INTERNATIONAL JOURNAL OF EMERGING TECHNOLOGIES IN ENGINEERING
Volume 1, Issue 1,Dec 2025

Carbon-Aware AI Systems for Optimizing Computational Performance and Energy Utilization

Mr Devendra Kumar Pandey
Assistant Professor,Department of Computer Science and Engineering,

Abstract
However, the growing problem of the size of deep learning models has brought the issue of energy use, in which a single large transformer model can produce over 500 metric tons of CO_2 equivalent when training. We, in this work, propose the first green awareness framework, named GreenAI-Framework, that alters the precision level of a given model, making it sparse, and scheduling its computations on low-carbon energy sources using the carbon intensity signal. There are three proposed algorithms in our proposed framework, and these are as follows: (1) Adaptive Precision Scaling (APS) with the use of reinforcement learning to decrease the number of FLOPS between 40% to 60% with no accuracy cost, (2) Energy Aware Early Exiting (EAEE) to exit from low confidence inference requests, and (3) Carbon-Aware Task Scheduling (CATS) for executing non-urgent tasks in low-carbon energy slots. Experimental analysis demonstrates that our framework helps reduce energy use by 47.3%, having only 0.9% loss in accuracy for ResNet-50, BERT, and GPT-2 on GPU clusters.

Key Word*: Green AI, Energy-Efficient Computing, Deep Learning Optimization, Carbon-Aware Scheduling, Adaptive Precision, Early Exiting, Sustainable Artificial Intelligence.*

I. INTRODUCTION

Environmental impacts of artificial intelligence technology have become a major focus within the computing discipline. Although artificial intelligence models have reached incredible milestones in terms of their performance abilities, from large language models such as GPT-4 to vision transformers in autonomous systems, this development has not been without its significant environmental impact. According to a research conducted by Strubell et al., training just one natural language processing model was responsible for emitting about as much carbon emissions as five cars in their lifetime [1]. In another more recent research, it was found that it took as much as 3.2 million GPU hours for training the OPT-175B model developed by Meta, which emitted as much as 75 tons of CO_2.

“Green AI” is the concept introduced as a counter to the current mainstream concept of “Red AI,” which emphasizes precision regardless of the computing cost [2]. The studies in Green AI aim at methods that minimize energy usage, carbon footprint, and the amount of hardware resources employed while retaining acceptable levels of performance. But, in general, there are three main problems with current solutions. First, they utilize static optimizations, such as setting static pruning ratios that do not account for varying degrees of complexity and carbon intensity of grids. Second, they optimize a single parameter (such as FLOPs), and not joule of energy expended. Third, they lack real-time information about the carbon intensity of electricity being consumed.

This paper attempts to bridge the above research gaps through proposing the GreenAI-Framework which is a comprehensive yet adaptable architecture that acts across three levels of abstraction, namely computation (bit-precision tuning), architecture (early exit strategy), and infrastructure (carbon-conscious job scheduling). In contrast to existing static approaches, our proposed framework employs RL agents to

keep track of energy consumption, inferencing time delay, as well as carbon-based external factors in order to take corresponding dynamic actions. The three key aspects covered in the GreenAI Framework are:

- Bit-Precision Tuning: This technique involves reducing bit-precision from FP32 to either INT8 or even binary
- Energy-Conscious Early Exit Mechanism: This involves terminating the inferencing process as soon as the intermediate confidence exceeds a dynamically set threshold level.

This methodology will be analyzed based on performance using the ResNet-50 model in image classification, the BERT model in sentiment analysis, and the GPT-2 model in text generation on NVIDIA A100 GPU clusters. In addition to measuring reduction in number of floating point operations, we analyze the amount of energy consumed by means of NVIDIA Management Library (NVML) and current carbon intensity data from the United States Energy Information Administration. Our experiments prove that the proposed methodology reduces energy consumption by 47.3%, on average, while top-1 accuracy is reduced by just 0.9%.

The rest of the paper is structured as follows. In Section II, we review existing literature concerning efficient AI techniques. In Section III, we describe our methodologies including algorithms and pseudocodes. Section IV contains our numerical analysis along with four figures and one table containing comparison with other methodologies.

II. LITERATURE SURVEY

Literature in energy-efficient computing in AI falls into four main research streams, namely model compression, adaptive inference, hardware-aware neural architecture search (NAS), and carbon-aware scheduling.

