Innovations in Systems and Software Engineering (2025) 21:517–531
https://doi.org/10.1007/s11334-023-00546-x

ORIGINAL PAPER

Timeshifting strategies for carbon-efficient long-running large language model training

Akshaya Jagannadharao[1] · Nicole Beckage[2] · Dawn Nafus[2] · Scott Chamberlin[1]

Received: 1 August 2023 / Accepted: 18 November 2023 / Published online: 19 December 2023

Abstract
Language models play a vital role in various natural language processing tasks, but their training can be computationally intensive and lead to significant carbon emissions. In this study, we explore the effectiveness of timeshifting strategies to mitigate the environmental impact of long-running large language models (LLMs). We develop a simulation tool that estimates carbon emissions for LLMs, enabling developers to make informed decisions prior to running their workloads. By leveraging historical carbon intensity data from WattTime, we investigate the potential benefits and limitations of timeshifting in different locations, considering diverse energy profiles. Our findings demonstrate that timeshifting can substantially reduce emissions, but it is highly dependent on the region's carbon intensity and energy mix. We present insights into the trade-offs between emissions reduction and workload runtime, acknowledging the need for further advancements in carbon-aware computing practices. Our research contributes to the growing field of sustainable computing and encourages developers to adopt environmentally conscious strategies in language model training.

Keywords Large language model (LLM) · Energy consumption · Carbon-awareness · Timeshifting

1 Introduction

With the rapid growth of Large Language Models (LLMs) following the release of GPT-3, their transformative impact on various natural language processing tasks is undeniable. However, their development has been plagued by a critical challenge: the extraordinary amount of hardware, energy resources (and associated CO_2 emissions to generate this energy), and time required for training. Despite their impressive performance, the training of LLMs comes at a substantial cost to our environment. The massive energy consumption directly contributes to a significant increase in CO_2-eq emissions, potentially exacerbating the ongoing climate crisis. Further, this increase of CO_2-eq emissions in relation to Natural Language Processing (NLP) models is expected to grow exponentially over time [1, 2]. One applicable technique, which hasn't yet been deployed at scale for large-scale machine learning (ML) training, is carbon-awareness. Carbon-awareness can be defined as modifying the behavior of your software to do more work (i.e., consume more energy) when higher amounts of renewable energy are accessible, thereby lessening the burden on the power grid. On the other hand, it also involves reducing energy consumption during periods of lower availability to renewable, which supports the grid in managing workloads more effectively, ensuring that tasks are scheduled in line with the grid's generation capacity [3]. The purpose of using WattTime's Marginal Operating Emissions rate (MOER) is not to do with load per se, but specifically managing the load in ways that avoid the use of 'peaker plants'—plants that are turned on only to deal with peak load—which in turn contributes to carbon intensity. As the world grapples with the urgent need to address carbon emissions, there is a unique opportunity to embed carbon-awareness in the development and deployment of large-scale

✉ Akshaya Jagannadharao
akshaya.jagannadharao@intel.com

Nicole Beckage
nicole.beckage@intel.com

Dawn Nafus
dawn.nafus@intel.com

Scott Chamberlin
scott.chamberlin@intel.com

[1] Software and Advanced Technology Group (SATG), Intel Corporation, Mission College Blvd, Santa Clara, CA 95054, USA

[2] Sociotechnical Systems, Intel Labs, Jones Farms, Hillsboro, OR 97124, USA

ML algorithms. Introducing carbon-awareness into the training and deployment of LLMs provides a path for sustainable innovation.

The primary objective of this paper is to explore the concept of carbon-awareness in LLMs and propose a novel simulation toolkit using temporal shifting. For this paper, we consider temporal shifting to be a tool that optimizes the training of the model to the current power grid, taking advantage of the dynamic nature of renewable energy resources.

By developing simulations to estimate the total training time of an LLM under different timeshifting assumptions, we demonstrate how timeshifting can significantly reduce the carbon footprint of LLMs. Leveraging the dynamic nature of the grid with the batchable workloads of LLM training, we are able to see a meaningful trade-off between emissions savings and time. Beyond our exploration of timeshifting, we also consider the variability across and within different power grids and simulate emission estimates for different workloads relative to *where* and *when* these workloads are run.

In the subsequent sections of this paper, we will lay the groundwork for developing simulations to explore the effects of timeshifting on training LLMs. Our methodology will involve constructing realistic training scenarios and evaluating the impact on model performance and how we can improve on existing techniques to build more carbon-awareness into existing applications. We build on generic timeshifting strategies to create a two-threshold timeshifting methodology. We believe this research expands the strategies available to researchers to decarbonize LLM research.

2 Background and previous work

There is a growing community of researchers focusing on the need for sustainability in deep neural network development. From this body of work, it has become clear that energy efficiency, particularly performance per watt, is not a complete view of the carbon cost of training algorithms. Optimizing for performance per watt neglects the complexity of the energy grid which can vary both in time and space. Figure 2 shows how the energy grid is dynamic. There is clear location, seasonal, and time variability. For example, an energy grid in California has diurnal variability: solar power is only available when the sun is shining. Further, overall grid usage can vary based on changing demands which can be difficult to estimate.

As more renewables come online, and as battery technology lags in its ability to stabilize the grid, this intermittency is only going to grow in the future. Being "carbon-aware," then, means being aware of, and resilient to, this intermittency. We focus on one solution to build resilience and adapt to grid intermittency: timeshifting model training.

There is also increased societal and regulatory pressure to improve carbon reporting in general, and for assessments of AI emissions specifically [4]. LLMs have become a reputational issue for the technology industry as a whole [5]. While there is a belief that ML models will plateau in size and then reduce [6], there is no shortage of innovation in modeling that considers bigger architectures and more parameters to be the easiest solution for more predictive and accurate models. If model efficiency is not inevitable, and efficiency efforts in the technology industry tend to lead to more emissions in the long term, not less, due to Jevon's paradox [7], decarbonization—and even just foregoing work that does not need to be done—must play a stronger role in developer decisions than it does currently.

Much of the work to improve the carbon footprint of models has been directly tied to improving efficiency and performance of models, relating improvement in efficiency to sustainability. Some of these techniques for improving efficiency directly try to reduce the number of model parameters through methods such as distillation, e.g., [8], sparsity, e.g., [9], reducing space and/or time complexity, e.g., [10] or even introducing lighter weight LM models for training data selection [11]. There is a body of work describing and categorizing these efficiency improvements (e.g., [12] for transformers or [13] for foundation models and their respective references) but efficiency is not the same thing as sustainability. Sustainability requires a broader view of the problem, beyond simply ensuring the code, architecture, and model parameter size is correctly scaled to the application of a model.

Other approaches to sustainability considerations in the development, training and deployment of models have been emerging across the literature. Some of these works include resources and models to estimate carbon emissions, e.g., [14–16], while others focus on and quantify the effect of optimizing specific aspects of the algorithms to lower carbon cost, e.g., [17] via grid and resource consideration. Other focus on the need for data and accurate reporting [18, 19], suggesting various tools such as leaderboards to incentivize the larger ML community [19] These efforts collectively capture a growing change in perspective in the community. We capitalize on this growing interest and perspective to focus on the usefulness and applicability of defining and exploring the use of algorithms that dynamically adapt to the current energy and grid environment. Specifically, we focus on timeshifting the workload to start, pause, resume and end with carbon cost in mind.

Previous work in the timeshifting domain has primarily focused on smaller language models with relatively short training times [20]. Dodge et al. demonstrated that timeshifting can lead to an 80% reduction in carbon emissions when applied to these models within a 6–24 h window [20]. How-

ever, LLMs present a unique challenge as their training can extend over several months. Meta AI, for example, estimates that it took 5 months to train their LLaMa (Large Language Model Meta AI) LLM model [21]. To address this challenge, we have developed a simulation tool that estimates emissions for larger models without the need to physically run the workload. The tool also incorporates the cost of system idle time to gain insights into the overall cost of timeshifting. By providing developers with this tool, we aim to empower them to make informed decisions in reducing carbon emissions before writing a line of code.

3 Challenges

Calculating carbon emissions for a piece of software, such as an LLM, presents a multifaceted and intricate task. Following the Software Carbon Intensity Specification detailed by the Green Software Foundation [22], we break down emissions into two components: operational carbon emissions and embodied carbon emissions. The sum of these components makes up the carbon emissions for a particular workload.

