Learning to Reduce Data Center Carbon Emissions via Forecast-Integrated Workload Scheduling

Amirhossein Sohrabbeig, Omid Ardakanian, and Petr Musilek
Department of Electrical and Computer Engineering, University of Alberta,
Edmonton, AB, Canada

***Abstract*—Data centers power large-scale AI and cloud services but contribute substantially to global carbon emissions, especially in regions where electricity generation remains heavily dependent on fossil fuels. To reduce these emissions, AI training jobs, which are not interactive or latency-sensitive, can be assigned to data centers in regions with cleaner energy mixes. We propose a differentiable, emission-aware scheduler that integrates regional carbon intensity forecasts generated by a time-series model. Unlike conventional predict-then-optimize methods, our approach trains the forecasting model end-to-end, directly optimizing for the downstream decision objective. Experimental results show that this end-to-end design achieves significantly larger emission reductions than models optimized solely for forecasting accuracy, highlighting the importance of coupling prediction and scheduling in carbon-aware data center operations.**

***Index Terms*—AI Data Centers, Decision-focused Learning**

I. Introduction

The rapid proliferation of AI-driven applications, from text, code, image, and video generation to scientific discovery, has dramatically increased the demand for computational resources, resulting in a surge in data center energy consumption. According to the International Energy Agency (IEA), global data center electricity consumption has risen rapidly and is projected to more than double from 415 TWh in 2024 to around 945 TWh by 2030, increasing its share of global electricity use from 1.5% to nearly 3% [1]. Meanwhile, power grid expansion and renewable energy deployment have struggled to keep pace, resulting in significant variability in carbon intensity across regions and times of day. Consequently, reducing the carbon footprint of data centers has become imperative for hyperscale cloud providers seeking to meet sustainability targets while maintaining service-level agreements (SLAs).

Carbon-aware workload scheduling is one of the most effective strategies for mitigating emissions from data centers. By shifting or adapting computation based on spatial [2], [3] and temporal variations [4], [5] in carbon intensity, hyperscale cloud providers can better align their data center energy consumption with periods and locations of cleaner electricity supply, without compromising performance. This approach is particularly well suited for long-running AI training workloads, where network latency differences are negligible compared to total execution time, yet carbon intensity fluctuations over multi-hour intervals can substantially affect cumulative emissions. For such workloads, spatial workload shifting across geo-distributed data centers enables meaningful reductions in overall carbon emissions [3].

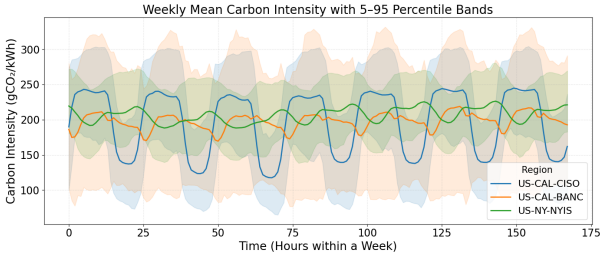

Fig. 1. Average carbon intensity in three U.S. regions. Shaded bands show the range between the 5th and 95th percentile of the carbon intensity distribution.

Prior studies have proposed schedulers that use current regional carbon intensity and available data center capacity to assign workloads to the lowest-emission site [6], [7]. However, as shown in Figure 1, regional carbon intensity traces often intersect over multi-hour time scales—relevant for AI training workloads—making it difficult to select the optimal site based solely on present conditions. This underscores the importance of carbon-intensity forecasting, the central problem addressed in this paper. We focus on assigning long-running training jobs to geographically distributed data centers operated by a single cloud provider. Rather than shifting workloads in time, we leverage region-specific carbon-intensity forecasts to optimize job placement at submission time, considering job deadlines, resource availability, and hardware constraints to minimize total emissions while meeting performance targets.

