630106.3658542

[23] Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. 2023. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model. *Journal of Machine Learning Research* 24, 253 (2023), 1–15. http://jmlr.org/papers/v24/23-0069.html

[24] Eric Masanet, Arman Shehabi, Nuoa Lei, Sarah Smith, and Jonathan Koomey. 2020. Recalibrating Global Data Center Energy-Use Estimates. *Science* 367, 6481 (2020), 984–986. doi:10.1126/science.aba3758

[25] McKinsey & Company. 2023. The Economic Potential of Generative AI: The Next Productivity Frontier. https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier Accessed: 2026-06-27.

[26] Microsoft. 2025. Emissions Impact Dashboard for Azure. https://www.microsoft.com/en-us/sustainability/emissions-impact-dashboard Accessed: 2026-04-21.

[27] Mistral AI. 2025. Our Contribution to a Global Environmental Standard for AI. https://mistral.ai/news/our-contribution-to-a-global-environmental-standard-for-ai Accessed: 2026-06-27.

[28] Jacob Morrison, Clara Na, Jared Fernandez, Tim Dettmers, Emma Strubell, and Jesse Dodge. 2025. Holistically Evaluating the Environmental Impact of Creating Language Models. *arXiv preprint* (2025). https://arxiv.org/abs/2503.05804

[29] Open Compute Project and Fraunhofer IZM. 2025. Multi-Stakeholder Effort to Develop a Product Category Rule (PCR) for Data Center Equipment. Announced December 2025; moderated by Fraunhofer IZM under the Open Compute Project.

[30] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. *arXiv preprint arXiv:2104.10350* (2021). https://arxiv.org/abs/2104.10350

[31] Ian Schneider, Hui Xu, Stephan Benecke, David Patterson, Keguo Huang, Parthasarathy Ranganathan, and Cooper Elsworth. 2025. Life-cycle Emissions of AI Hardware: A Cradle-to-Grave Approach and Generational Trends. *arXiv preprint arXiv:2502.01671* (2025). https://arxiv.org/abs/2502.01671

[32] Scope3. 2025. Training Methodology – Modeling the Environmental Impact of AI Model Training. https://preview.methodology.scope3.com/training Accessed: 2026-04-21.

[33] Vector Institute. 2025. Sustainable Open-Source AI Requires Tracking the Cumulative Footprint of Derivatives. *arXiv preprint arXiv:2601.21632* (2025). https://arxiv.org/abs/2601.21632

[34] World Resources Institute and World Business Council for Sustainable Development. 2004. The Greenhouse Gas Protocol: A Corporate Accounting and Reporting Standard. https://ghgprotocol.org/corporate-standard

[35] Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, et al. 2022. Sustainable AI: Environmental Implications, Challenges and Opportunities. *Proceedings of Machine Learning and Systems* 4 (2022), 795–813. https://arxiv.org/abs/2111.00364