The complexity arises primarily due to several interrelated challenges:

1. Carbon Intensity: Obtaining precise carbon intensity values corresponding to different energy sources requires access to a comprehensive and up-to-date database. In our research, we utilize WattTime to retrieve carbon intensity values from the grid [23], similar to other work in timeshifting [20]. WattTime makes predictions of carbon intensity values up to 72 h into the future. Our timeshifting approach uses these predictions to schedule work in a way that reduces overall carbon emissions. However, these are only predictions and the actual associated emissions may differ widely from the predicted values.
   In addition, WattTime reports MOER carbon intensity values. MOER values represent the impact of adding an additional load to the grid. In comparison, average emissions represent the overall sustainability of the system. The flexibility of our simulation allows us to switch between MOER and average emissions. However, considering resource constraints, we have made a deliberate choice to concentrate our efforts on MOER values for our calculations. This decision is driven by the desire to comprehensively explore the effects of timeshifting on immediate carbon emissions and its alignment with grid dynamics.
   Carbon intensity information is notoriously inconsistent across sources, often due to scoping issues [20]. Even units suffer inconsistencies. For example, Dodge et al [20] also used WattTime, and indicated that WattTime uses carbon dioxide equivalent (CO_2e), which includes all greenhouse gasses converted to the equivalent amount of global warming potential of carbon dioxide. Yet WattTime's documentation suggests their data refers to CO_2 only. Data from the US Environmental Protection Agency [24] suggest that this might ultimately make little practical difference numerically, but it demonstrates the inconsistencies that are commonplace in this area.
   Hence, we refrain from attempting to make our estimates comparable across data sources by choosing only one source: WattTime. This ensures that there is a level of consistency as we do our calculations.
2. LLM-Specific Challenges: LLMs, with their immense computational demands and prolonged training times, exacerbate the complexities of measuring carbon emissions compared to conventional software applications. The distributed and parallel nature of LLM training further complicates the assessment of energy consumption and carbon impact. We address this issue by assuming the workload uses all available compute resources. However, this assumption may introduce some level of uncertainty.
3. Data center management: Similar to maintaining a power grid to always have electricity available, data centers are tasked with keeping computing resources at the ready for customers. However, this means that systems remain in an idle state until work is scheduled. Maintaining large amounts of compute resources, even in idle states, requires a large amount of power (and thus releases a large amount of carbon).
   Our simulation attempts to consider the cost of leaving the system idle by using the idle power of the device in our calculations; however, this is a simplified estimation of idle power and it does not take into account the dynamic nature of idle such as when the system switches between standby mode and background tasks.
   There is a growing need to understand how idle systems contribute to carbon cost and how scheduling when and how many servers are available at any given time impacts the overall carbon cost of LLM compute. Further, understanding different system states and their impact on carbon emissions may provide simple strategies to lowering the carbon emissions of systems at rest.

4 Power grid analysis

Understanding the intricacies of the power grid is essential for sufficiently accurate carbon calculations and effective workload planning. In this paper, we focus on a few specific grids, namely a relatively 'green' grid run by the California ISO North balancing authority (CAISO_NORTH), a grid with a large amount of variability run by SPP Western Nebraska (SPP_WESTNE), and a fairly 'dirty' grid run by WAPA Rocky Mountain Region (WACM). Our default grid for anal-

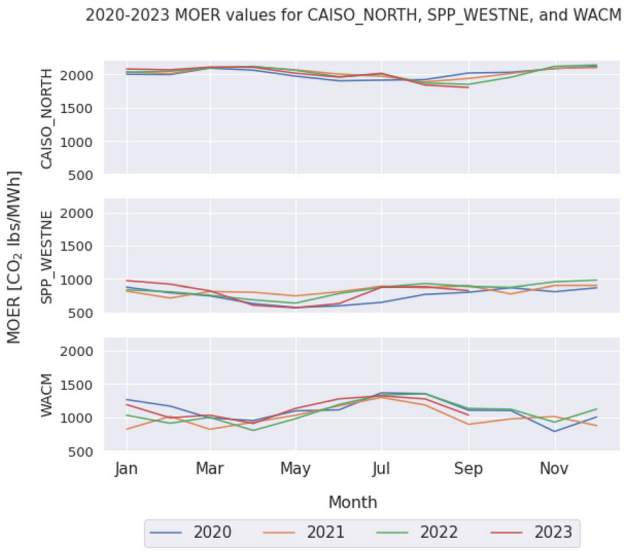

Fig. 1 Marginal Operating Emissions Rate (MOER) of three different grids across different years and months

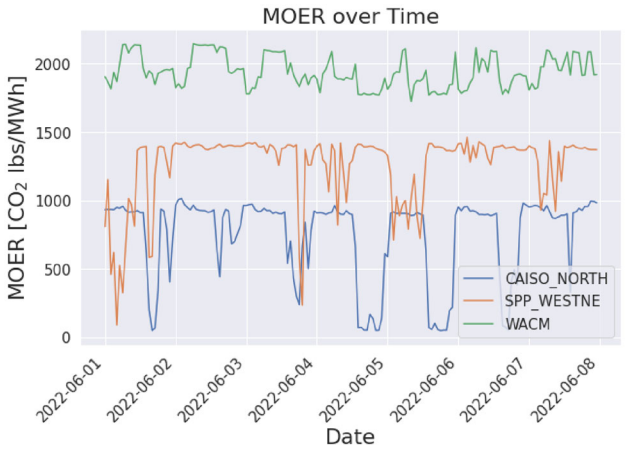

Fig. 2 Marginal Operating Emissions Rate (MOER) for our three grids during the week of June 1, 2022

yses is California ISO North,[1] but as can be seen in Fig. 1, there is a lot of variability in marginal emissions across different grids and across different times of the year.

Each grid also has different mixes of fossil fuels and renewable energy. While the MOER values take into account the carbon cost of the fuel source, it can be helpful to know more information about the grid. For example. California's energy mix exhibits a significant dependence on fossil fuels, constituting about 66% of its energy consumption in 2021 [25]. Conversely, renewable energy sources contribute only 33% to the state's energy supply, with solar contributing 14% and wind contributing 11%. The disparity between these figures underscores the importance of addressing the carbon footprint of software workloads, as they significantly contribute to greenhouse gas emissions in regions like California (Fig. 2).

Graphing the emissions reported by WattTime over the year for these three test locations, we see a similar global pattern where certain grids have more overall marginal emissions but also that there is variability in day-to-day on some grids more than others.

Insights gained from this power grid analysis are invaluable for effective workload planning and carbon-aware algorithm design. It is obvious that choosing a grid with overall lower marginal carbon emissions can greatly affect the overall carbon impact of the model. However, by understanding the temporal variations in marginal carbon intensity, we can go one step further and build training schedules and software execution that can capitalize on the temporal variability in the grid. We can, for example, align training with periods of lower carbon emissions, contributing to a greener and more sustainable software development approach.

5 Approach

5.1 System and energy estimation

Our simulation tool kit allows for quick and easy estimations of the overall carbon cost of a particular large-scale training procedure. The simulation breaks down the problem of estimating the carbon cost of training a large-scale model into three main components: the system, carbon emission estimation, and the time shift. We begin by defining the number of nodes and devices in the system. Using the idle and Thermal Design Power (TDP) characteristics of the hardware, we estimate the system's idle and max power consumption.

For this simulation, we assume that each node in the system is running at the same rate and either in idle or max state. The idle state for a node is the product of the CPU/GPU power usage at rest multiplied by the number of CPUs/GPUs used in each node. The node max is similarly defined as the TDP value associated with the specific hardware. Thus the system idle/max is just the number of nodes multiplied by the calculated node level idle or max power. We obtain TDP and idle power values from the manufacturer's product specifications.

In our system definition, we approximate a system with a generic CPU and GPU. Our estimation tool needs only an estimate of each component's Thermal Design Power (TDP) and idle power consumption. TDP estimation is publicly available from many different sources. Idle power, however, is more difficult to estimate, as this information is not readily publicly available and can vary significantly based on individual system configuration and hardware. We use TDP

[1] We choose California ISO North as our default grid as the MOER data is accessible to all via WattTime without a subscription.

estimates from CodeCarbon [26] and use ballpark estimations of CPU idle at 10 watts and GPU idle at 15 watts. We leave this discussion of power usage estimation for the future work section in favor of focusing on our calculations for estimating carbon costs.

For simplicity, we currently consider only these two system states. Still, future work could incorporate more granularity by using power curves distributed by manufacturers and better-characterizing hardware utilization as the workload runs. There is also the potential to add more devices like accelerators and memory to the system definition.

5.2 Carbon estimation

To calculate carbon emissions, we utilize historical data obtained from WattTime. WattTime data is available as part of an API where registered users can access location-specific marginal carbon emissions. The MOER is computed using the EPA CAMPD data [27] which provides hourly electricity generation and emissions of major fossil-fuel power plants in the United States. These values are derived by applying a regression model to the fossil-fuel emissions reported by power plants and are supplemented with additional data about renewable energy sources [28]. WattTime reports the carbon intensity of the grid as MOER values at 5-minute intervals. The location granularity of WattTime is dictated by the region's power balancing authority which is responsible for grid balancing. In total, WattTime has 100 unique power grid locations, covering the full United States and parts of Canada. While access to this API requires a subscription or data request from WattTime, individuals can replicate our work with free access from WattTime to historical data from California ISO North's MOER values over time.