The regional carbon intensity forecasting model can be trained using two primary approaches. The first, known as predict-then-optimize, minimizes forecast error independent of the scheduling task. The second, a decision-focused approach, directly optimizes the downstream objective, i.e. minimizing the increase in total data center emissions resulting from forecast inaccuracies. Our main contribution is to examine whether integrating forecasting and decision-making within a unified, differentiable framework yields better scheduling outcomes than traditional two-stage (predict-then-optimize) methods. The results show that higher forecast accuracy does not necessarily translate into lower emissions. Across three workload datasets and using granular carbon-intensity traces from ElectricityMaps [8], the proposed end-to-end approach achieves a 1–1.4% greater reduction in emissions-related relative regret than the best two-stage method, a difference that translates into a significant cut in total emissions at scale.

This work has been supported by the Resilient and Clean Energy Systems Initiative under the MIF program of the Government of Alberta, Canada.

979-8-3315-8138-1

II. Model and Assumptions

The training workload consists of jobs submitted to the data center service provider at different times. Each job consists of multiple instances representing independently schedulable execution units, each requiring the same amount of computational resources and running on a single machine in a data center. We assume that job characteristics, including the number of instances and GPU requirements, are known when a job is submitted. Execution times are also assumed to be known as they can be accurately predicted from historical data. This is consistent with prior evidence that most ML tasks are predictable within 25% error [9].

The proposed scheduler assigns the submitted jobs to data centers located in different regions, each with a distinct supply mix. It cannot delay the execution of a job once it is assigned to a data center, nor can it temporarily pause the job execution. Since network latency and migration costs are negligible for long-running training jobs, the scheduling decision depends solely on resource availability and predicted carbon intensity during job execution. All instances of a job must be assigned to the same data center and launched simultaneously, potentially on different machines, an approach known as gang scheduling.

The power consumption of a machine is assumed to be proportional to its utilization (i.e. energy proportionality):

$$P_m = P_m^{\text{idle}} + \alpha_m L_m, \tag{1}$$

where P_m^{idle} is idle power, α_m is the load coefficient, and L_m is the utilization of machine m. Each data center contains identical machines, however machine types, and thus idle power and load coefficients, may differ across data centers.

III. Problem Formulation

Let $\mathcal{D} = \{1, \ldots, D\}$ denote the set of data centers operated by a hyperscale cloud provider. Each data center $i \in \mathcal{D}$ contains a set of machines $\mathcal{M}_i$ and the total number of machines across all data centers is $M = \sum_{i=1}^{D} |\mathcal{M}_i|$. The matrix $\mathrm{Q} \in \{0,1\}^{M \times D}$ maps machines to data centers: $\mathrm{Q}_{m,i} = 1$ if machine m belongs to data center i and is 0 otherwise. The per-machine marginal emissions cost vector, derived from the machine's power consumption and regional carbon intensity forecasts, is denoted as $\mathrm{c} \in \mathbb{R}^M_{\geq 0}$, and the per-machine available GPU capacity vector is denoted as $\mathrm{r} \in \mathbb{R}^M_{\geq 0}$.

A job consists of n identical instances, each requiring a fraction g of the GPU of the machine that it runs on. These instances must run on different machines in the same data center and finish execution before a deadline d. We use a binary vector $\mathrm{x} \in \{0,1\}^D$ to indicate the data center selected to run a submitted job, i.e. $\mathrm{x}_i = 1$ if the job is assigned to data center i and is 0 otherwise. We also use a vector of non-negative integers $\mathrm{u} \in \mathbb{N}^M$ to represent the number of instances assigned to a machine in any of the data centers.

The assignment of a job to a particular data center affects the total execution time of that job as data centers contain different types of machines. We use the vector $\mathrm{t} \in \mathbb{R}^D$ to collect the expected execution time of a job in every data center.

We now present the design of our carbon-aware scheduler which aims to minimize the expected increase in carbon emissions across all data centers, represented compactly as $\mathrm{c}^\top \mathrm{u}$, while ensuring that only one data center is selected (2b), all job instances are scheduled (2c) and they are assigned to machines within the selected data center (2d), GPU/server capacity constraints are respected (2e), and the job execution finishes before its deadline (2g). Specifically, the scheduler solves the following combinatorial optimization problem (which is an integer linear program) to assign a newly submitted job to one of the data centers and the corresponding job instances to a subset of machines within that data center:

