Towards a Methodology and Framework for AI Sustainability Metrics

Tamar Eilam, Pedro D. Bello-Maldonado, Bishwaranjan Bhattacharjee, Carlos Costa, Eun K. Lee, Asser Tantawi
IBM T. J. Watson Research Center, Yorktown Heights, USA

ABSTRACT

Recently, we are witnessing truly groundbreaking achievements using AI models, such as the much talked about generative large language models, the broader area of foundation models, and the wide range of applications with a tremendous potential to accelerate scientific discovery, and enhance productivity. AI models and their use are growing at a super-linear pace. Inference jobs are measured by the trillions, and model parameters by the billions. This scaling up comes with a tremendous environmental cost, associated with every aspect of models' life cycle: data preparation, pre-training, and post deployment re-training, inference, and, the embodied emission of the systems used to support the execution. There is an urgent need for the community to come together and conduct a meaningful conversation about the environmental cost of AI. To do that, we ought to develop an agreed upon set of metrics, methodology, and framework to quantify AI's sustainability cost in a holistic and complete fashion. In this paper, we propose unified AI Sustainability metrics that can help foster a sustainability mind-set and enable analysis, and strategy setting. To do that, we build on and extend the data center sustainability metrics, defined in [19], by introducing (for the first time to our knowledge) the concept of *embodied product emission (EPC)* to capture the creation cost of software assets, such as software platforms, models, and data-sets. We then use this new concept to expand the job sustainability cost metrics (*JCS* and *ASC*) offered in [19] to factor in the context of the execution of jobs, such as a platform as-a-service, or a model serving inference jobs. The result is applicable to any data center job, not just for AI, and contributes towards accuracy and completeness. We then show how to apply this approach to AI, with a particular focus on the entire life cycle, including all phases of the life cycle, as well as the provenance of models, where one model is used (distilled) to create another one. We demonstrate how the metric can be used to inform a more meaningful debate about AI strategies and cost. Te novelty of the approach is that it can be used to reason about strategies and trade-offs *across the life cycle* and 'supply-chain' of models.

CCS CONCEPTS

• **Information systems → Data centers**; • **Social and professional topics → Sustainability**; • **General and reference → Metrics**; **Measurement**.

KEYWORDS

green AI, foundation models, sustainable AI, sustainable computing

ACM Reference Format:
Tamar Eilam, Pedro D. Bello-Maldonado, Bishwaranjan Bhattacharjee, Carlos Costa, Eun K. Lee, Asser Tantawi . 2023. Towards a Methodology and Framework for AI Sustainability Metrics. In *2nd Workshop on Sustainable Computer Systems (HotCarbon '23), July 9, 2023, Boston, MA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3604930.3605715

1 BACKGROUND AND MOTIVATION

Artificial Intelligence (AI) is one of the fastest growing technology domains, involving academic research, businesses, and users. The enormous investment in AI led to groundbreaking applications in a diverse set of areas. AI is used for accelerating the discovery of drugs (e.g., [44], [39]), driving efficiencies at work (e.g., [37]), discovering new materials towards renewable storage (e.g., [52]), and more. At the same time, we are also in the midst of a heated debate about the potential of AI to harm (e.g., [7]). Risks frequently associated with AI include, fake news, biases, job losses, and, the enormous environmental cost.

The amount of compute used to train deep learning models have increased 300,000× in six years [42]. Data has increased significantly, reaching exabyte scale [49]. The data size increase has led to a super-linear growth trend in model size [49]. For example, GPT3 based language translation tasks have increased in size 1000× ([5]). In contrast, systems' memory capacity only grew moderately, which has motivated a variety of scale-out infrastructure solutions (e.g., [36, 49]), involving thousands of AI accelerators and other specialized systems. There is no doubt that these trends come with a dire environmental cost (stemming both from embodied and operational costs). Indeed, multiple researchers and practitioners have raised the alarm on the environmental cost of AI, and offered a calls-to-action ([23, 28, 42, 43]).

Meanwhile, a recent development in the field of AI is the concept of foundation models (FMs) coming to the front stage (e.g., [3]). FMs are trained on very broad datasets using self supervision at scale. One of the interesting characteristics of foundation models is that through transfer learning [45] they can be adapted (e.g., fine-tuned, or distilled) to a wide range of downstream tasks. In fact, the majority of state-of- the-art NLP models are now adapted from one of a few foundation models, such as BERT [14], RoBERTa [34], BART [32], T5 [38], BLOOM [48], LLaMA [46]. Foundation models are not new, but the scale, scope, and emergent capabilities of foundation models in the last few years have exceeded everyone's imagination. For example, GPT-3 has 175 billion parameters (in comparison with the 'modest' 1.5 billion parameters of GPT-2), and it can be adapted via natural language prompts to perform a range of tasks despite not being trained explicitly to do many of