While WattTime has some predictive models focused on estimating grid usage in the future, the time horizon for these predictions is only 72 h, meaning most of our workloads are beyond the prediction time frame. Thus we turn to historical data.

To align with our system-level assumptions in which power measurements are computed in hours, we downsample the data to derive hourly MOER values. As previously mentioned, MOER values represent the impact of an additional load on the grid. To ensure the accuracy and consistency of our data, and to account for potential changes in methodologies and energy sources over time, we have chosen to specifically utilize WattTime's model version 3.2 (referred to as MOER version 3.2 in the data) for our investigation. To calculate MOER for a specific workload, we take a subset of that dataset starting from when the workload is scheduled to begin and calculate the hypothesized end time. We then multiply the MOER values (sampled hourly) by our estimated system power to calculate the MOER cost for the entire runtime.

5.3 Replication study

To ensure the representativeness of our calculations for real-world workloads, we closely replicate the setup of Meta AI's long-running LLM model named LLaMa [21]. By replicating the overall carbon cost of training a long-running LLM, we validate our simulated power consumption of the underlying system on which long-running LLMs are trained. This allows us to rely on our simulated power consumption model *instead* of measuring the power consumption directly via training. By showing we have a reliable estimate of power consumption, we can then explore algorithms that adapt to the grid without having to train any LLMs directly.

LLaMa required 2048 GPUs and a training time of approximately 5 months, resulting in an estimated energy consumption by the original authors of around 2,638 MWh. No information was provided about CPU usage. The authors estimated the carbon emission of LLaMa at 1015 tons of COe_2. This is equivalent to 128 homes' energy for 1 year according to the United States Environmental Protection Agency [29]. They mention their utilization of the GPU's Thermal Design Power (TDP) of 400W to approximate the power consumption of the workload.

Assuming the same number of GPUs and runtime for the purposes of synthetic analysis, our estimation tool projected the system to consume about 2949.12 MWh of energy during training. The discrepancy is relatively modest at 12%. This suggests that while models, hardware, hardware utilization, system configuration will vary, our synthetic analysis can reasonably approximate a realistic LLM scenario. Estimating carbon emissions, especially retrospectively, can be very difficult. Our goal is not absolute precision, but rather accuracy sufficient to enable developers to make a ballpark estimate *prior to* training a large-scale production model. Making such an estimate requires very little emissions, and supports the ability to optimize training time for sustainability.

Carbon emissions depend heavily on the location where the workload runs. While LLaMa assumes the US national carbon intensity of 0.385 kg CO_2-eq/kWh, in this study, we specifically use carbon intensity data from California ISO North. This allows us to explore the benefits of timeshifting (below) while considering the region's actual carbon footprint and providing more accurate insights into the potential environmental impact.

If the energy source were restricted to California ISO North, we estimate that this model or a similar model would have released 1099 tons of CO_2. This is approximately an 8% difference, which might be accounted for by the location, the difference in emissions data sources, or a combination of both.

Whereas the approximation of the carbon emissions estimate of LLaMa from the original paper [21] uses the national

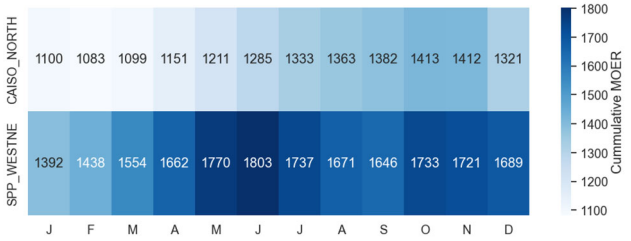

Fig. 3 Heatmap of marginal emissions of training LLaMa on different grids (y-axis) with different start times as months in 2022 (x-axis). We exclude WAPA Rocky Mountain Region (WACM) from this figure because the estimated carbon cost was between 2883 and 3039 tons, masking the seasonal variability of the other locations

average, we can easily test what the carbon emissions would look like at varying locations. Figure 3 shows our estimation of carbon emissions for training LLaMa on different grids and starting training at different times of the year. We choose to look at 3 different grids California ISO North (our original estimation), California ISO Palm Springs, and SSP Western Nebraska. We choose California ISO Palm Springs because it has historically been one of the greenest grids on average. SSP Western Nebraska is a grid with high variability both in the short term and across seasonality and thus offers insight into how a highly variable grid may impact and alter the amount of carbon emissions training a large-scale model.

We note that the similarity in estimating both the power usage as well as the overall carbon cost of LLaMa with such minimal information related to carbon training costs verifies the need, and use, for such tools as ours that allow individuals to do a back of envelop calculation to understand how training a model of this size would impact the environment. Additionally, the ease of adjusting which grid the model is trained on, and when training starts, empowers users to make more sustainable decisions with low overhead cost (see Fig. 3).

5.4 Introduction to timeshifting

Timeshifting is a largely effective strategy to reduce carbon emissions of software in data centers, including major players like Google [30]. Timeshifting refers to the strategic scheduling of computing tasks and processes to align with periods of lower energy demand or higher availability of renewable energy sources. For instance, data centers can analyze historical data and predictions on future renewable energy availability to dynamically shift resource-intensive workloads, such as data processing, computational tasks, and server operations, to periods when the energy grid relies more heavily on sustainable energy generation [30].

5.5 Two-threshold timeshifting

Simply modeling the system and linking it to location-specific MOER measurements allows us to approximate the overall carbon cost of training a model at various times of years and various locations. As can be seen, by Fig. 2, both seasonality and location play a large role in the overall carbon emissions cost. Noting the difference that can be made by choosing a different time of year to train a large-scale model, we incorporate a basic timeshifting algorithm into our simulation. The idea is to automatically place the system into idle when the grid is detected to be 'too dirty'. We can then decide when the grid is 'clean enough' to start training the model again.

To illustrate the usefulness of timeshifting and its impact on carbon, we investigate the impact on emissions by considering a cutoff to stop training and a separate cutoff to restart training. How these thresholds are chosen can quickly get complicated and be directly related to the various temporal changes a specific grid undergoes. While one could imagine optimizing these cutoffs for the grid and time in which the model will be trained, we instead define a very naive and simple cutoff procedure in order to isolate the effect of timeshifting specifically. We define these thresholds to be based on a percentile MOER usage averaged over the last few years for a specific grid location. This allows our carbon-aware algorithm to adapt to the specific location (allowing for cross-location comparisons) without making assumptions about future grid utilization. We then explore various cutoffs for stopping and separately resuming training. Such a timeshifting procedure is clearly not defined when the restart cutoff is less than the stop cutoff as the system would simply instantly restart. Figure 9 shows two cutoffs we considered in exploring the use of timeshifting.

Utilizing these thresholds, we explore various permutations by pausing and resuming the workload based on the specific MOER values at the identified percentiles. For instance, we can pause the workload when MOER values cross the 95th percentile (985.33 lbs CO_2/MWh for California ISO North's power grid) and resume the workload when the carbon intensity drops below the 75th percentile (949.0 lbs CO_2/MWh for California ISO North's power grid).

Of note, in our approach, we calculate the MOER thresholds over the course of several years. This methodology could introduce biases in the model due to systematic over/under estimation of normal grid use at the current time. This bias could be due to changes in the energy mix (e.g., introduction of more green energy sources resulting in lower MOER esti-

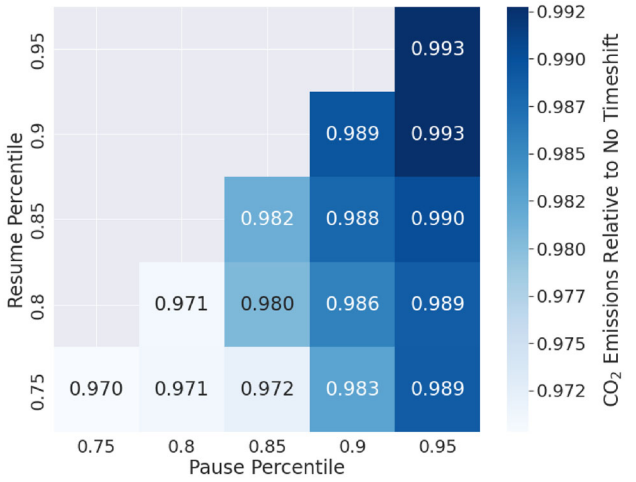

Fig. 4 Heatmap of timeshifted emission costs for LLaMa normalized by the non-timeshifted workload starting in January 2022. X-axis represents the carbon intensity percentile we pause running the workload; Y-axis represents the carbon intensity percentile we resume the workload. The emissions are represented by the color of the cells, where lighter cells indicate lower emissions. Percent carbon savings can be estimated as 1 minus the cell value