$$\min_{\mathrm{x},\mathrm{u}} \; \mathrm{c}^\top \mathrm{u} \tag{2a}$$

$$\text{s.t.} \quad \mathbf{1}^\top \mathrm{x} = 1 \tag{2b}$$

$$\mathbf{1}^\top \mathrm{u} = n \tag{2c}$$

$$\mathrm{u} \preceq n \cdot \mathrm{Q}\,\mathrm{x} \tag{2d}$$

$$g \cdot \mathrm{u} \preceq r \tag{2e}$$

$$\mathrm{t}^\top \mathrm{x} \leq d \tag{2g}$$

IV. Methodology

Solving the above Integer Linear Programming (ILP) for the carbon-aware scheduling requires knowledge of the marginal emissions cost vector c. Computing this cost for each of the D data centers involves summing the product of the regional carbon intensity forecast and the marginal power consumption of the machines executing the job instances in that data center for each time slot over the job's runtime.

Let $\hat{I}_h(i)$ be the carbon intensity forecast of data center i containing machine m, in time slot h. With unit-length time slots, the marginal emissions cost for running one instance to completion on machine m is given by:

$$\mathrm{c}_m \;=\; \alpha_m\, g \sum_{h=t_0}^{t_0+\mathrm{t}_i} \hat{I}_h(i). \tag{3}$$

Thus, the scheduling and carbon-intensity forecasting problems are inherently coupled, as forecasts directly influence the optimal workload assignment and resulting emissions.

We adopt the DLinear [10] neural network to predict future regional carbon intensity based on historical observations. Rather than training this model independently by minimizing a prediction loss, e.g. mean squared error (MSE), and then solving the optimization problem using the resulting forecasts, we attach a *differentiable decision head* that approximates the optimization mapping and train the whole model in an end-to-end fashion. We call this decision-focused learning approach "Diff. Head". The decision head comprises two modules: a feasibility checker that determines whether an incoming job satisfies machine- and data-center-level constraints given the current system state and job requirements, and an estimator that computes the marginal emissions cost of assigning the job to each data center using forecast carbon intensities and

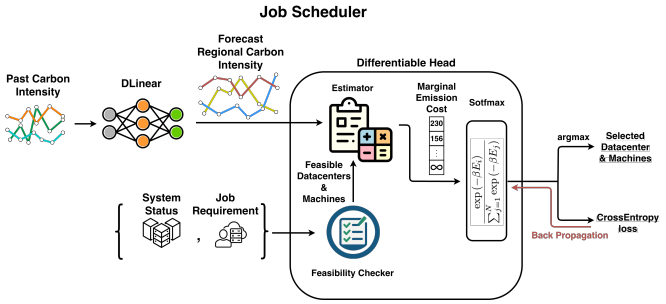

Fig. 2. The proposed job scheduler architecture. DLinear's regional carbon intensity predictions are fed to a differentiable decision head which considers job requirement and system state to estimate marginal emissions for all feasible options, and assigns each job to the data center and machines that have the lowest marginal emissions. The scheduler is trained end-to-end.

feasibility outcomes. Infeasible assignments are assigned infinite cost. Thus, the forecasting model outputs regional carbon intensity predictions, which are passed to the decision head to compute marginal costs for all feasible scheduling options; a softmax layer then produces a probability distribution over these options, ensuring differentiability for end-to-end training. Figure 2 shows the architecture of the proposed scheduler.

Since the scheduler must select a data center and allocate machines for each arriving job, the decision-making is inherently a classification task. We, therefore, minimize a task-specific objective (cross-entropy loss) that directly reflects the downstream objective, i.e. reducing total carbon emissions, allowing the forecasting model to learn representations most useful for decision-making, even if it sacrifices prediction accuracy in some cases. To generate labels for incoming jobs, ground-truth carbon intensity values are provided to the Diff. Head to find data centers with the minimum cost among those that have enough capacity. During training, DLinear's carbon intensity predictions are fed into the Diff. Head, and the resulting outputs are compared with the ground-truth labels using cross-entropy loss to update the model parameters.