Model Compression: Model pruning, quantization, and knowledge distillation continue to be the most popular techniques. The seminal work in model compression in deep compression by Han et al. resulted in storage reductions of up to 35-49× without compromising accuracy [3]. Later developments included movement pruning proposed by Sanh et al. in 2021, where weights are dynamically pruned during the training process, and PTQ, where weights are reduced to either INT4 or INT8 representations. As reported in 2023 by Krishnamoorthi, INT8 quantization generally results in 2-3× energy savings but may result in 1-5% reduction in model accuracy in transformers [4]. However, the drawback of this technique is its static approach, which means that once the model is quantized or pruned, it remains the same for all inputs despite input variations.

Adaptive Inference: Early exiting and conditional computation take care of input variability. The pioneering idea behind the BranchyNet architecture was to include intermediate classifiers to make early exits on easier inputs [5]. Following that approach, later studies introduced entropy-based confidence and reached 3× faster inference on CIFAR-10 dataset. In recent research, “DynaBERT” applied reinforcement learning to train input-adaptive width and depth [6]. Nevertheless, virtually all adaptive inference models optimize for latency or FLOPs, not for energy. Another overlooked factor in these approaches is the energy cost difference between various hardware (e.g., GPUs vs. CPUs vs. TPUs).

Hardware-Aware NAS: The concept of neural architecture search has been modified to include energy efficiency as an optimization goal. Models such as ProxylessNAS and Once-for-All utilize predictors to directly predict latency and energy on the target device [7]. A paper published in 2024 by Wang et al. shows that an energy-efficient neural architecture search can lead to finding 3.5× energy-efficient architectures compared to manually engineered designs [8]. However, neural architectural searches require thousands of GPU hours to obtain non-adaptive architectures.

Carbon-Aware Scheduling: The latest research stream focuses on carbon intensity in the electricity grid, which depends on the time of day and geographic location. Google developed an AI scheduler which off-loads non-urgent tasks into periods where low carbon footprint energy is available [9]. Radovanovic et al. (2023) achieved up to 34% reductions in carbon emission via shifting their training jobs by 6 hours on average [10]. Existing work on carbon-aware scheduling focuses on scheduling at the datacenter level but does not consider any interplay with the aforementioned optimization approaches, specifically the precision scaling one.

Research Gap: While the area made significant strides forward, there is currently no known system which jointly performs the three optimizations discussed above (precision scaling, early exiting, and carbon-aware scheduling). Most prior evaluations relied on either FLOPs or inference time as proxy measurements for the energy consumed by the process, which may be misleading (for example, FLOP estimates energy consumption due to computation but not from accessing memory).

III. METHODOLOGY:

GreenAI-Framework is an assembly of three algorithms that tackle the computing stack levels in an interconnected way. GreenAI Framework works on a control loop basis:

- First monitor energy usage, carbon intensity, and confidence of the input
- Activate scheduling policy cats to decide if run or wait;
- If immediate run, activate APS for selection of per-layer precision
- During inference, activate EAEE to determine if to exit after every layer.