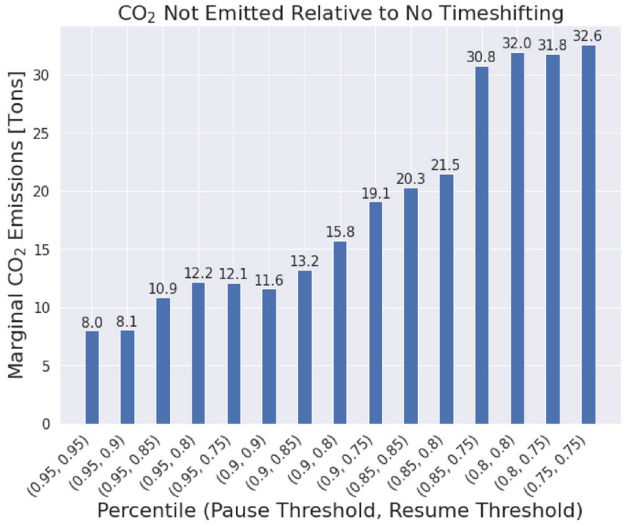

Fig. 5 Histogram of CO_2 savings for timeshifted LLaMa workload starting in January

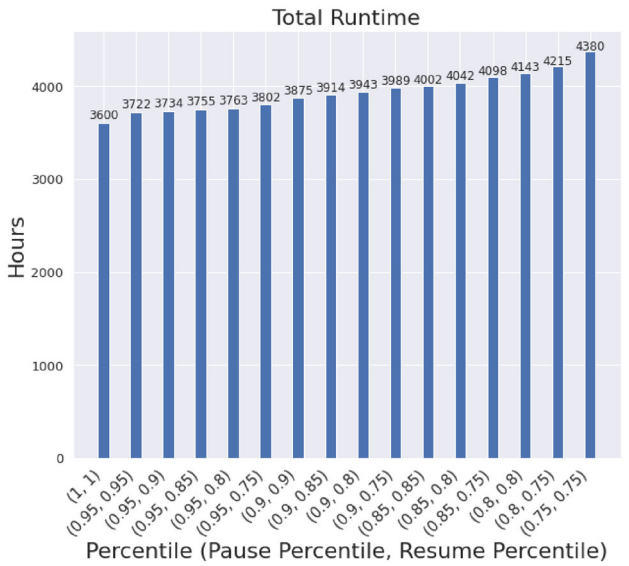

Fig. 6 Histogram of runtime costs for timeshifted LLaMa workload starting in January

mates over time) or due to a growing demand and utilization on the overall grid. Thus, if we are using our timeshifting model to minimize carbon cost, we can expect to see larger carbon savings in January than in July. However, we can also expect a longer runtime for the timeshifted January models than the July models. The longer runtime cuts into the potential carbon savings since the system cannot be made totally carbon neutral and must wait in idle until it can be resumed. Such biases arise due to the inclusion of emissions data from various seasons and periods with different energy usage patterns. To mitigate this effect, users of this tool should carefully consider how to define the stop-training and resume-training cutoffs. Because there are strong seasonal patterns within grids, there may be an advantage to considering seasonal adjustments along with realistic time expectations. Careful consideration and sensitivity analysis are necessary to account for the seasonal variations in emissions and ensure the robustness of our simulation results.

5.6 Further exploration: timeshifting

After employing our two-threshold approach, we present a heatmap of our findings (Fig. 4). For a 5-month-long workload initiated in January within California ISO North, our strategy yields minimal carbon savings. In fact, we can only prevent approximately 32 tons of CO_2 emissions (Fig. 5), while extending the runtime by 780 h (equivalent to 32.5 days) (Figs. 6, 7).

Upon rescheduling the workload to run one month later, we observe that the emissions worsen. While commencing the workload in January 2022 could have potentially prevented the emission of up to 32 tons of CO_2 (Fig. 8), shifting the start to February only offers a maximum reduction of 13 tons of CO_2. Consequently, timeshifting the workload introduces no significant benefits and, in turn, extends the workload's run time.

These results likely stem from initiating the workload during the grid's peak emissions period. However, this highlights the significance of our simulation tool. By having the capability to simulate the timing and execution of long-running workloads before their actual implementation, we can avoid potentially costly errors. Such foresight empowers us to optimize workload scheduling and minimize the environmental

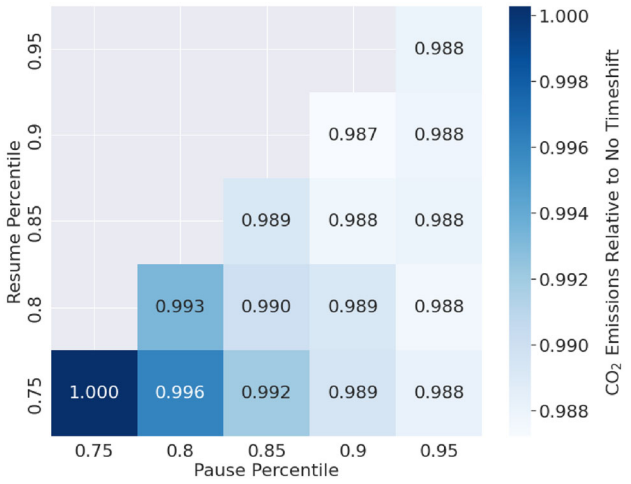

Fig. 7 Heatmap of timeshifted emission costs for LLaMa normalized by the non-timeshifted workload starting in February 2022

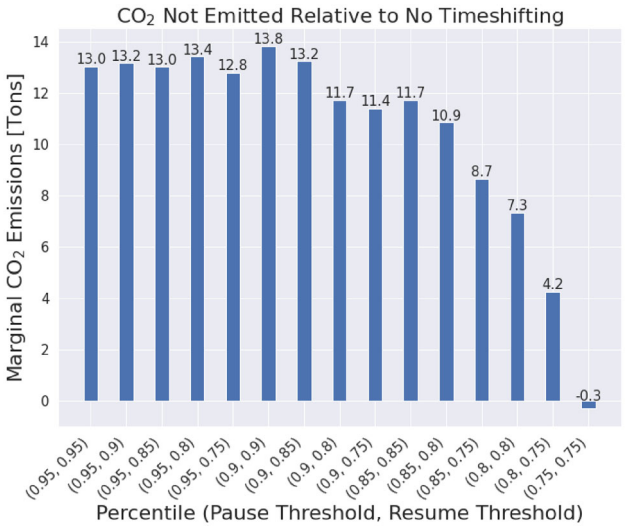

Fig. 8 Histogram of CO_2 savings for timeshifted LLaMa workload starting in February

impact, making the tool indispensable for responsible and efficient computing practices.

5.7 Exploring other workloads

In this section, we delve into the realm of other workloads, expanding the scope of our investigation beyond the previously examined scenarios. As we venture into diverse applications of timeshifting, we explore how this innovative approach can be adapted to various types of workloads, each presenting unique challenges and opportunities. By delving into these different workloads, we aim to unlock valuable insights into the potential benefits and limitations of timeshifting strategies, fostering a deeper understanding of their applicability in real-world computing scenarios.

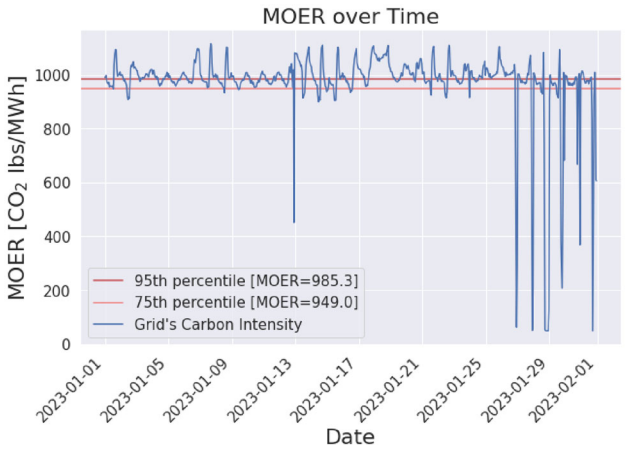

Fig. 9 Grid emissions for California ISO North in January

We examine a system made up of 1000 nodes where each node contains 2 CPUs and 8 GPUs. We assume, for the sake of this exploration that each CPU consumes 10 watts in idle and 271 watts when fully utilized. Each GPU consumes 15 watts in idle and 700 watts when fully utilized. Our system is located in California ISO North and we would like to run the workload in January 2023.