V. Experiments

A. Datasets

We evaluate the proposed carbon-aware scheduler at a continental scale using the U.S. carbon intensity data and three production GPU traces (see Table I). Carbon-intensity data for 51 U.S. regions in 2021 are obtained from Electricity Maps [8]. We use both hourly and 5-minute datasets: the 5-minute data are used in forecasting for jobs with runtime under 6 hours to ensure finer-grained control, while the hourly data are used for longer jobs to reduce computational overhead. For the workload traces, we retain only GPU jobs with valid completion status. All traces are shifted to 2021 with weekday alignment preserved to match the carbon dataset. A 80:10:10 split is used for training, validation, and testing.

The Alibaba PAI trace [9] consists of production ML workloads from over 1,300 users. As the number of data centers is not disclosed, we assume it is equal to the number of machine types in the trace, which is five. The Helios trace [11] covers four clusters and includes diverse ML workloads (e.g., training, inference, preprocessing, quantization). We exclude short jobs (mostly inference) that finish in less than one hour. The Philly trace [12] is modeled as 14 independent data centers to study spatial scheduling across multi-tenant environments. Since data center locations are not disclosed in these traces, we map each data center to one of the 51 U.S. regions in our carbon-intensity dataset. Specifically, for each trace, we select n regions (matching its number of data centers) whose carbon-intensity profiles intersect frequently, making the data-center selection non-trivial and sensitive to carbon intensity forecasts.

TABLE I
Comparison of cluster workload traces.

Dataset	Period	Infrastructure size	# Jobs
Alibaba	2 months	6,742 GPUs, 1,897 nodes	1.2M jobs
Helios	6 months	6,416 GPUs, 802 nodes	3.36M jobs
Philly	75 days	2,490 GPUs, 552 nodes	100K jobs

B. Baseline Methods

We compare our approach with baselines representing other approaches to carbon-aware scheduling, grouped as follows:

1) Predict-And-Optimize (PNO): These baselines represent state-of-the-art decision-focused learning methods capable of handling downstream tasks formulated as combinatorial optimization problems. Unlike our Diff. Head approach, they explicitly solve the optimization problem using Gurobi in the forward pass. Nevertheless, all methods train the same forecasting model (DLinear) in an end-to-end fashion.

- **SPO+** [13]: Uses a regret-based surrogate. For each cost prediction $\hat{c}$, it solves the ILP once with a perturbed cost vector $2\hat{c} - c$, and the difference between the resulting decisions forms the SPO+ gradient direction. This provides an upper bound on regret and enables end-to-end training without differentiating through the ILP.
- **NID** [14]: Applies the Negative Identity Differentiation rule, treating the ILP solver as a negative identity map during backpropagation. The loss signal associated with the solver's chosen decision is passed directly back to the predicted costs with a sign flip, yielding a lightweight surrogate for linear-objective combinatorial problems while still using the exact ILP in the forward pass.

2) Predict-Then-Optimize (PTO): These baselines use the same DLinear model but follow the traditional two-stage pipeline: DLinear is first trained by minimizing the MSE loss on the carbon intensity dataset, then the trained model produces $\hat{I}$, which is used to calculate c plugged into Problem (2). This optimization problem is then solved by Gurobi. Thus, the training of DLinear remains independent of the scheduling problem. We consider two variants of the forecasting model:

- **Slot-level forecasting:** The forecasting model $\mathcal{F}_\theta$ outputs carbon-intensity predictions over a horizon H, determined by the maximum job execution time. Specifically, it maps the most recent l observations to a forecast vector of length H, which is subsequently used in Eq. (3):

$$\widehat{I}_{t_0+1:t_0+H} = \mathcal{F}_\theta(I_{t_0-l+1:t_0})\,. \tag{4}$$

- **Interval-average forecasting:** The forecasting model $\mathcal{F}_\phi$ outputs the average carbon-intensity value over the job's execution interval, effectively approximating the summation in Eq. (3). This baseline is inspired by prior work [3].

$$\widehat{I}_{\text{avg}} = \mathcal{F}_\phi(I_{t_0-l+1:t_0}), \tag{5}$$

where $\widehat{I}_{\text{avg}}$ is a scalar representing the average intensity forecast for $[t_0{+}1,\, t_0{+}H]$.