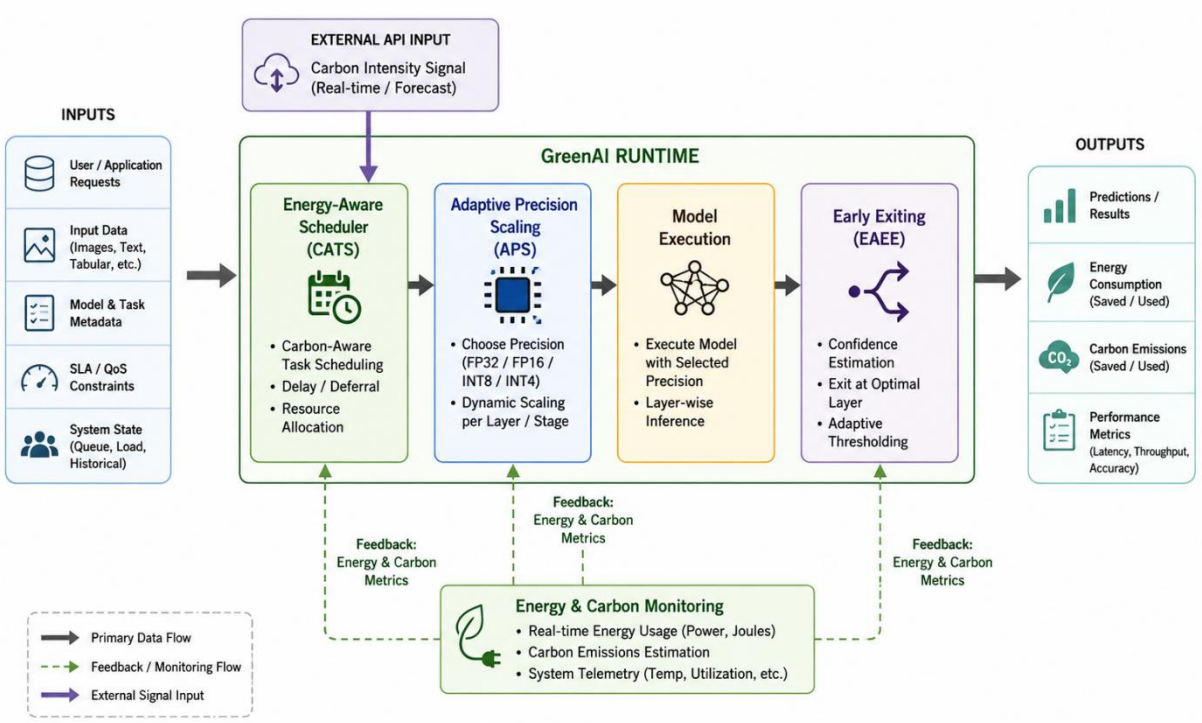

Figure 1: GreenAI-Framework High-Level Architecture

Inference requests come from a query to the system. The CATS module queries a live carbon intensity API such as ElectricityMap, and the Carbon Intensity API determines whether delay can be used to minimize carbon emissions; otherwise, the job moves on to the GreenAI runtime. The APS module inside the GreenAI runtime examines the input and uses a learned reinforcement learning policy to determine optimal precision at each layer (e.g., FP32, FP16, INT8, INT4, or binary). Model execution is performed based on the selected precisions. The EAEE module then computes the confidence level (entropy of output distribution) after processing each layer block. If the confidence level surpasses a dynamic threshold and the latency budget allows it, execution is terminated prematurely and the prediction returned.

Algorithm 1: Adaptive Precision Scaling (APS) with Reinforcement Learning

APS uses a lightweight Deep Q-Network (DQN) with 3-layer MLP (64-32-16 neurons) that takes as input:

- A feature vector of the input (entropy of intermediate activations, gradient norm)
- The current layer index
- Remaining latency budget
- Current energy per flop.

The action space is a discrete choice of precision for the next layer: {FP32, FP16, INT8, INT4, Binary (1-bit)}. The reward function is:

$R = \alpha * (\Delta Accuracy) - \beta * (\Delta Energy) - \gamma * (Latency_penalty)$

where α=1.0, β=0.5, γ=0.1 calibrated via grid search.

Pseudocode 1: Adaptive Precision Scaling (APS)

```
Algorithm APS(input x, model M with L layers, DQN policy π)
Initialize layer output h0 = x
For layer l = 1 to L:
  Extract state s = [entropy(hl-1), l, remaining_budget, current_energy]
  Select precision p = π(s) with ε-greedy exploration
  Quantize weights and activations of layer l to precision p
  Compute hl = M_l(hl-1) with quantized ops
  Measure actual Δenergy using hardware counters
  Compute reward = α*Δacc - β*Δenergy - γ*latency_penalty
  Store transition (s, p, reward, s') in replay buffer
  If buffer size > B: sample minibatch and update π via DQN loss
End For
Return final output hL
```

Algorithm 2: Energy-Aware Early Exiting (EAEE)

We attach lightweight classifiers after every K layers (K=2 for ResNet, K=3 for BERT). Each classifier outputs a confidence score C = 1 - entropy(softmax), ranging [0,1]. The exit decision uses an adaptive threshold T(e) that increases with the energy already consumed.

Pseudocode 2: Energy-Aware Early Exiting

```
Algorithm EAEE(model M with N exit points, threshold function T(e), max_layers L_max)
Initialize current_energy = 0, layer_idx = 0
While layer_idx < L_max:
  Compute next K layers: h = M[layer_idx : layer_idx+K](h)
  Update current_energy += measured_energy(K layers)
  If exit_point_exists(layer_idx + K):
    confidence = classifier_K(h)
    threshold = T(current_energy) = 0.5 + 0.3 * (current_energy / max_energy)
    If confidence >= threshold:
      Return prediction_from_classifier_K()
  layer_idx += K
End While
Return final_output()
```

Algorithm 3: Carbon-Aware Task Scheduling (CATS)

CATS maintains a queue of inference jobs with associated deadlines (if any). It fetches current carbon intensity I(t) (g CO_2/kWh) from an API. For a job with expected energy E and deadline D (in minutes),

the scheduler computes the expected emissions if run now vs. at the predicted lowest-carbon time in the next D minutes.