HotCarbon '23, July 9, 2023, Boston, MA, USA

ACM ISBN 979-8-4007-0242-6/23/07.
https://doi.org/10.1145/3604930.3605715

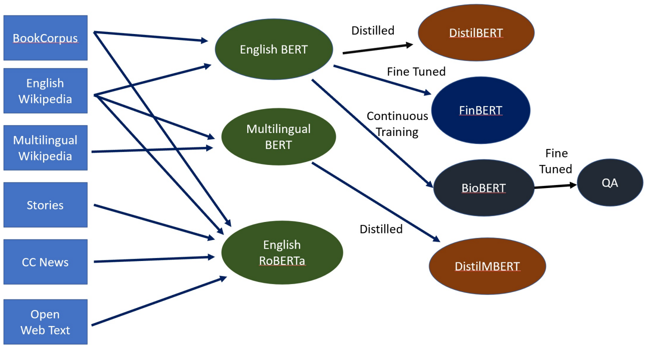

Figure 1: The 'Supply Chain' of datasets and models

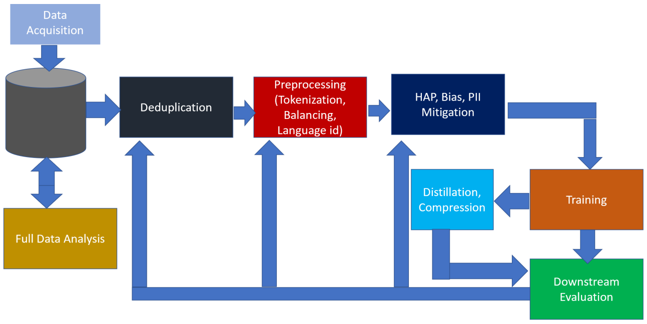
Data Acquisition
Deduplication
Preprocessing (Tokenization, Balancing, Language id)
HAP, Bias, PII Mitigation
Full Data Analysis
Distillation, Compression
Training
Downstream Evaluation

Figure 2: A typical preprocessing pipeline

those tasks [5]. Proponents of foundation models claim that the enormous amount of upfront carbon cost to train a broad model is balanced in the 'big picture' by the low cost of re-using it via fine-tuning or distillation for a particular task (e.g., [3]). Alas, with no unified metric that analyses the entire provenance chain of models, datasets, and their associated cost, it is very hard to substantiate or refute this claim.

In this position paper, we apply broad knowledge of sustainability principles, protocols and standards, including the GHG Protocol ([47]), and the accompanying guidance document for cloud services ([20], chapter 4), and product Life Cycle Assessment principles (LCA) [16], to the domain of AI, in order to develop metric that can be meaningfully used to drive sustainability mindset and examine trade-offs across the life cycle of AI models, and across the 'supply-chain' of models and datasets (see Fig. 1). As in [19], and also as in the Software Carbon Intensity (CSI) specification offered by the Green Software Foundation [21], we focus on a unit of work submitted on behalf of an end-user, i.e., inference jobs in the case of AI, as the primary object of interest. The metric is meant to capture the actual amortized sustainability cost of a job, taking into account operational overheads and associated embodied cost.

We leverage and expand the metrics defined in [19], concerning the sustainability cost of jobs. Specifically, we define and motivate a new term *Embodied Product Cost* for the sustainability cost of software, such as the development and testing of platforms delivered as an always running service (such as AWS's Lambda [29]), and, in the domain of AI, the preparing of datasets, and training of models. We then expand the definition of job sustainability cost metrics (*JSC* and *ASC*), to also factor in the associated embodied product cost. The expanded definition can contribute towards accuracy and completeness of job sustainability metrics. We then show how to apply this expanded definition to the area of AI. To summarize, the contribution of our paper is as follows:

(1) We define a new metric *Embodied Product Cost*, that aims at expressing the 'embodied' carbon of software assets, i.e., the carbon cost of 'manufacturing' a software asset, such as the development and testing of an on-line platform, or the pre-deployment training of an AI model.
(2) We expand the definitions of Job Sustainability Cost (*JSC*), and Amortized Sustainability cost (*ASC*), defined in [19] to factor-in the operational cost of the software asset used as the context of the execution of the job, as well as, the 'embodied' costs of these software assets. The expanded definitions can be used generally for any data center job, not just AI, and contribute towards accuracy and completeness.
(3) We specialize and apply these new metrics to the case of AI. We show that our approach can be used to investigate trade-offs across the life cycle, and in particular can be used to analytically prove or refute the claim that foundations model re-use for downstream tasks is advantageous to the environment, relative to the construction of smaller more specific models, from scratch.
(4) We analyse the new opportunities for sustainability based research across the life-cycle and provenance chain of models.

1.1 The life cycle of an AI model

To fully understand the real environmental impact, and to be able to develop the right approach to assess the impact, we must closely examine the model life cycle, end-to-end, including: data collection, model exploration and experimentation, model training, model distillation, and fine tuning, deployment, and then re-training, and inference cost.

A pre-processing pipeline, needed to curate data for training a model, could consist of the following steps: Data acquisition: via crawling (for NLP), or running simulations (materials or physics); De-duplicating: to ensure there is one copy of a document from multiple sources; Selecting documents of certain languages of interest; Splitting the documents into sentences for training; Identifying Hate Abuse Profanity and PII information one may want to filter; Forming the data files for training based on a given format. For example, NLP datasets such as Wikipedia, Stories, OpenWebText, BookCorpus, CC News are constructed via web crawling. Each of them could potentially be processed through all the above steps before it is ready for being used for training. Fig. 2 shows such a pipeline.

The datasets are used to train multiple models in various combinations. BERT, for example, was trained using English Wikipedia and BookCorpus. Whereas Multilingual BERT (mBERT) was trained on multilingual Wikipedia over 100+ languages [33]. RoBERTa on English datasets spanning Wikipedia, Stories, OpenWebText, BockCorpus as well as CC-News.