The grid emissions in January are fairly consistent with the 95th percentile of emissions at 985 lbs/MWh and the 75th percentile emissions at 949 lbs/MWh (Fig. 9).

We have two workloads that we need to run at this time. One workload takes 300 h to run and the other takes 720 h.

When simulating the 300-hour workload in this setting, the heatmap reveals intriguing data points. In Fig. 10, we observe that the highest emissions occur when we pause the workload at the 95th percentile and resume it at the 85th percentile of carbon intensity. We attribute this phenomenon to the fact that the energy mix at the 85th percentile could comprise both fossil fuels and renewable energy, leading to increased uncertainty when the system relies on renewable sources.

Simulating the 720-hour workload, on the other hand, reveals a more intuitive heatmap (Fig. 11) where the lower we set the threshold, the less emissions we have. In our experiment, the 720-hour workload has more than a 50% reduction in emissions at the 75th percentile of emissions.

These findings suggest that the effectiveness of timeshifting strategies can vary significantly depending on the duration of the workload. Workloads, such as the 720-hour one, seem to offer greater potential for emissions reduction through timeshifting, while shorter workloads may have limited gains. This observation underscores the importance of carefully tailoring timeshifting approaches to the specific characteristics and duration of workloads. By understanding these nuances, we can optimize the implementation of timeshifting techniques. We note that other papers (namely

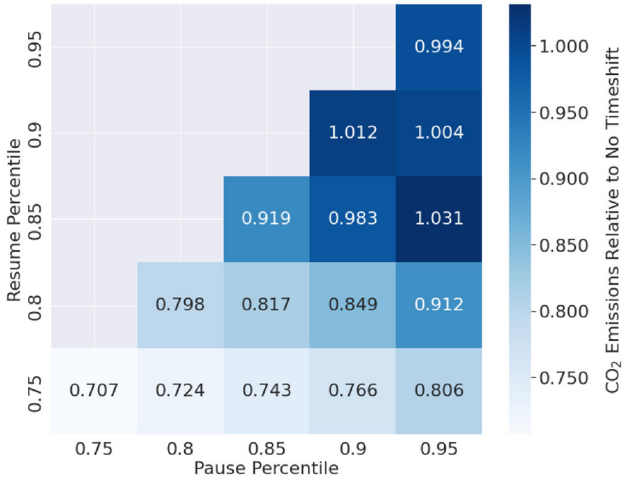

Fig. 10 300-hour workload heatmap of emissions relative to non-timeshifted workload

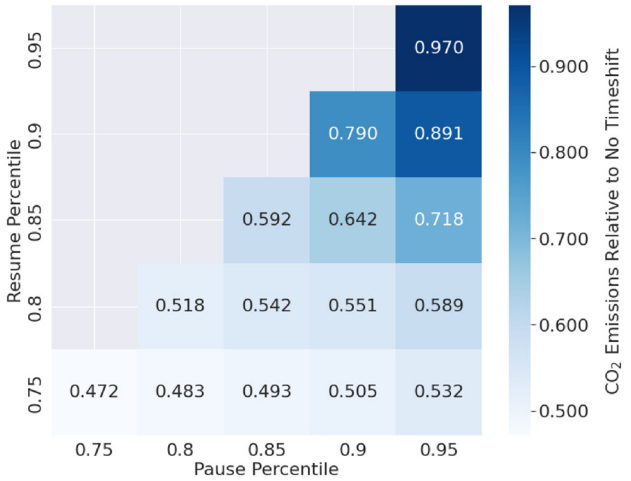

Fig. 11 720-hour workload heatmap of emissions relative to non-timeshifted workload

[20]) have found an advantage to timeshifting, especially for small workloads. We believe this discrepancy between our findings and theirs comes from the fact that we account for idle power consumption even when delaying the start time of the model. If the model has a short runtime, the energy savings for running at cleaner times is offset by the carbon cost of maintaining the idle system.

However, to get the lowest possible emissions out of both workloads, we extend the runtime by a nontrivial amount. For the 300-hour workload, the runtime increases by a factor of 7x (Fig. 12), resulting in only a 30% reduction in CO_2 emissions (Fig. 10). On the other hand, the 720-hour workload performs better, achieving over a 50% reduction in CO_2 emissions (Fig. 11) with a runtime extension of 4.3x (Fig. 13).

The two-threshold simulation offers valuable insights into the workload's behavior over time. Both the 300-hour workload (Fig. 14) and the 720-hour workload (Fig. 15) exhibit

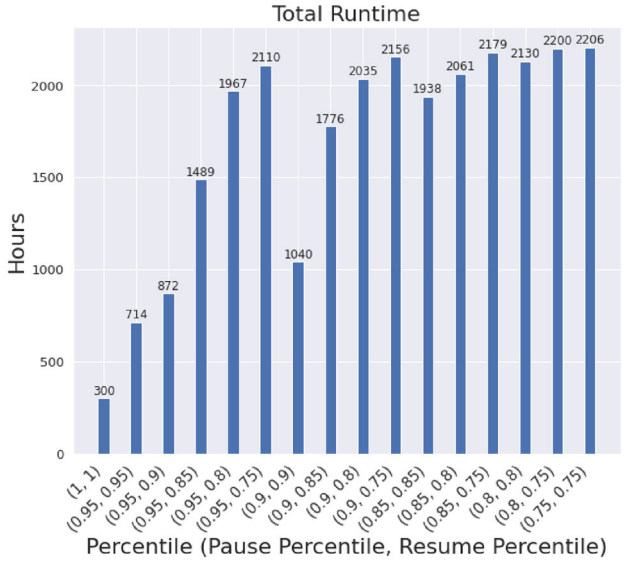

Fig. 12 Histogram of runtimes for timeshifting the 300-hour workload

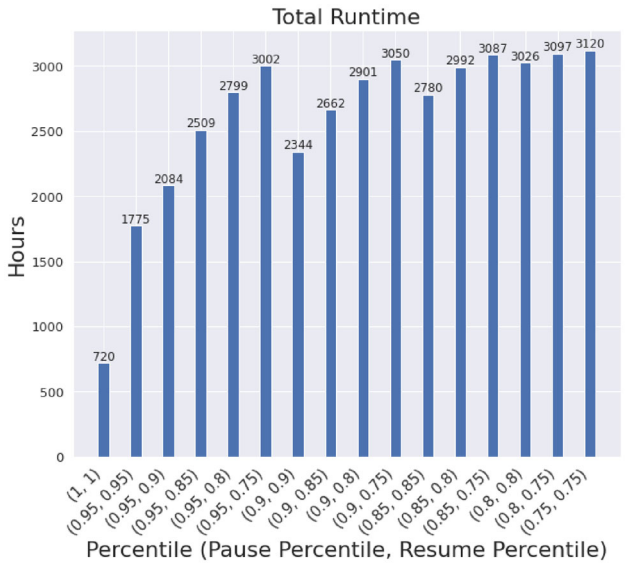

Fig. 13 Histogram of runtimes for timeshifting the 720-hour workload

frequent bursts of activity, contributing to longer runtimes. However, it's essential to note that the simulation has not yet accounted for the cost of caching memory frequently. Future work aims to refine our calculations by incorporating MOER thresholds that more accurately correspond to the renewable-to-non-renewable energy supply mix.

5.8 Exploring other locations

While our earlier analyses focused on California ISO North, we expanded our investigation to include other regions with distinctive energy profiles. By doing so, we aimed to unveil

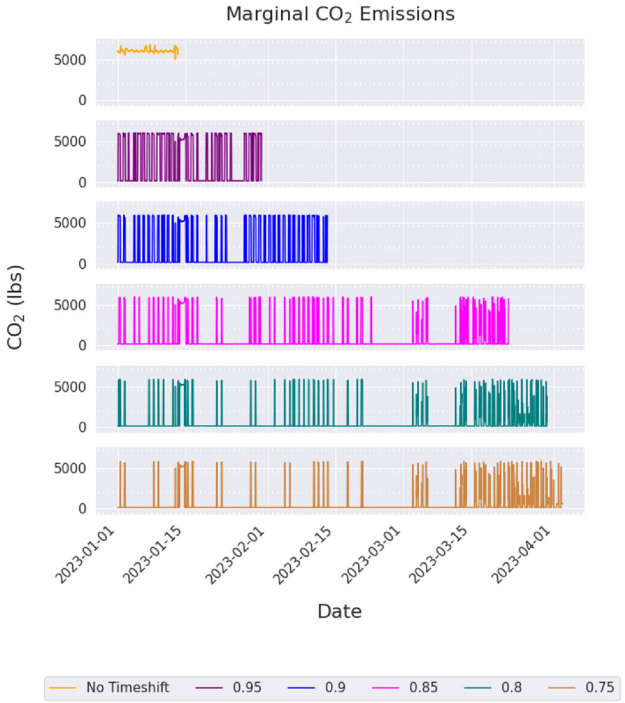

Fig. 14 A comparison of running the 300-hour workload with different thresholds. Each subplot represents the emissions over the course of the workload when the pause and resume thresholds are the same values