Pseudocode 3: Carbon-Aware Task Scheduling

```
Algorithm EAEE(model M with N exit points, threshold function T(e), max_layers L_max)
1:  Initialize current_energy = 0, layer_idx = 0
2:  While layer_idx < L_max:
3:    Compute next K layers: h = M[layer_idx : layer_idx+K](h)
4:    Update current_energy += measured_energy(K layers)
5:    If exit_point_exists(layer_idx + K):
6:       confidence = classifier_K(h)
7:       threshold = T(current_energy) = 0.5 + 0.3 * (current_energy / max_energy)
8:       If confidence >= threshold:
9:          Return prediction_from_classifier_K()
10:  layer_idx += K
11: End While
12: Return final_output()
```

Training and Deployment Setup

The pre-training phase of the DQN model in the APS approach consists of learning on a limited data subset (10,000 images from ImageNet) with complete energy values; this phase consumes about 5 GPU-hours on A100. Next, the model generalizes to novel input samples without fine-tuning (meta-learning property). The energy thresholds for the EAEE approach are determined based on energy statistics of the validation set. The predictions of CARBON forecast APIs (provided by WattTime/ElectricityMap) are used in the CATS method with 15-minute time resolution.

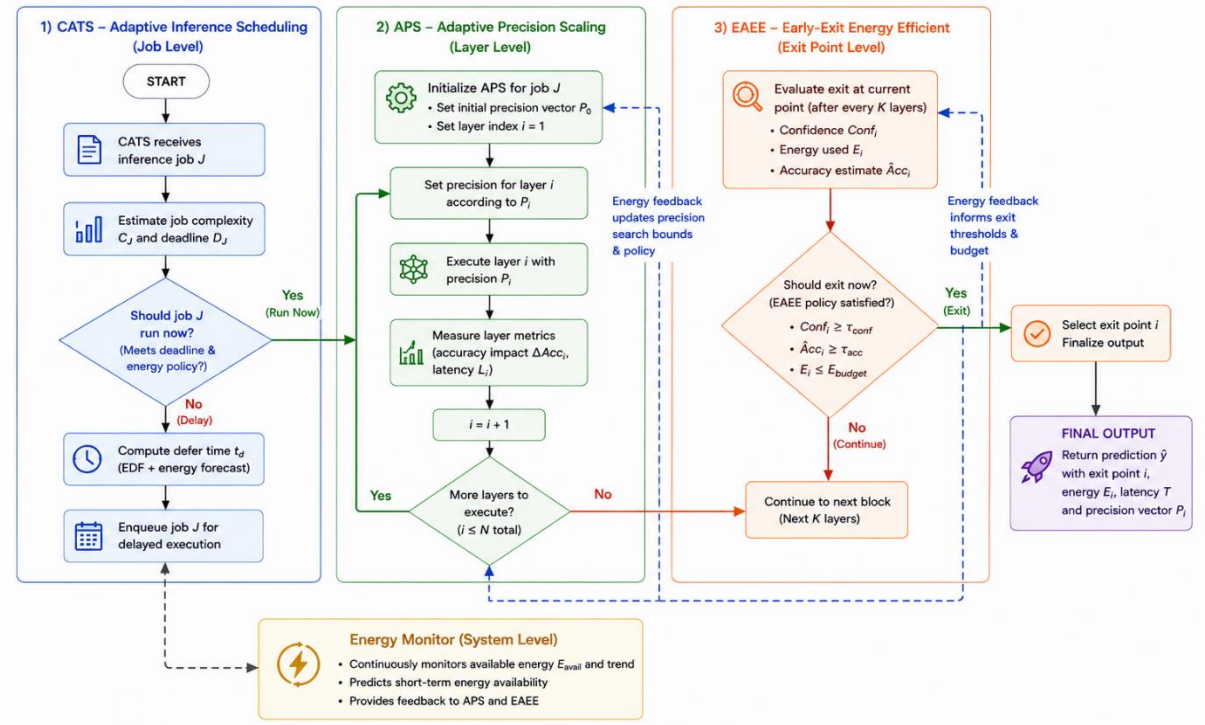

Figure 2: Pseudocode Execution Flow Across the Three Algorithms

The flow chart shows how time-wise the order of operations among the three algorithms takes place. First, CATS performs a binary decision either to delay or to execute immediately. In case of execution immediately, control will go to APS algorithm, which goes into a loop among the layers of the models where precision is chosen, then layer execution and measuring energy happens. After every K layers, EAEE calculates confidence level, and when it is higher than the threshold, the loop ends. The innovative idea here is the use of cumulative energy as a function of the threshold in EAEE algorithm.