After a model is trained, the model may be distilled to bring it to a form factor which fits certain latency or space budget. For example, DistilBERT [41] and DistilMBERT are based on BERT and mBERT, respectively. After the base model is deployed, if often needs to be continuously trained to maintain accuracy. A model trained before 2019 would not have known about COVID. The frequency of re-training varies. For example, [49] reported re-training in weekly, or daily frequency for two different use cases.

1.2 Related Work

Sustainable computing has gained significant attention since a decade ago, owing to the escalating demand for computing power in Cloud computing and hyper-scale data centers, which has resulted in a significant surge in energy consumption at the data center and energy proportional computing [2]. To address this concern, there have been continuous efforts aimed at reducing power/energy consumption across different levels, ranging from chip/HW components [6, 25, 26] to data center scale [12, 27, 30, 50], including cooling techniques (Power Usage Effectiveness, PUE) and/or total cost ownership (TCO) [8, 35, 40, 51].

Multiple recent works take a holistic view of the problem area of sustainability in computing. Gupta et al. [22] posits that attention must be given to the embodied emission of systems, which is becoming the dominant factor in the carbon cost of computing environments. [9] argues that we need to replace static metric such as PUE and grid emission factors (CI) with dynamic, time series based, metric to enable de-carbonization of data centers by co-optimization. We agree with these assertions and believe they are complementary to the approach presented here. The most relevant to this paper is the work by Gandhi et al. [19] that argues that we should focus on the unit of work performed (namely, *a job*), and, include in its amortized sustainability cost also a 'slice' of the embodied emission cost of the systems used for the execution and other operational overheads such as for cooling. This approach agrees with the software Carbon Intensity (SCI) specification [18], put forward by the Green Software Foundations [21]. We agree that we should aim at amortizing all these aspects into the carbon sustainability cost of jobs. Toward that goal, we posit that [19] neglected to consider additional aspects associated with the *software product(s)* used as an execution context for the job. Examples of software products are a software platform, deployed as an always-running service, required for the execution of jobs (such as a serverless job running on the AWS Lambda platform service [29]), and, in the domain of AI, a model, that is used to serve inference requests initiated by end users. In both these cases there is a significant overhead associated with (1) the on-going maintenance of the product such as service operations, or, model continuous re-training, and, (2) the up-front construction of the product (including, development and testing, or, data preparation and training). We propose new metrics, that build on the previous ones, but include these two neglected factors.

In the area of AI, an astounding amount of progress has been accomplished on the systems side, to design energy efficient accelerators specifically for AI workloads, optimized for metrics multiplication, and incorporating low precision, and low voltage (e.g., [1, 4]). Beyond the lens of system design, data scientists spend most of their attention fiercely competing over accuracy as a primary goal, and time-to-value as a secondary goals (e.g., [34]). The papers [43] and [42], were among the first to call attention to the issue of the environmental cost of AI, focusing in particularly, on the training phase of the life cycle. Michel et al. [31], offers a single metric that can be used to compare the energy efficiency of different models with respect to their complexity (e.g., based on number of classes), and their accuracy. The work [36], asserts the difficulty in assessing the *gC*02*e* of AI models, which necessitate considering the energy mix in the location of the training, and the hardware architecture used. By considering these two aspect, they correct previous estimates, such as in [43] by as much as 88×. They also join the call to define standards and norms for Sustainable AI. Lastly, Wu et al. [49], is first to suggest we need a *holistic* approach to Sustainable AI that considers *all stages of the life cycle*, and also including the embodied emission of systems used. Some of the interesting findings in this paper are: (1) that the often completely overlooked data preparation phase, consumes in some cases an equal amount of energy as the training phase, and, (2) that the ratio between energy spent in different phases of the model life cycle *varies across different use cases.* These two key observations contributed to motivating this position paper. First, we focus on two specific software assets: datasets, and models. By defining the concept of 'embodied emission of software' we capture the gCO2e cost of pre-preparation of data set, and pre-training of models. Further we factor-in these costs, as well as the, mostly overlooked, cost of maintaining a model with frequent re-training, in amortizing the *gCO2e* cost per inference transaction, which is the unit of work performed on behalf on an end user. In contrast with other approaches that focus on a single phase only, e.g., training, (such as, [43] and [42]), or consider each phase in isolation (such as, [49]), we develop a metric that factors in the entire life cycle, including data preparation, training, and fine-tuning, to amortize the life cycle carbon cost of inference jobs. By using this metric, data scientists can evaluate different strategies, e.g., the amount of data to use, or number of model parameters, based on the expected number of inference jobs, with the objective of minimizing the eventual amortized job cost. They can compare life cycle strategies such as, using a smaller, but more specific model, or synthesizing multiple models, or tuning a very large model, with the objective of reducing the eventual amortized carbon cost of jobs. End users of models, can choose to use models with smaller amortized job cost, based on a single number that reflects the carbon cost of each transaction.

2 TOWARDS METRICS AND METHODOLOGY FOR SUSTAINABLE AI

As we stated above, our approach includes defining a new concept, which we term *Embodied Product Cost* (Section 2.2), then using it to expend the metric definition from [19] (Section 2.2), and lastly, applying the result to the domain of AI (Section 2.3).

2.1 Overview of Data Center Sustainability Metric

In this section, we give a brief overview of two of the relevant metrics defined in [19]. The reader is referred to [19] for a complete description. All metrics defined, employ the unit of "carbon dioxide equivalent" or gCO2e [1].