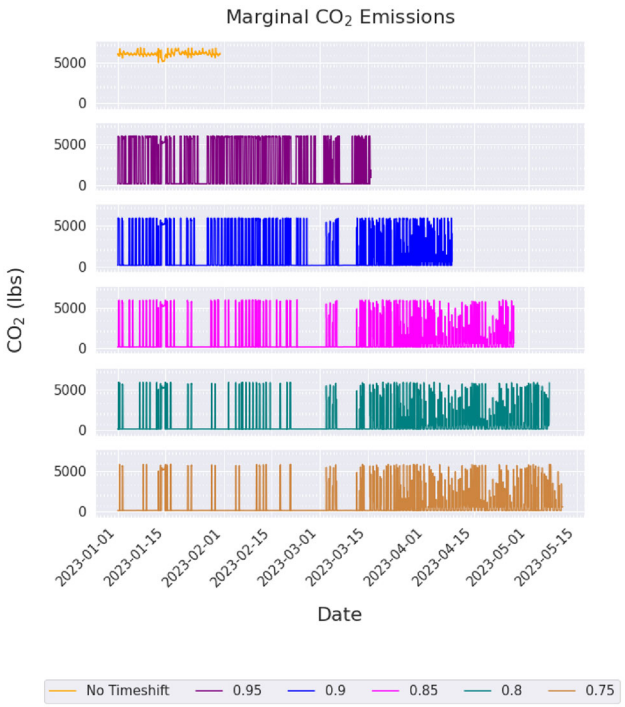

Fig. 15 A comparison of running the 720-hour workload with different thresholds. Each subplot represents the emissions over the course of the workload when the pause and resume thresholds are the same values

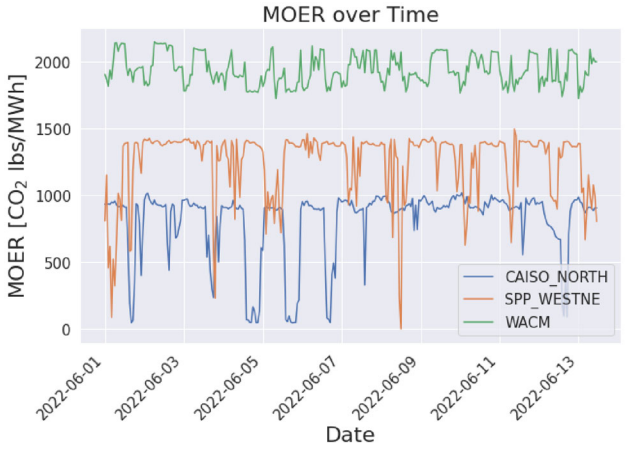

Fig. 16 Carbon intensity of California ISO North (CAISO_NORTH), SPP West Nebraska (SPP_WESTNE), and WAPA Rocky Mountain Region (WACM) grid regions from June 1, 12 AM–June 13, 12 PM (300 h)

the influence of varying carbon intensities on the effectiveness of timeshifting strategies.

Keeping the same system definition and assuming that the data our workload needs is already present at the new locations, we compare the CO_2 savings when running workloads in June 2022 in regions covered by the power grids California ISO North (CAISO_NORTH), SPP Western Nebraska (SPP_WESTNE), and WAPA Rocky Mountain Region (WACM).

Figure 16 depicts the grid intensity patterns of the respective areas during a 300-hour period from June 1, 12 AM to June 13, 12 PM. In the case of California ISO North, there is a noticeable level of variability, with MOER values fluctuating between 0 and approximately 1000 lbs CO_2/MWh. SPP West Nebraska exhibits even greater variability, with grid intensity ranging between 0 and 1400 lbs CO_2/MWh. On the other hand, WAPA Rocky Mountain Region demonstrates comparatively lower variability but higher MOER values, spanning from 1700 to 2200 lbs CO_2/MWh. Based on the grid intensity insights, it seems probable that California ISO North will encounter more occurrences of timeshifting compared to the WAPA Rocky Mountain Region.

Running our simulation, we generate histograms of the amount of carbon saved in each region for various pause/resume thresholds (CAISO_NORTH—Fig. 17, SPP_WESTNE—Fig. 18, WACM—Fig. 19). Interestingly, we see that no matter how you implement timeshifting in CAISO_NORTH, there will be some carbon savings. In the case of SPP_WESTNE, significant carbon savings appear only upon reaching the 85th percentile of carbon intensity in the region. However, this region presents ample potential for curbing carbon emissions, particularly at lower carbon intensity thresholds. When juxtaposed with the CAISO_NORTH results, there is a 25% reduction in carbon emissions. Con-

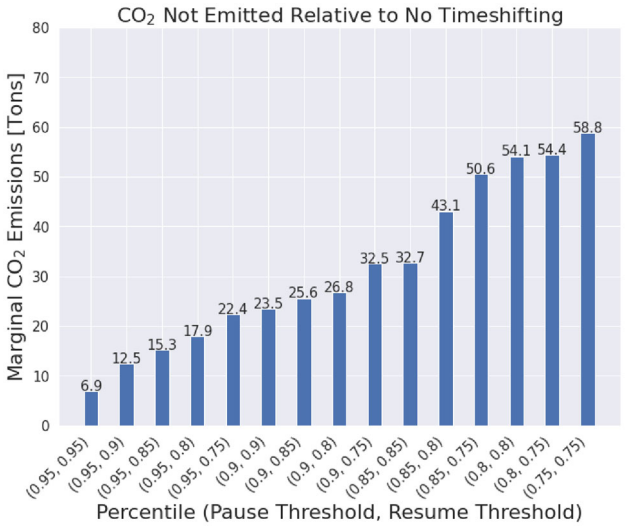

Fig. 17 Histogram of tons of carbon not emitted compared to non-timeshifted workload in California ISO North (CAISO_NORTH) starting in June 2022

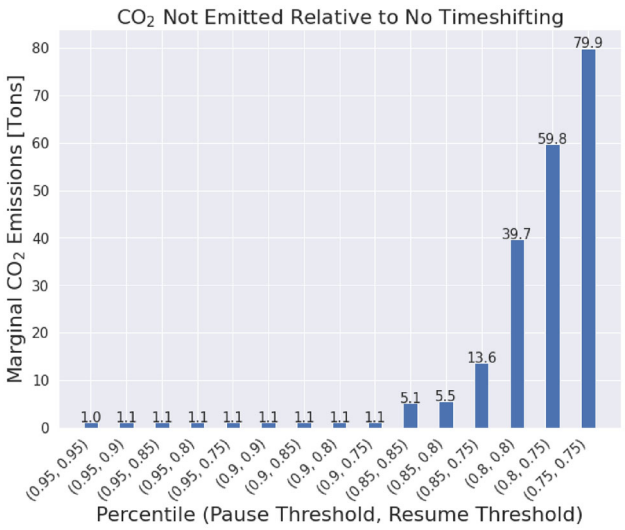

Fig. 18 Histogram of tons of carbon not emitted compared to non-timeshifted workload in SPP West Nebraska (SPP_WESTNE) starting in June 2022

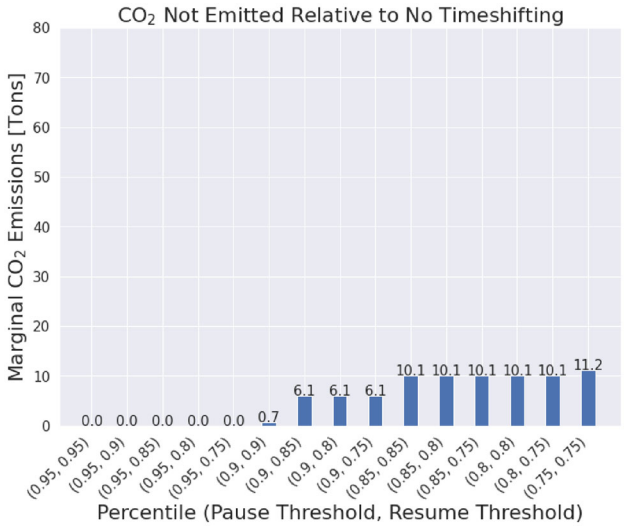

Fig. 19 Histogram of tons of carbon not emitted compared to non-timeshifted workload in WAPA Rocky Mountain Region (WACM) starting in June 2022