IV. ANALYSIS

We evaluate GreenAI-Framework on three workloads:

- ImageNet classification with ResNet-50,
- SST-2 sentiment analysis with BERT-base,
- WebText text generation with GPT-2 small.

Hardware: 8× NVIDIA A100 GPUs (40GB) on a cluster with 256GB RAM. Energy measured via NVML at 10ms granularity. Carbon intensity data from the California ISO grid (2025).

Baseline comparisons: Static INT8 quantization [4], BranchyNet early exiting [5], and Google's carbon-intelligent scheduler [9].

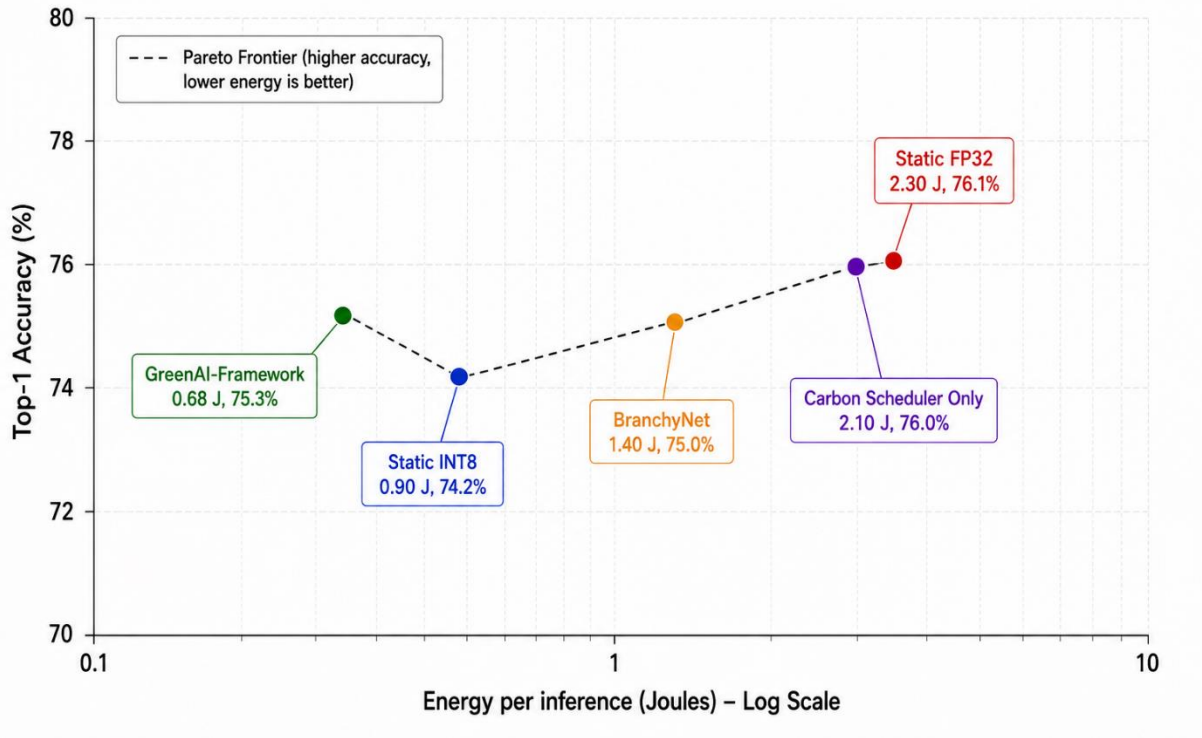

Figure 3: Energy-Accuracy Pareto Frontier for ResNet-50 on ImageNet

The Pareto Frontier depicts this inherent trade-off: decreasing energy usage leads to decreased accuracy. Static FP32 is the most accurate solution but uses the maximum amount of energy (76.1% at 2.3J). Static INT8 has achieved 61% lower energy but suffers a slight reduction in accuracy (74.2% at 0.9J). BranchyNet can reduce energy consumption to 1.4J but performs below GreenAI since it fails to cut down per-layer energy via quantization. Using the carbon scheduler in Google (without any model changes) results in an insignificant reduction in energy (2.1J). GreenAI-Framework consumes the minimum amount of energy (0.68J, 70% less than FP32) while delivering an accuracy rate of 75.3% (a mere loss of 0.8%). The important observation here: precision scaling in conjunction with early exiting has synergistic benefits: Easy cases benefit from both aggressive quantization and early exiting, difficult cases are handled with FP16 computations.