The **Job Sustainability Cost (JSC)** is aimed at quantifying the carbon footprint associated with running a job. A *job* is any unit of work, that an application performs relevant for scaling (i.e., work jobs require more resources). The Job Sustainability Cost (JSC) is calculated as the sum of energy used by the job's share of all systems participating in executing the job, and also including a 'tax' for the cooling and power loss overheads. Energy is converted to carbon based on the mix of energy sources and their associated carbon. For example, consider a job j that consumes 1 kJ energy executing on a host, and an additional 0.08 kJ due to cooling and power distribution losses. If the energy source mix is 80% coal and 20% solar, then, using carbon-intensity values (and converting kJ to kWh), we have $JSC(j) = 1.08 \times (0.8 \times 820 + 0.2 \times 48) \div 3600 \approx 0.2\,\text{gCO2e}$.

The **Amortized Sustainability Costs (ASC)** is aimed at further including an additional 'tax' for the life cycle cost of the systems used in the computation. It is calculated by adding the job's share (or tax) of the systems' embodied emission and other costs of all systems participating in the computation to the previously defined JSC. For the job j from the example above, assume that it runs exclusively for 5 minutes on a system S with an expected lifetime of 3 years, and the embodied cost of S is 10,000 gCO2e. Let $ec(j)$ be the 'tax' for embodied carbon. Then, $ec(j) = 10{,}000 \times \frac{5}{3 \times 365 \times 24 \times 60} = 0.031$, and $ASC(j) = JSC(j) + ec(j) \approx 0.231\,\text{gCO2e}$.

The paper defines other useful metrics, such as **Job Quality per Cost Rate (JQCR)** which we will not get into here due to space limits.

2.2 Proposed Extension

We agree with [19] that an end-to-end data center sustainability metrics must ultimately focus on the cost of a unit of work executed, and that all overheads to the extent possible, must be added in. However, one overhead which is not accounted for in the definitions in Section 2.1, is that of a job *execution context*. Most jobs today, in cloud or on-premise environments, execute in a context of a continuously running shared software platform. For example, a Serverless job on AWS executes in the context of a platform termed Lambda [29]. A container in IBM Cloud, is executed in a context of a platform termed IBM Kubernetes Service (IKS) [24]. There are significant overheads associated with the operations and life cycle of these platforms. For example, in IKS, there are a number of servers dedicated to managing the service in every location where the service is deployed. These management servers are always running, independently of the number of jobs executing. They are used to run management software to, e.g., monitor the health of the systems, and to meter usage for billing purposes. In storage and data services, such as IBM's Cloud Object Storage (COS) [11], processes wake up periodically, unprompted by any user initiated action, to perform cleanup, and tier management. Other shared services may be used across services (i.e., logging service), and the proportionate share of their use must be taken into account as well (see, [15]). In addition, we ought to not forget about the cost of continuous deployment (CD), and testing of the platform.

We use the term *software product* (in short, *product*) to denote any software asset, such as, a software platform that is used as-a-services to support the execution of jobs, or, an AI model used in serving inference requests, or, a datasets which are prepared and then used for the training of AI models, etc. Each software product is associated with a life cycle, including development, deployment, and use.

Next, we define a new metric to capture the 'embodied' carbon of software products, according to the same principles that are used for calculating the embodied carbon of systems. For systems, activities include material extraction, transportation, manufacturing processes, etc, factoring-in the *em entire supply chain* leading to the end product. For software, activities may vary based on the type of software product. For a platform delivered as a service they will include development and testing; for a dataset it includes data preparation and de-duplication; and, for an AI model it includes training. Each such activity consists of humans working on systems that consume energy from certain power grid, with a particular energy mix.

The **Embodied Product Cost (EPC)** is the upfront development cost of any given software product (up to the point of its deployment and use). It is calculated as the carbon footprint associated with all activities needed to create the product. We can for example refer to an activity of a developer, or, a tester, or a data scientist, a day, as a job j. The Embodied Product Cost EPC, is the summation over all $ASC(j)$ of all the jobs over the course of the creation of the product (a period that can easily take a couple of years).

Next, we propose to expand the definition of JSC to factor-in the platform's operational cost. To avoid confusion, we denote the expanded definition as JSC^e.

The **Expanded Job Sustainability Cost** (denoted JSC^e) is calculated as the sum of energy (converted to carbon) of all systems participating in the computation, and a 'tax' for cooling and power loses, and a 'tax' for the platform overhead if a platform is used as the execution context. For example, assume that the job j is a container that runs in the context of a platform such as IBM's IKS [24]. This platform, in a particular location such as Dallas, uses 3 servers dedicated to management, and their associated carbon is 50 gCO2e for every minute of operation. Let's assume that the job j executes for 5 minutes, and that while it is executing, there are concurrently 9 other jobs, of roughly equal size, executing on IKS in Dallas. Then, our job is 'taxed' $\frac{50}{10} \times 5$ for its share of the platform overhead. This number is added to $JSC(j)$ to derive $JSC^e(j)$. In addition, we also need to add the cost of service maintenance, and continuous development and testing. We leave this as an exercise to the reader.

Lastly, we expand accordingly the definition of the Amortized Sustainability Cost (ASC) of jobs. We claim that in addition to the embodied emission of systems participating in the execution of a job, we also have to add the embodied emission of any software product that serves in the execution.