3) Naive Baselines: These are simple methods for calculating c, which provide reference points for comparison:

- **Repeat-Last:** Repeats the most recent observed carbon intensity for all future steps.
- **Recent-Average:** Uses the average of recent carbon intensity values within a look-back window.
- **Random:** Samples randomly from the range of carbon intensity values observed recently.
- **Worst-Case:** At each step, selects the data center with the highest emissions according to ground-truth data.

These baselines solve the same optimization problem, with no training involved.

C. Evaluation Metrics

We consider four metrics for evaluation. The symmetric Mean Absolute Percentage Error (sMAPE, computed as $\frac{2}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t|/(|y_t| + |\hat{y}_t|)$) measures the forecast accuracy by comparing predicted and actual carbon intensity values. The remaining metrics quantify *relative regret* with respect to an oracle that assigns jobs to data centers using knowledge of future carbon intensity values for all regions.

Concretely, the relative regret can be defined with respect to carbon emissions, energy consumption, and completion time:

$$\text{RR}_X = \frac{X - X_{\text{oracle}}}{X_{\text{oracle}}}, \quad X \in \{\text{Emission}, \text{Energy}, \text{Time}\}.$$

These three metrics quantify the suboptimality of a forecast-based decision relative to the full-information decision. Although the proposed scheduler optimizes for emissions only, we report the relative regret in energy consumption and job completion time to provide a more comprehensive assessment.

D. Experimental Results

Table II summarizes the results of our experiments, comparing our joint forecasting and workload scheduling approach with the baseline methods. Several key observations emerge:

- Emissions Reduction: The proposed method (Diff. Head) achieves the lowest relative regret in carbon emissions across all datasets (2.13% for Alibaba, 3.07% for Helios, and 4.80% for Philly), as shown in Figure 3. This shows the efficacy of end-to-end training for the carbon-aware scheduling task. Interestingly, while Diff. Head does not explicitly optimize energy consumption, it achieves the lowest RREnergy in one dataset (Philly), and the second lowest in another dataset (Alibaba).
- Forecast vs. Decision Quality: While the Slot-Level baseline achieves the lowest forecast error in two datasets, we consistently outperform it in terms of relative regret with respect to carbon emissions. This highlights a disconnect between forecast accuracy and downstream task performance, motivating our end-to-end learning approach. Specifically, for carbon-efficient workload scheduling, what matters is correctly identifying the most efficient data center; when regional carbon intensities are close, over- or underestimating future carbon intensity of one region could alter this ranking and degrade decision quality. Consequently, average forecasting error across regions is a poor proxy for decision quality.
- Comparison within PNO: Diff. Head attains the lowest emission regret among PNO methods across all datasets. Although all methods use the same forecaster trained end-to-end, Diff. Head bypasses solving the combinatorial optimization problem during training and uses a differentiable decision head that more effectively guides the training process toward emissions-minimizing decisions.
- Naive Methods: The Repeat-Last and Recent-Average baselines exhibit reasonable performance on some datasets, suggesting that recent carbon intensity values can be informative for shorter-term scheduling decisions. But, they are consistently outperformed by our method.
- Worst-Case Baseline: This baseline, which yields an upper bound on emissions, achieves extremely high regret (86.01%, 101.92%, and 73.01% for the three datasets), confirming the benefit of carbon-aware scheduling.

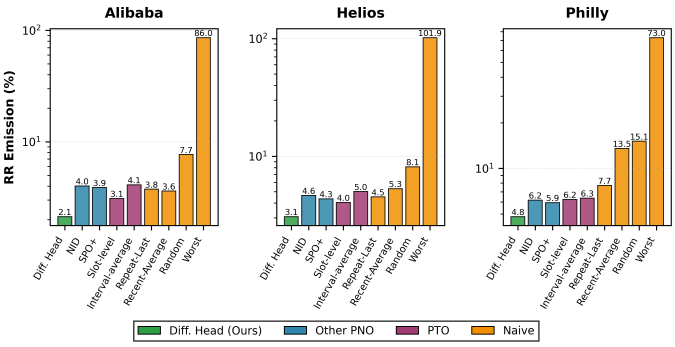

Fig. 3. Comparing the relative regret in carbon emissions (logarithmic scale), across the three datasets.