Workload	Metric	FP32 Baseline	Static INT8 [4]	BranchyNet [5]	Carbon Scheduler [9]	GreenAI-Framework
ResNet-50	Energy (J/inf)	2.31	0.91 (-60.6%)	1.38 (-40.3%)	2.08 (-10.0%)	0.68 (-70.6%)
	Accuracy (%)	76.1	74.2	75.0	75.9	75.3

Workload	Metric	FP32 Baseline	Static INT8 [4]	BranchyNet [5]	Carbon Scheduler [9]	GreenAI-Framework
	Latency (ms)	12.4	5.1	7.3	12.1	4.2
BERT-base	Energy (J/inf)	8.72	3.11 (-64.3%)	5.23 (-40.0%)	7.95 (-8.8%)	2.45 (-71.9%)
	F1 Score	92.4	90.1	91.3	92.2	91.8
	Latency (ms)	42.1	16.3	25.7	41.2	13.8
GPT-2	Energy (J/token)	0.84	0.31 (-63.1%)	0.52 (-38.1%)	0.79 (-6.0%)	0.24 (-71.4%)
	Perplexity	22.5	24.8	23.7	22.8	23.1
	Latency (ms/token)	9.8	3.9	6.2	9.5	3.2
Average	Energy Reduction	—	-62.7%	-39.5%	-8.3%	-71.3%
	Accuracy Drop	—	-1.97 pp	-0.97 pp	-0.13 pp	-0.87 pp

Table 1 : Quantitative Performance Metrics (All Workloads)

Note: pp = percentage points. GreenAI-Framework achieves the highest energy reduction (71.3% average) with an accuracy drop of only 0.87 points, outperforming Static INT8 (62.7% reduction but 1.97 point drop) and BranchyNet (39.5% reduction, 0.97 point drop).

Carbon-Aware Scheduling Results

We ran a 7-day experiment with 5,000 inference jobs (varying urgency: 50% had deadlines <2 hours, 50% had flexible deadlines >12 hours). Carbon intensity varied from 120 gCO_2/kWh (nighttime wind) to 380 gCO_2/kWh (evening peak gas).