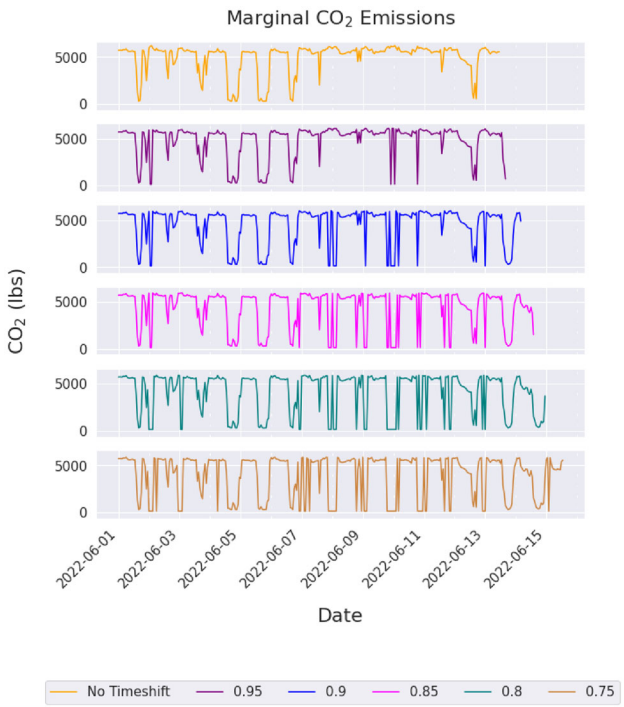

Fig. 20 Time series of the workload running in California ISO North (CAISO_NORTH) starting in June 2022. From top to bottom: non-timeshifted, pause/resume=0.95, pause/resume = 0.9, pause/resume = 0.85, pause/resume = 0.8, pause/resume = 0.75

versely, there is minimal influence of timeshifting on carbon reduction in WACM.

Looking closer at how the workload is paused and resumed over the course of time (Figs. 20, 21, 22), we see some interesting patterns. CAISO_NORTH has more sporadic timeshifted segments; in fact, the pauses are hard to distinguish from the natural fluctuations in the grid. Furthermore, for each percentile we examined, there is a consistent upward trend in the duration of the workload resulting in it extending its runtime.

We also see that the workload does not get timeshifted after a while in WACM. This observation challenges the thresholds selected based on historical data, suggesting that they may not accurately capture this specific period. Consequently, WACM necessitates a more sophisticated approach to implementing carbon-aware applications. A timeshifting technique is probably not the right solution for this region as there is no good time to pause/resume the workload.

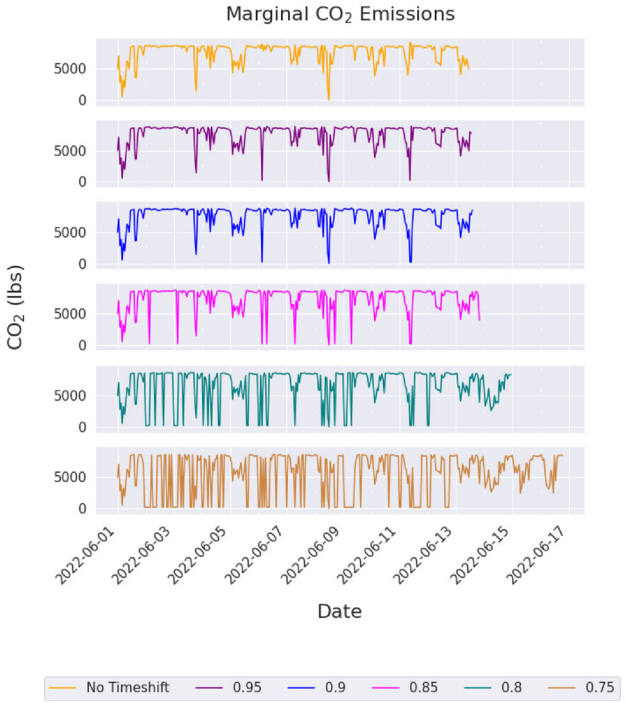

Fig. 21 Time series of the workload running in SPP West Nebraska (SPP_WESTNE) starting in June 2022. From top to bottom: non-timeshifted, pause/resume = 0.95, pause/resume = 0.9, pause/resume = 0.85, pause/resume = 0.8, pause/resume = 0.75

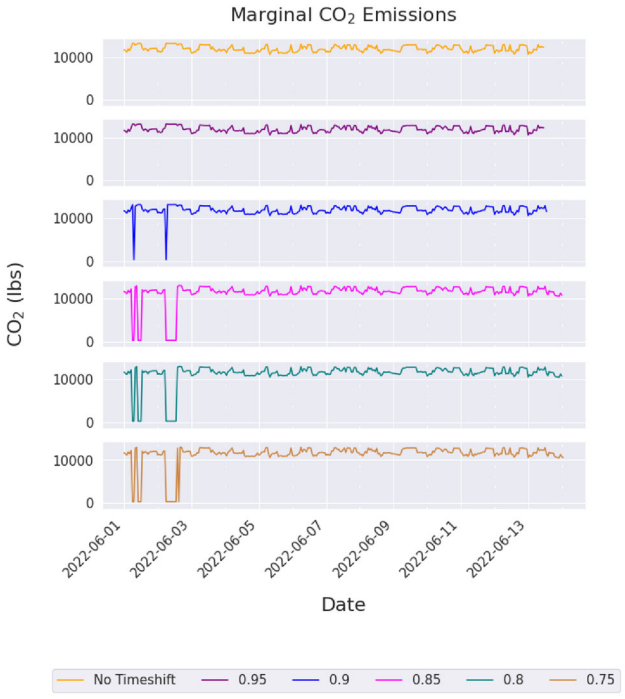

Fig. 22 Time series of the workload running in WAPA Rocky Mountain Region (WACM) starting in June 2022. From top to bottom: non-timeshifted, pause/resume = 0.95, pause/resume = 0.9, pause/resume = 0.85, pause/resume = 0.8, pause/resume = 0.75

From our observations, timeshifting involves more than merely selecting an optimal time for running a workload; it requires a thorough comprehension of the energy mix in the supplying power grid of that region. Running a carbon-aware application that utilizes timeshifting in a region with limited renewable energy will inadvertently increase carbon emissions. As more data becomes available about the energy mix from the power grid, it is important to align our timeshifting strategies with the local energy landscape. Only then can we maximize the carbon-reducing potential of our applications.

6 Future work

There are several avenues for extending and enhancing the capabilities of our estimation tool and simulation to provide more comprehensive insights into the carbon emissions of long-running workloads.

1. Utilizing Power Curves: Currently, our estimation tool relies on the Thermal Design Power (TDP) values provided by hardware manufacturers. In future work, we can enhance the accuracy of our energy consumption estimates by incorporating power curves distributed by manufacturers. Power curves offer a more detailed representation of a hardware component's power consumption across various utilization levels, enabling a more fine-grained estimation of energy usage during workload execution.
2. Scaling Hardware and Networking Costs: To capture a more realistic and complete view of energy consumption, we can expand our simulation to account for larger-scale hardware configurations. Including various hardware components and networking costs will provide a more comprehensive analysis of the energy requirements and carbon emissions associated with different workload setups. This extension will be particularly useful for understanding the impact of workloads in distributed computing environments.
3. Cost of Idle Power: In our early exploration of the timeshifting work, we found that idle power consumption has a large impact on carbon savings especially with an optimized timeshifting strategy. While we have shed some light on the impact of idle power, there remains an opportunity to delve deeper into this phenomenon. Exploring how idle power varies across different times of the day, workload types, and hardware utilization and configurations can help us design better software and systems.
4. Fidelity and Access of Data from the Power Grid: While our current framework integrates valuable insights into carbon emissions and energy intensity, there exists an opportunity to delve deeper into the intricacies of power grid dynamics. Different regions and organizations may

adopt different calculation methods, leading to inconsistencies that can impact the accuracy of our simulation. Understanding the variations in calculations can help us increase the accuracy and reliability of our simulation. Additionally, data from the power grid in relation to carbon emissions from renewable resources is not widely available or reported. Global coverage of emissions data is poor, access to information on power usage and source behind the meter is limited, and geographic smoothing often makes access to individual emissions impossible. Increasing access to data around power, and increasing granularity of where power comes from, could result in more attention to and potential solutions for how to effectively minimize carbon contributions of AI workloads.

Future work must consider how the role of timeshifting is impacted by differences in grid variability and intermittent access to renewable workloads. We saw many cases where timeshifting a workload would reduce the amount of carbon released into the atmosphere, but we also saw cases in which timeshifting was not helpful in minimizing emissions. We believe the usefulness of timeshifting is directly related to grid variability. This suggests two future directions for this work. One direction is to characterize the types of grids where timeshifting is most amenable and the second direction is to develop adaptive algorithms similar in their adaptability to the timeshifting algorithm presented here that effectively work to reduce carbon emissions on grids where timeshifting is not applicable.