Figure 3 provides a visual comparison of the relative regret in carbon emissions across all methods and datasets. It reveals important patterns in performance: PTO methods are generally competitive with PNO methods with the exception of our method (Diff. Head) that achieves superior performance in all cases; naive baseline methods yield considerably higher regret.

E. Generalization Across Forecasting Models

To assess whether the observed advantage of the proposed method over the two-stage predict-then-optimize baseline is tied to a specific forecasting model (e.g. DLinear), we conduct additional experiments using other powerful forecasting models. In particular, we consider LSTM, TCN [15], AutoFormer [16] and PatchTST [17]. For each forecaster, we train the model on the same data and evaluate both the two-stage and end-to-end pipelines under identical conditions.

TABLE II

PERFORMANCE COMPARISON OF DIFF. HEAD WITH BASELINES. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD** AND SECOND-BEST RESULTS ARE UNDERLINED. PREDICTION LOSS FOR THE INTERVAL-AVERAGE METHOD IS MARKED WITH AN ASTERISK TO NOTE IT IS NOT COMPARABLE TO OTHERS.

Group	Method	Alibaba				Helios				Philly			
		RREmission	sMAPE	RREnergy	RRTime	RREmission	sMAPE	RREnergy	RRTime	RREmission	sMAPE	RREnergy	RRTime
PNO	Diff. Head	**2.13**	11.14	3.72	7.14	**3.07**	13.68	0.04	-1.00	**4.80**	12.53	**-0.09**	5.47
	NID	4.01	12.62	8.52	2.17	4.64	15.37	1.07	-0.12	6.17	14.65	0.14	6.11
	SPO+	3.88	11.58	14.01	3.22	4.35	14.71	0.15	0.45	5.93	**12.28**	2.94	5.99
PTO	Slot-Level	3.10	11.16	8.65	1.36	4.05	**13.38**	1.08	**-1.47**	6.23	14.21	0.98	7.19
	Interval-Average	4.11	10.38*	12.78	5.81	5.01	12.45*	0.83	3.44	6.35	13.39*	3.65	6.31
Naive	Repeat-Last	3.75	14.20	**1.42**	0.55	4.52	17.08	-0.06	1.23	7.71	13.09	1.19	15.63
	Recent-Average	3.61	**10.82**	3.79	-12.52	5.30	14.04	0.05	-1.01	13.52	14.78	3.17	**2.04**
	Random	7.69	N/A	16.80	**-17.56**	8.14	N/A	0.56	1.73	15.11	N/A	19.09	31.00
	Worst	86.01	N/A	35.62	5.66	101.92	N/A	**-16.61**	164.77	73.01	N/A	23.73	58.41

TABLE III

RREMISSION OF DEEP LEARNING FORECASTING MODELS USING SLOT-LEVEL AND DIFF. HEAD APPROACHES. LOWER IS BETTER.

Model	Alibaba		Helios		Philly	
	Slot-Level	Diff. Head	Slot-Level	Diff. Head	Slot-Level	Diff. Head
LSTM	8.07	4.59	5.12	3.76	6.54	4.39
TCN	6.84	5.03	6.31	4.98	6.68	7.12
AutoFormer	5.89	5.37	5.13	4.95	7.25	10.73
PatchTST	5.41	6.12	4.64	5.31	6.79	6.16

Table III shows that Diff. Head outperforms the two-stage baseline in most cases, though there are instances where the two-stage approach yields better performance. The extent of improvement varies across forecasters and datasets, but the overall pattern favors our end-to-end training method. This suggests that incorporating forecasts from a model trained end-to-end typically results in lower data center carbon emissions than a model trained by minimizing the prediction loss.