The **Expanded Amortized Sustainability Cost**, denoted, ASC^e, is calculated as the sum of JSC^e, and, the job's share (or tax) of the systems' embodied emission (for all systems participating in the computation), and, the Embodied Product Cost (EPC) of the platform supporting the computation. As an example, if j is a container

[1] gCO2e stands for CO2 equivalent emissions, accounting for carbon dioxide and all the other greenhouse gases, such as methane and nitrous oxide

running on IKS, in Dallas, and it runs for 5 days, concurrently with 9 other equally sized jobs, and lets us assume that the expected life time of the IKS service in Dallas is 6 years, then it is taxed $EPC(IKS) \times \frac{5}{10\times365\times6}$, which is added to $ASC(j)$ to derive $ASC^e(j)$.

With the introduction of a new metric EPC to capture the embodied cost of software products, and with the expanded definition of JCS and ASC, we are now ready to turn our attention to the unique and fascinating life of AI models. We show how we apply these three metrics to meaningfully calculate the sustainability cost of AI inference jobs.

2.3 Metrics for Sustainable AI

We are now ready to apply the concepts and definitions from Section 2.2 towards metrics for Sustainable AI.

There are two 'products' that are of key relevance for AI. The first is a dataset. Refer to Section 1.1 for activities used to create a dataset. Once a dataset is ready, it can be used to train multiple different models. For a dataset d, we calculate the Embodied Product Cost $EPC(d)$ as the sum of carbon footprint of all activities involved in preparing the dataset. If we refer to each such activity as a single job then $EPC(d) = \sum_j ASC^e(j)$.

The second 'product' that is relevant to AI, is an AI model. A model is prepared (i.e., developed, or 'manufactured') via a process of experimentation, and training (see Section 1.1) leveraging one or more datasets. A given model can also be used to develop another, sometimes task-specific, model, in a process called *distillation* or *fine-tuning*.

For a model m, $EPC(m)$ is calculated as the sum of the carbon footprint associated with the development ('manufacturing') of a model, recursively. Formally, $EPC(m) = ASC^e(j) + w_1 \times EPC(m') + w_2 \times \sum EPC(d_i)$, where, j is defined as the 'job' of preparing m based on either another model m' or multiple datasets $d_1, d_2, \ldots$ and, w_1 is a tax weight to factor-in the re-use of a different model m'. It is 0 if there was no model that was re-used, or proportioned, based on its share of model re-use, and finally, w_2 is the tax weight, if the model was created based on datasets $d_1, d_2, \ldots$. It is 0 if there was no dataset that was used, or proportioned, based on its share of re-use. As an example, consider the RoBERTa model ([34]). It was pre-trained based on a set of datasets (Wikipedia, CC-NEWS, Stories, OpenWebText, BookCorpus), using 1024 32 GB NVIDIA V100 GPUs for approximately one day. Assuming that the GPUs were working at full capacity, the maximum power consumption is 300 J/s. Thus, the energy consumed for pre-training is $1024 \times 300 \times 360 \times 24 = 737$ kwh. We still have to add cooling/power-loss overheads, as well as, the 'tax' for the embodied system carbon footprint of the GPUs, as well as the fraction of embodied product carbon of the datasets used $EPC(\text{CC} - \text{NEWS} + \text{BookCorpus} + \text{Wikipedia} + \text{OpenWebText} + \text{Stories})$ (and then convert to carbon based on the grid energy mix). Yet as another example, DistilBERT ([41]), was created based on BERT via distillation. It has about half the total number of parameters of BERT base and retains 95% of BERT's performances on the language understanding benchmark GLUE. To create DistilBERT based on BERT, the team ([41]) used eight 16 GB V100 GPUs for approximately three and a half days. Again assuming GPUs were used to their max capacity (250 J/s) the energy for distillation turns out to be 14.4 kWh, and we need to add a tax for the fraction of re-use of BERT based on its EPC, and the other components as explained above.

Once a model is 'ready' it is deployed, and used to serve inference jobs. We can say that a model is *deployed* when it is available to be used to serve inference jobs. We can refer to the life cycle phase when it is serving inference jobs as *the operational phase.*

In addition to serving inference jobs, the model must be kept accurate, thus, it is being continuously re-trained. The frequency of re-training, and the cost of it, varies across models, and use cases. For example, in [49], the frequency reported for two different use-cases is daily, and weekly.

Let's assume that at time interval t, a model m was used to serve n inference jobs, and the carbon cost of re-training at that interval was cf_{rt} (note, there may have been multiple re-training 'jobs' at time interval T, in which case we take their sum). Then, the carbon cost of re-training cf_{rt}, must be split equally (or in proportion to the size, for non-equal jobs), across all jobs executing at time interval t, i.e., cf_{rt}/n is added.

As an example, inference jobs based on DistilBERT take ≈ 410 seconds to complete on an Intel Xeon E5-2690 v3 Haswell 2.9 GHz CPU, which translate to ≈ 0.015 kWh of energy. This is about 60% of the cost of inference with the original BERT. This demonstrates the benefits of tolerating the additional upfront cost associated with distillation, for downstream efficiency gains. To calculate JCA^e we need to also include the cost of re-training which is use case specific. Adding in the cost of model re-training will encourage data scientists to practice sustainability mindset, examining, for example, the needed frequency.