Beyond extending our system estimation tool or additional algorithms that adapt to the variability of emissions, we also must consider extending our estimations to include more than just the cost of *training* these LLMs, as there are many other important aspects of the development and deployment of LLMs that can substantially contribute to carbon emissions. Inference has its own carbon contribution that can vary greatly by how LLMs are integrated with other types of systems and marketed to end users. Further, we neglect the carbon contribution of model development and research that lead to, for example, the development and success of Meta AI's LLaMa model. To fully understand the impact and role of LLMs in contributing to greenhouse gas emissions, we need a unified way of calculating carbon emissions at these, and many other, stages. Beyond simply considering the carbon cost of development and deployment of LLMs, we also need to consider the emissions released in the production of the hardware used to train these LLMs.

7 Conclusion

Our work provides a back-of-the-envelope calculation of the energy cost of training a long-running LLM . We enhance this calculation by grounding it in power-grid data to estimate the carbon cost of model training. We also propose a timeshifting algorithm that pauses training of the model during times when the grid is particularly dirty and resumes when the carbon cost of the grid is below a given threshold. We show how the computational requirements of the model, the location where the model is trained, and even when the model is trained all have a dramatic influence on the carbon cost of LLMs.

Specifically, this paper highlights the significance of timeshifting for long-running workloads in LLMs, presenting a promising avenue for reducing carbon emissions. Our simulation offers valuable insights into the potential and limitations of timeshifting, revealing intriguing patterns that influence model performance and model runtime. Acknowledging the need for further advancements in carbon-aware model training, we envision incorporating more comprehensive system definitions and exploring energy mix dynamics to refine our approach as an area of future research. As we move forward, prioritizing carbon-aware algorithms and sustainable computing practices will play a pivotal role in mitigating the environmental impact of LLMs and other workloads, fostering a greener and more sustainable computing landscape.

Acknowledgements We extend our gratitude to WattTime for their support and collaboration. Their dedication to advancing carbon transparency has been instrumental in shaping the research presented in this paper.

Author Contributions AJ coded timeshifting. AJ and NB wrote the main text, reviewed the code, and generated images. DN provided background and history for the paper, and a portion of the editing. SC provided the initial idea and direction for the paper. All authors reviewed the text.

Funding Not applicable.

Availability of data and materials This paper extensively uses WattTime to access the carbon intensity of the power grids. Access to this API requires a subscription or data requests from WattTime. However, individuals can replicate our work with free access to historical data from California ISO North. Upon completion of the necessary review process at Intel, we intend to release our estimation tool as an interactive Jupyter notebook through GitHub following this publication.

Declarations

Conflict of interest The authors of this paper are full-time employees of Intel Corporation. The research project received no external sources of funding.

Ethical Approval Not applicable.

Appendix A: Pseudocode for two-threshold MOER value selection

```
# Define the thresholds for pausing
# and resuming the workload
t1 := pause_threshold
t2 := resume_threshold
paused := False

# Loop through the MOER values
for M in MOER_vals:
    if M > t1:
        paused = True
    elif not paused and M < t1:
        paused = False
    elif paused and M > t2:
        paused = True
    elif paused and M < t2:
        paused = False
```

References

1. Strubell E, Ganesh A, McCallum A (2020) Energy and policy considerations for modern deep learning research. In: Proceedings of the AAAI conference on artificial intelligence, VOL 34(09), PP 13,693–13,696
2. Strubell E, Ganesh A, McCallum A (2019) Energy and policy considerations for deep learning in NLP. arXiv preprint arXiv:1906.02243
3. Green Software Foundation, Green Software Practitioner: Carbon Awareness. [Online]. Available: https://learn.greensoftware.foundation/carbon-awareness
4. Tiple V (2020) Recommendations on the European commission's white paper on artificial intelligence-a European approach to excellence and trust, com (2020) 65 final (the'AI white paper)
5. Stokel-Walker C (2023) Turns out there's another problem with AI—its environmental toll. The Guardian. [Online]. Available: https://www.theguardian.com/technology/2023/aug/01/techscape-environment-cost-ai-artificial-intelligence
6. Patterson D, Gonzalez J, Hölzle U, Le Q, Liang C, Munguia L-M, Rothchild D, So DR, Texier M, Dean J (2022) The carbon footprint of machine learning training will plateau, then shrink. Computer 55(7):18–28
7. Galvin R (2015) The ICT/electronics question: structural change and the rebound effect. Ecol Econ 120:23–31
8. Sanh V, Debut L, Chaumond J, Wolf T (2019) DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108
9. Fedus W, Zoph B, Shazeer N (2022) Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. J Mach Learn Res 23(1):5232-5270
10. Choromanski K, Likhosherstov V, Dohan D, Song X, Gane A, Sarlos T, Hawkins P, Davis J, Mohiuddin A, Kaiser L et al (2020) Rethinking attention with performers. arXiv preprint arXiv:2009.14794
11. Antonello R, Beckage N, Turek J, Huth A (2021) Selecting informative contexts improves language model finetuning. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, pp 1072–1085
12. Tay Y, Dehghani M, Bahri D, Metzler D (2020) Efficient transformers: a survey. arXiv preprint arXiv:2009.06732
13. Bommasani R, Hudson DA, Adeli E, Altman R, Arora S, von Arx S, Bernstein MS, Bohg J, Bosselut A, Brunskill E et al (2021) On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258
14. Bannour N, Ghannay S, Névéol A, Ligozat A-L (2021) Evaluating the carbon footprint of NLP methods: a survey and analysis of existing tools. In: Proceedings of the second workshop on simple and efficient natural language processing, pp 11–21
15. Lacoste A, Luccioni A, Schmidt V, Dandres T (2019) Quantifying the carbon emissions of machine learning. arXiv preprint arXiv:1910.09700
16. Lannelongue L, Grealey J, Inouye M (2021) Green algorithms: quantifying the carbon footprint of computation. Adv Sci 8(12):2100707
17. Patterson D, Gonzalez J, Le Q, Liang C, Munguia L-M, Rothchild D, So D, Texier M, Dean J (2021) Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350
18. Cao Q, Balasubramanian A, Balasubramanian N (2020) Towards accurate and reliable energy measurement of NLP models. arXiv preprint arXiv:2010.05248
19. Henderson P, Hu J, Romoff J, Brunskill E, Jurafsky D, Pineau J (2020) Towards the systematic reporting of the energy and carbon footprints of machine learning. J Mach Learn Res 21(1):10 039-10 081
20. Dodge J, Prewitt T, Tachet Des Combes R, Odmark E, Schwartz R, Strubell E, Luccioni AS, Smith NA, DeCario N, Buchanan W (2022) Measuring the carbon intensity of ai in cloud instances. In: 2022 ACM conference on fairness, accountability, and transparency, pp 1877–1894
21. Touvron H, Lavril T, Izacard G, Martinet X, Lachaux M-A, Lacroix T, Rozière B, Goyal N, Hambro E, Azhar F, Rodriguez A, Joulin A, Grave E, Lample G (2023) LLaMA: open and efficient foundation language models. arXiv:2302.13971
22. Green Software Foundation, Green Software Practioner: Measurement. [Online]. Available: https://learn.greensoftware.foundation/measurement
23. WattTime, The Power to Choose Clean Energy. [Online]. Available: https://www.watttime.org/
24. O. US EPA, Download data. May 2022. [Online]. Available: https://www.epa.gov/egrid/download-data
25. California Energy Commission, 2021 Total System Electric Generation. [Online]. Available: https://www.energy.ca.gov/data-reports/energy-almanac/california-electricity-data/2021-total-system-electric-generation
26. "CodeCarbon." [Online]. Available: https://mlco2.github.io/codecarbon/
27. EPA Clean Air Markes Program Data (CAMPD), Power plant emissions, compliance, and allowance data. [Online]. Available: https://campd.epa.gov
28. WattTime, Marginal Emissions Methodology. [Online]. Available: https://www.watttime.org/marginal-emissions-methodology/

29. EPA Office of Air and Radiation (OAR), Greenhouse Gas Equivalencies Calculator. 2015. [Online]. Available: https://www.epa.gov/energy/greenhouse-gas-equivalencies-calculator
30. Radovanović A, Koningstein R, Schneider I, Chen B, Duarte A, Roy B, Xiao D, Haridasan M, Hung P, Care N, Talukdar S, Mullen E, Smith K, Cottman M, Cirne W (2023) Carbon-aware computing for datacenters. IEEE Trans Power Syst 38(2):1270-1280