VI. DISCUSSION AND FUTURE WORK

As data center energy demand and emissions continue to rise, workload shifting will become vital for achieving efficient and sustainable data center operations without requiring major power grid reinforcement. This work demonstrates that end-to-end learning for carbon-aware workload scheduling can substantially reduce a hyperscaler's total emissions compared to both conventional scheduling methods that ignore carbon intensity and two-stage predict-then-optimize approaches that decouple forecasting from decision-making. Empirical evaluations using real-world workloads and high-resolution carbon intensity traces highlight the tangible benefits of integrating prediction and optimization within a unified framework.

Limitations and future work. We focus on settings where regional carbon intensity curves frequently cross, so forecast errors can substantially alter job assignments. This assumption is plausible in interconnected grids [18], where regional carbon intensities are correlated and neighboring regions exhibit similar values. When regional intensities are well separated, the performance gains we report may diminish. One promising avenue for future work is to relax the immediate job scheduling requirement and enable temporal shifting, i.e. allowing jobs to be deferred to periods of lower carbon intensity to achieve greater emission reductions. Another direction is to jointly minimize operating cost and emissions subject to latency SLAs, and to characterize the Pareto frontier between sustainability and cost. Additionally, conducting experiments on a high-fidelity data center simulator would enable more accurate modeling of job completion time and energy consumption.

REFERENCES

[1] IEA, "Energy demand from AI," https://www.iea.org/reports/energy-and-ai/energy-demand-from-ai, 2025, accessed: 2025-11-01.

[2] J. Dodge *et al.*, "Measuring the carbon intensity of AI in cloud instances," in *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. ACM, 2022, pp. 1877–1894.

[3] L. Wu *et al.*, "CarbonEdge: Leveraging mesoscale spatial carbon-intensity variations for low carbon edge computing," in *Proceedings of the 34th International Symposium on High-Performance Parallel and Distributed Computing*, ser. HPDC '25. ACM, 2025, pp. 1–13.

[4] P. Wiesner *et al.*, "Let's wait awhile: How temporal workload shifting can reduce carbon emissions in the cloud," in *Proceedings of the 22nd International Middleware Conference*. ACM, 2021, pp. 260–272.

[5] W. A. Hanafy *et al.*, "CarbonScaler: Leveraging cloud workload elasticity for optimizing carbon-efficiency," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 7, no. 3, pp. 1–28, 2023.

[6] A. Souza *et al.*, "CASPER: Carbon-aware scheduling and provisioning for distributed web services," in *Proceedings of the 14th International Green and Sustainable Computing Conference*. ACM, 2023, pp. 67–73.

[7] D. Maji *et al.*, "Bringing carbon awareness to multi-cloud application delivery," in *Proceedings of the 2nd Workshop on Sustainable Computer Systems*, 2023, pp. 1–6.

[8] Electricity Maps, "United States 2021 5-Minute Carbon Intensity Data," https://www.electricitymaps.com, 2025, version January 27, 2025.

[9] Q. Weng *et al.*, "MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, 2022.

[10] A. Zeng *et al.*, "Are transformers effective for time series forecasting?" 2022. [Online]. Available: https://arxiv.org/abs/2205.13504

[11] Q. Hu *et al.*, "Characterization and prediction of deep learning workloads in large-scale GPU datacenters," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '21. ACM, 2021.

[12] M. Jeon *et al.*, "Analysis of large-scale multi-tenant GPU clusters for DNN training workloads," in *Proceedings of the USENIX Annual Technical Conference*. USENIX Association, 2019, pp. 947–960.

[13] A. N. Elmachtoub and P. Grigas, "Smart "predict, then optimize"," 2020. [Online]. Available: https://arxiv.org/abs/1710.08005

[14] S. S. Sahoo *et al.*, "Backpropagation through combinatorial algorithms: Identity with projection works," *arXiv preprint arXiv:2205.15213*, 2022.

[15] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[16] H. Wu *et al.*, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in neural information processing systems*, vol. 34, pp. 22 419–22 430, 2021.

[17] Y. Nie, "A time series is worth 64 words: Long-term forecasting with transformers," *arXiv preprint arXiv:2211.14730*, 2022.

[18] X. Zhang and D. Wang, "A GNN-based day ahead carbon intensity forecasting model for cross-border power grids," in *Proceedings of the 14th ACM International Conference on Future Energy Systems*. ACM, 2023, pp. 361–373.