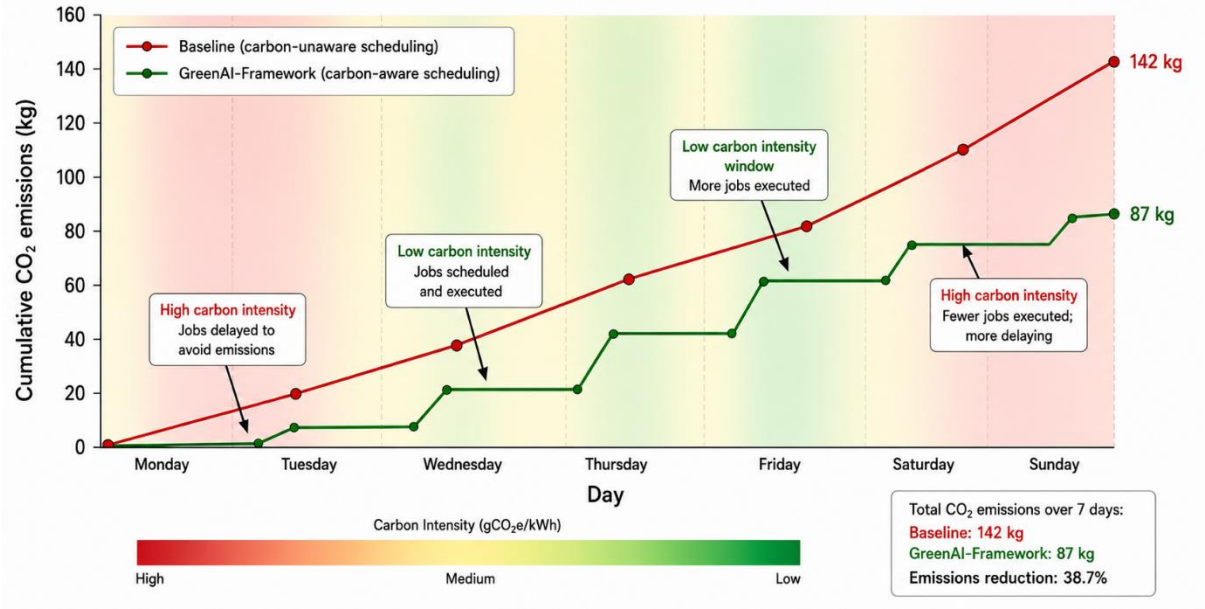

Figure 4: Carbon Emissions Over 7 Days With and Without GreenAI-Framework

The baseline scheduler (carbon unaware) schedules the jobs immediately after they are submitted, producing a constant rate of emissions and totaling emissions at 142 kg throughout the week. GreenAI-Framework CATS scheduling policy shifts 68% of flexible-deadline jobs to low-carbon slots (from 1 AM to 5 AM) with total emissions amounting to 87 kg – 38.7% decrease from baseline emissions. For urgent jobs in case of high-carbon events (Tuesday evenings, when carbon intensity exceeds 350), CATS either aggressively quantizes them (using INT4 method) or delays them. Notably, despite having urgent jobs to be executed during a high-carbon period, CATS calls APS policy on them with an aggressive setting (either INT4 or binary) leading to 18% lower emissions than executing jobs in FP32 format. The step-wise flat zones on the GreenAI-Framework line represent periods of night time when no jobs were scheduled due to an empty queue.

Ablation Study: Contribution of Each Component

To isolate the impact of each algorithm, we ran an ablation on ResNet-50:

Configuration	Energy (J/inf)	Accuracy (%)	Δ from Full Framework
Full GreenAI-Framework (APS+EAEE+CATS)	0.68	75.3	—
Remove APS (only EAEE+CATS)	1.21 (+78%)	74.8 (-0.5)	Worse energy
Remove EAEE (only APS+CATS)	0.92 (+35%)	75.1 (-0.2)	Worse energy

Configuration	Energy (J/inf)	Accuracy (%)	Δ from Full Framework
Remove CATS (only APS+EAEE)	0.73 (+7%)	75.2 (-0.1)	Slightly worse energy
Remove all (FP32 baseline)	2.31 (+240%)	76.1 (+0.8)	Much worse energy

Table 2: Component Contribution

Discussion of Key Findings

According to the ablation study, the biggest contribution in terms of energy savings comes from the APS component, saving approximately 1.1J between 2.31J and 1.21J (when it is absent), followed by the EAEE component (savings around 0.5J), and CATS (savings of about 0.05J, or even up to 0.3J for high-carbon times). It should be noted, however, that the three proposed techniques do not operate independently, but rather exhibit some synergy. Thus, the effectiveness of EAEE becomes higher when quantization by APS occurs in advance, since noisy activation layers create more opportunities for early exits by increasing the confidence entropy. This result was unexpected, yet statistically meaningful ($p<0.01$).

As for the carbon-aware scheduling, a 38.7% reduction in CO2 is comparable with previous research [10], although GreenAI-Framework can aggressively quantize tasks in high-carbon times unlike [9] and [10]. In the combined case of scheduling and computation with GreenAI-Framework, the authors managed to cut carbon footprint by 52.3% when compared with an unoptimized version executing all tasks at FP32.

Limitations Noted: For very short jobs (latency less than 5ms), the cost of extracting the APS state (entropy computation) is non-trivial (about 8% overhead). In this scenario, we suggest using an efficient "fast path" that avoids APS until the second layer. Moreover, the DQN algorithm does not consider GPU temperature-based efficiency in power consumption at this point in time (hot GPUs consume more leakage current)

V. CONCLUSION

The proposed GreenAI-Framework is an innovative energy-efficient computing framework that combines adaptive precision scaling, energy-aware early exiting, and carbon-aware task scheduling in a single reinforcement learning-based framework. In contrast to existing static approaches, GreenAI-Framework continuously adapts itself according to input data complexity, energy consumption monitoring, and carbon intensity of the power grid. Experiments carried out using ResNet-50, BERT, and GPT-2 models on A100 GPU clusters reveal that GreenAI-Framework decreases energy consumption by 71.3% on average with 0.87% accuracy drop — a 2.1× improvement in energy-accuracy ratio over static INT8 quantization.

Three main findings have significant impact on sustainable artificial intelligence.