Finally, lets now discuss how we calculate $ASC^e(j)$, where j is an inference job performed against a model m. Here, we leverage our Embodied Product Cost metric ($EPC(m)$), in order to account for the carbon associated with the construction of a model, as well as the embodied system cost. Thus, $ASC^e(j)$ is calculated adding in both. For a model m, if the lifetime expectancy of the model is LT, for example, 3 years, and the execution time of an inference job is t, and there are n parallel jobs executing at any given time, then $ASC^e(j) = ASC(j) + \frac{EPC(m)\times t}{LT\times n}$, where $ASC(j)$ is defined as expected, adding to $JSC^e(j)$ the embodied system emission tax, and related costs.

Again, this metric exemplifies the usefulness for fostering a sustainability mindset. A datasetentist will be encouraged to ask questions, such as 'what is the right allocation of energy to train a model based on the expected usage, and expected life time'.

2.4 Sustainability measurements for AI

For a sustainability metric to be applicable, each term of the equation needs to be measured as accurately as possible. The EPC of the AI life-cycle comes from the infrastructure used to train, fine-tune, and deploy the model of interest. Primarly nowadays, this infrastructure is comisioned by cloud computing resources like AWS, Google Cloud, Azure, or IBM Cloud, to name a few. On-prem resources are also widely used thus expanding the heteroginity of options available to ML developers when it comes to choosing an AI infrastructure. For this reason, it is critical to have in hand software tools that make the measuring of energy and carboon footprint easily accessible and readily available. For cloud computing and applications running on containerized environments, Kepler [17] exposes hardware metrics and power measurements of containers running on Kubernetes or OpenShift. These metrics are exposed through Prometheus and allow users to write queries to aggregate energy data of different components. For workloads running on NVIDIA GPUs it is also possible to use DCGM [10] to collect power measurents that can be aggregated to compute energy data. Similarly, power data can be queried directly using `nvidia-smi` which comes with a driver installation for NVIDIA devices. Together, Kepler and DCGM can be used to aggregate power data of the harware components running the AI model, and can be paired with in house tools so organizations can measure their energy consumption for the infrastructure they use in AI, or other applications of their interest.

3 OPPORTUNITIES

A fundamental requirement would be to establish a reliable system to collect, store and report data concerning the power/energy utilization across the complete lifecycle of the AI model. Such a challenge poses significant difficulties as certain facilities may lack the capacity for monitoring or measuring. Furthermore, even in facilities equipped with monitoring capabilities, the methodologies employed for measuring power/energy could vary widely, such as how to allocate shared resources for model training. Consequently, this issue can be bifurcated into two principal aspects: standardization and reliable information tracking.

First and foremost, it is important to standardize the procedures employed for measuring, collecting, and reporting power/energy/carbon consumption data pertaining to the AI model. This measure would enable consistent and comparable reporting metrics. This paper takes a step towards that goal, more work is needed on the system energy consumption, in particular in lieu of sharing. Additionally, it is important to ensure accuracy and integrity of the records. Models and data sets should be published with a data sheet that reports based on the metrics. Yet another area is use case based optimization. The first advantage of the proposed approach is mind-set shifts in the AI model design process. Historically, AI models have been developed with an emphasis on performance. However, upon considering our AI lifetime observability matrics, system designers would begin to prioritize both performance and energy efficiency in their AI model architectures, as evidenced by factors such as neural network depth, width, input resolution, and parameters [13]. This would facilitate the development of AI models that are not only high-performing but also energy-efficient, thereby reducing the need for unnecessary training practices. In particular, it would be necessary to develop methods to compare different life cycle strategies in the context of use cases, i.e., re-use (and what) vs. start from scratch.

Furthermore, this approach would allow for the identification of inefficiencies within the various phases of AI, which could be targeted for optimization. Through the implementation of scheduling techniques and power knob configurations (such as power capping and dynamic voltage and frequency scaling), we would be able to enhance the utilization of each resource while simultaneously reducing energy consumption.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a new metric to evaluate the efficiency of AI models that associates with inference jobs the amortized life cycle carbon cost, factoring in data preparation, pre-training, fine-tuning, distillation, and on-going training. While other proposed evaluation techniques and metrics are useful for particular purposes, we believe this metric is necessary in order to reason about life-cycle strategies, such as, whether to train a smaller but more specific model, or, alternatively work to fine-tune a broad model for the target task, or, to decide on the optimal frequency of re-training, the amount of distillation, the size of the data set, number of tokens, and so on. These are real problems that data scientist are struggling with in the field. Any decision can be very costly in terms of time and energy/carbon consumption, and may lead to sub-optimal results. Data scientists need a way to strategize about these key life cycle decisions, and for that they need an objective function to base on. One way to do that is through the lens of life cycle carbon cost. The point is, that these decisions cannot be done in isolation for each phase, because this will not reflect trade-offs such as spending more on one phase in order to reduce cost in a downstream phase. For example, deciding to 'invest' in costly NAS may result in significant downstream efficiency improvements. In the future, we plan to demonstrate the usefulness of the metric by using it to compare and evaluate different life cycle strategies for realistic use cases.

REFERENCES

[1] Ankur Agrawal, Jungwook Choi, Kailash Gopalakrishnan, Suyog Gupta, Ravi Nair, Jinwook Oh, Daniel A. Prener, Sunil Shukla, Vijayalakshmi Srinivasan, and Zehra Sura. 2016. Approximate computing: Challenges and opportunities. In *2016 IEEE International Conference on Rebooting Computing (ICRC)*. IEEE, 1–8.