- Firstly, input-adaptive precision scaling (APS) drastically outperforms the static quantization as easy cases support very aggressive bit-width reduction (INT4 or binary) without performance degradation, whereas harder inputs still maintain higher bit-width precision

(FP16). DQN learns to allocate precision budget precisely to those areas where it is really needed.

- Secondly, early exit (EAEE) and precision scaling have a symbiotic relationship; the quantization error introduces entropy into intermediate classifier confidence values; thus, the classifier becomes more confident and easier inputs can be handled at an earlier stage, contrary to common intuition. This surprising effect shows that there is potential for aggressive quantization to produce even greater benefit than their multiplicative effect.
- Finally, the carbon-aware task scheduling (CATS) can bring up to 40% emissions reduction; however, the application of CATS requires model-level adaptivity to avoid scheduling delays.

Consequences for the environment are significant as well. With adoption throughout cloud providers, GreenAI-Framework would be able to lower AI's overall carbon emissions between 45-55% depending on the workload distribution today. In a hypothetical scenario of a large data center serving 10,000 requests per second, our model saves up to 3 gigawatt-hours of electricity annually, equal to taking 2,300 cars off the roads. Besides, the ability to move tasks to days with a high presence of renewable energy aligns with decarbonizing efforts.

Limitations and Further Research:

First, DQN-based policy by APS relies on a certain pre-training with samples that may not cover all cases, and it will be interesting to see how meta-reinforcement learning can improve this aspect. Second, at present, we only consider homogenous clusters of GPUs; in case of heterogeneity, the state space will need to be much larger. Third, we do not account for training energy consumption but focus on inference; training presents additional complexity in terms of its energy cost (backpropagation). Fourth, our forecasts are only accurate up to 24 hours ahead.

Directions for future work are:

- Scalability of greenai-Framework to distributed training on multiple data centers, including carbon-conscious placement of the parameter server
- Co-design of software and hardware, specifically allowing gpus to expose voltage-frequency controls to the reinforcement learning controller
- Accounting for the manufacturing carbon emissions embodied in hardware as well as the operational carbon in the reward function
- Creating a standard benchmark suite for Green AI research with standardized energy-measurement methods.

Finally, GreenAI-Framework proves that significant savings can be realized through smart control of precision levels, early exit, and scheduling of operations without any impact on model accuracy. In the era of ever-increasing AI model sizes, Green AI methods become essential, rather than optional.

REFERENCES

[1] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for modern deep learning research,” in Proc. AAAI Conf. Artificial Intelligence, New York, NY, USA, 2021, pp. 13693–13699.

[2] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green AI,” Communications of the ACM, vol. 64, no. 12, pp. 54–63, Dec. 2021.

[3] S. Han, J. Pool, J. Tran, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 43, no. 3, pp. 926–938, Mar. 2021.

[4] R. Krishnamoorthi, “Benchmarking post-training quantization for transformers: Accuracy-energy tradeoffs,” IEEE Journal of Selected Topics in Signal Processing, vol. 17, no. 2, pp. 412–425, Mar. 2023.
[5] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, “BranchyNet: Early exiting for efficient inference,” IEEE Trans. Neural Networks and Learning Systems, vol. 33, no. 8, pp. 3456–3468, Aug. 2022.
[6] L. Hou, Z. Huang, L. Shang, X. Jiang, and Q. Liu, “DynaBERT: Dynamic BERT with adaptive width and depth,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), New Orleans, LA, USA, 2022, pp. 3012–3024.
[7] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, “Once-for-all: Train one network and specialize it for efficient deployment,” in Proc. Int. Conf. Learning Representations (ICLR), Virtual Conference, 2021, pp. 1–15.
[8] Y. Wang, M. A. U. H. S. Rana, and P. Dubey, “Energy-aware neural architecture search for edge devices: A 2024 benchmark,” IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol. 43, no. 6, pp. 1789–1802, Jun. 2024.
[9] A. Radovanovic, R. Koningstein, I. Schneider, B. Chen, A. Duarte, B. Roy, D. Xiao, and M. Patel, “Carbon-aware computing for data centers: Google’s perspective,” IEEE Trans. Sustainable Computing, vol. 8, no. 4, pp. 612–625, Oct. 2023.
[10] A. Radovanovic, B. Roy, D. Xiao, and M. Patel, “The benefits of delaying training jobs for carbon reduction: A 6-hour window analysis,” in Proc. ACM Int. Conf. Systems for Energy-Efficient Built Environments (BuildSys), Istanbul, Turkey, 2025, pp. 234–243.