[2] Luiz Andr Barroso, Jimmy Clidaras, and Urs Hlzle. 2013. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines* (2nd ed.). Morgan & Claypool Publishers.

[3] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, and Chris Donahue et al. 2022. On the Opportunities and Risks of Foundation Models. arXiv:2108.07258 [cs.LG]

[4] D.M. Brooks, P. Bose, S.E. Schuster, H. Jacobson, P.N. Kudva, A. Buyuktosunoglu, J. Wellman, V. Zyuban, M. Gupta, and P.W. Cook. 2000. Power-aware microarchitecture: design and modeling challenges for next-generation microprocessors. *IEEE Micro* 20, 6 (2000), 26–44.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[6] Maurizio Capra, Beatrice Bussolino, Alberto Marchisio, Muhammad Shafique, Guido Masera, and Maurizio Martina. 2020. An updated survey of efficient hardware architectures for accelerating deep convolutional neural networks. *Future Internet* 12, 7 (2020), 113.

[7] Davide Castelvecchi. 2019. AI pioneer: 'The dangers of abuse are very real'. https://doi.org/10.1038/d41586-019-00505-2

[8] Aranya Chauhan and Satish G. Kandlikar. 2019. Characterization of a dual taper thermosiphon loop for CPU cooling in data centers. *Applied Thermal Engineering* 146 (2019), 450–458.

[9] Andrew A. Chien, Chaojie Zhang, and Liuzixuan Lin. 2020. Beyond PUE: Flexible Datacenters Empowering the Cloud to Decarbonize. *USENIX Hot Carbon* (2020). https://par.nsf.gov/biblio/10400420

[10] NVIDIA Corporation. 2023. NVIDIA Data Center GPU Manager (DCGM). https://developer.nvidia.com/dcgm

[11] COS. 2023. IBM Cloud Object Storage. https://www.ibm.com/cloud/object-storage

[12] Miyuru Dayarathna, Yonggang Wen, and Rui Fan. 2016. Data Center Energy Consumption Modeling: A Survey. *IEEE Communications Surveys & Tutorials* 18, 1 (2016), 732–794.

[13] Radosvet Desislavov, Fernando Martínez-Plumed, and José Hernández-Orallo. 2023. Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustainable Computing: Informatics and Systems* 38 (2023), 100857.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019). arXiv:1810.04805 [cs.CL]

[15] Tamar Eilam. 2021. Towards Transparent and Trustworthy Cloud Carbon Accounting. In *Proceedings of the 22nd International Middleware Conference: Extended Abstracts*. Association for Computing Machinery, 1–5.

[16] Environmental Protection Agency (EPA). 2023. Life Cycle Assessment (LCA). http://www.epa.gov/nrmrl/std/lca/lca.html

[17] Cloud Native Computing Foundation. 2023. Kubernetes Efficient Power Level Exporter (Kepler). https://sustainable-computing.io/

[18] Green Software Foundation. 2023. Software Carbon Intensity (SCI) specification project. https://greensoftware.foundation/articles/software-carbon-intensity-sci-specification-project

[19] Anshul Gandhi, Kanad Ghose, Kartik Gopalan, Syed Rafiul Hussain, Dongyoon Lee, David Liu, Zhenhua Liu, Patrick McDaniel, Shuai Mu, and Erez Zadok. 2022. Metrics for Sustainability in Data Centers. In *Proceedings of the 1st Workshop on Sustainable Computer Systems Design and Implementation (HotCarbon'22)*.

[20] GeSI. 2023. ICT Sector Guidance Built on the GHG Protocol Product Life Cycle Accounting and Reporting Standard. https://www.gesi.org/research/ict-sector-guidance-built-on-the-ghg-protocol-product-life-cycle-accounting-and-reporting-standard

[21] GSF. 2023. Green Software Foundation. https://greensoftware.foundation

[22] Udit Gupta, Young Geun Kim, Sylvia Lee, Jordan Tse, Hsien-Hsin S. Lee, Gu-Yeon Wei, David Brooks, and Carole-Jean Wu. 2020. Chasing Carbon: The Elusive Environmental Footprint of Computing. arXiv:2011.02839 [cs.AR]

[23] Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2020. Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning. *Journal of Machine Learning Research* 21, 1 (January 2020).

[24] IKS. 2023. IBM Cloud Kubernetes Service. https://www.ibm.com/cloud/kubernetes-service

[25] Qingye Jiang, Young Choon Lee, and Albert Y. Zomaya. 2020. The power of ARM64 in public clouds. In *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE, 459–468.

[26] Norman P. Jouppi, George Kurian, Sheng Li, Peter Ma, Rahul Nagarajan, Lifeng Nai, Nishant Patil, Suvinay Subramanian, Andy Swing, Brian Towles, Cliff Young, Xiang Zhou, Zongwei Zhou, and David Patterson. 2023. TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings. (2023). arXiv:2304.01433 [cs.AR]

[27] S. Kanagasubaraja, M. Hema, K. Valarmathi, Naveen Kumar, Bala Pradeep M Kumar, and N. Balaji. 2022. Energy Optimization Algorithm to Reduce Power Consumption in Cloud Data Center. In *2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*. 1–8.

[28] Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. 2019. Quantifying the Carbon Emissions of Machine Learning. arXiv:1910.09700 [cs.CY]

[29] Lambda. 2023. AWS Lambda. https://aws.amazon.com/lambda/

[30] Eun Kyung Lee, Hariharasudhan Viswanathan, and Dario Pompili. 2017. Proactive Thermal-Aware Resource Management in Virtualized HPC Cloud Datacenters. *IEEE Transactions on Cloud Computing* 5, 2 (2017), 234–248.

[31] N. Lenherr, R. Pawlitzek, and B. Michel. 2021. New universal sustainability metrics to assess edge intelligence. https://doi.org/10.1016/j.suscom.2021.100580

[32] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. (2019). arXiv:1910.13461 [cs.CL]

[33] Jindrich Libovický, Rudolf Rosa, and Alexander Fraser. 2019. How Language-Neutral is Multilingual BERT? *CoRR* abs/1911.03310 (2019). arXiv:1911.03310

[34] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. (2019). arXiv:1907.11692 [cs.CL]

[35] Dibyendu Mukherjee, Srabanti Chakraborty, Indranil Sarkar, Ahona Ghosh, and Sandip Roy. 2020. A detailed study on data centre energy efficiency and efficient cooling techniques. *International Journal of Advanced Trends in Computer Science and Engineering* 9, 5 (2020).

[36] 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:2104.10350 [cs.LG]

[37] Ruchir Puri, David S. Kung, Geert Janssen, Wei Zhang, Giacomo Domeniconi, Vladimir Zolotov, Julian Dolby, Jie Chen, Mihir Choudhury, Lindsey Decker, Veronika Thost, Luca Buratti, Saurabh Pujar, Shyam Ramji, Ulrich Finkler, Susan Malaika, and Frederick Reiss. 2021. CodeNet: A Large-Scale AI for Code Dataset for Learning a Diversity of Coding Tasks. arXiv:2105.12655 [cs.SE]

[38] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. (2020). arXiv:1910.10683 [cs.LG]

[39] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. 2022. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence* 4, 12 (2022), 1256–1264.

[40] G. Rostirolla, L. Grange, T. Minh-Thuyen, P. Stolf, J.M. Pierson, G. Da Costa, G. Baudic, M. Haddad, A. Kassab, J.M. Nicod, L. Philippe, V. Rehn-Sonigo, R. Roche, B. Celik, S. Caux, and J. Lecuivre. 2022. A survey of challenges and solutions for the integration of renewable energy in datacenters. *Renewable and Sustainable Energy Reviews* 155 (2022), 111787.

[41] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR* abs/1910.01108 (2019). arXiv:1910.01108

[42] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. Green AI. arXiv:1907.10597 [cs.CY]

[43] Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and Policy Considerations for Deep Learning in NLP. *CoRR* abs/1906.02243 (2019). arXiv:1906.02243

[44] Hannes Stärk, Octavian-Eugen Ganea, Lagnajit Pattanaik, Regina Barzilay, and Tommi Jaakkola. 2022. EquiBind: Geometric Deep Learning for Drug Binding Structure Prediction. arXiv:2202.05146 [q-bio.BM]

[45] Sebastian Thrun. 1998. Lifelong Learning Algorithms. *Learning to learn* 8 (1998), 181–209.

[46] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971 [cs.CL]

[47] World Resources Institue & wbcsd. 2023. The GHG Protocol. https://ghgprotocol.org

[48] BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, and Alexander M. Rush et al. 2023. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. arXiv:2211.05100 [cs.CL]

[49] Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, Michael Gschwind, Anurag Gupta, Myle Ott, Anastasia Melnikov, Salvatore Candido, David Brooks, Geeta Chauhan, Benjamin Lee, Hsien-Hsin Lee, Bugra Akyildiz, Maximilian Balandat, Joe Spisak, Ravi Jain, Mike Rabbat, and Kim Hazelwood. 2022. Sustainable AI: Environmental Implications, Challenges and Opportunities. In *Proceedings of Machine Learning and Systems*, D. Marculescu, Y. Chi, and C. Wu (Eds.), Vol. 4. 795–813.

[50] Wenfeng Xia, Peng Zhao, Yonggang Wen, and Haiyong Xie. 2017. A Survey on Data Center Networking (DCN): Infrastructure and Operations. *IEEE Communications Surveys & Tutorials* 19, 1 (2017), 640–656.

[51] Qingxia Zhang, Zihao Meng, Xianwen Hong, Yuhao Zhan, Jia Liu, Jiabao Dong, Tian Bai, Junyu Niu, and M. Jamal Deen. 2021. A survey on data center cooling systems: Technology, power consumption modeling and control strategy optimization. *Journal of Systems Architecture* 119 (2021), 102253.

[52] C. Lawrence Zitnick, Lowik Chanussot, Abhishek Das, Siddharth Goyal, Javier Heras-Domingo, Caleb Ho, Weihua Hu, Thibaut Lavril, Aini Palizhati, Morgane Riviere, Muhammed Shuaibi, Anuroop Sriram, Kevin Tran, Brandon Wood, Junwoong Yoon, Devi Parikh, and Zachary Ulissi. 2020. An Introduction to Electrocatalyst Design using Machine Learning for Renewable Energy Storage. arXiv:2010.09435 [cond-mat.mtrl-sci]