SustainML

Application Aware, Life-Cycle Oriented Model-Hardware Co-Design Framework for Sustainable, Energy Efficient ML Systems

Carbon footprint based model optimization tool

Deliverable D3.1

WP3 - Energy Consumption Optimized ML Toolkit and Methods

This project has received funding from the European Union's Horizon Europe research and innovation programme (HORIZON-CL4-2021-HUMAN-01) under grant agreement No 101070408

eProsima The Middleware Experts

DFKI

Inria

upmem

Technische Universität Kaiserslautern

IBM

Project

Title:	SustainML: Application Aware, Life-Cycle Oriented Model-Hardware Co-Design Framework for Sustainable, Energy Efficient ML Systems
Acronym:	SustainML
Coordinator:	eProsima
Grant agreement ID:	101070408
Call:	HORIZON-CL4-2021-HUMAN-01
Program:	Horizon Europe
Start:	01 October 2022
Duration:	36 months
Website:	https://sustainml.eu
E-mail:	sustainml@eprosima.com
Consortium:	**eProsima (EPROS)**, Spain **DFKI**, Germany **TU Kaiserslautern (TUK)**, Germany **University of Copenhagen (KU)**, Denmark **National Institute for Research in Digital Science and Technology (INRIA)**, France **IBM Research GmbH**, Switzerland **UPMEM**, France

Deliverable

Number:	**D3.1**
Title:	**Carbon footprint based model optimization tool**
Month:	6
Work Package:	WP3 - Energy Consumption Optimized ML Toolkit and Methods
Work Package leader:	KU
Deliverable leader:	KU
Deliverable type:	R,DEM
Dissemination level:	PU
Date of submission:	2024-05-31
Version:	v2.0
Status:	Finished

Version history

Version	Date	Responsible	Author/Reviewer	Comments
v2.0	31-05-2024	Raghavendra Selvan	Pedram Bakhtiarifard, Raghavendra Selvan	Revised with technical review comments
v1.0	31-03-2023	Raghavendra Selvan	Pedram Bakhtiarifard, Raghavendra Selvan	Initial submission

Executive summary

There are many aspects to consider when assessing the cost of developing artificial intelligence (AI) models. These can be of monetary nature such as infrastructure costs and access to compute devices, but also expertise and know-how within specific domains, in order to best utilize developer time. Primarily, development costs are encountered at different stages of the development life cycle: dataset curation, model selection, model training, and model deployment. Nevertheless, a common resource, or cost, for all stages is energy consumption. In the last decade, we have seen the energy consumption of selecting, training and deploying models grow exponentially, and it has become apparent that the bottleneck for scaling AI is energy consumption more than it is raw compute. Therefore, our goal in this work is to support the design of energy-efficient models that require less compute to train and thereby prioritizes environmental sustainability, targeting model selection and training.

Neural Architecture Search (NAS) strategies, which explore model architectures based on training and evaluation metrics, have demonstrated their ability to reveal novel designs with state-of-the-art performance. While promising, NAS comes with computational and energy-intensive demands, leading to significant environmental concerns due to the carbon footprint incurred due to energy consumption. Hence, there is an imperative to address the balance between performance and resource efficiency in NAS. Today's NAS algorithms and benchmarks focus mainly on standard performance measures like accuracy, and efficiency w.r.t. compute time. However, the ladder does not encapsulate the underlying dynamics of model design and hardware utilization and compute time can become obscured in scenarios like federated learning and parallel paradigms.

We advocate for including energy efficiency as an additional performance criterion in NAS. To this end, we introduce a tabular benchmark encompassing data on energy consumption for varied architectures. The benchmark, designated as EC-NAS, has been made available in an open-source format to advance research in energy-conscious NAS. EC-NAS incorporates a surrogate model to predict energy consumption, aiding in diminishing the energy expenditure of the dataset creation. Our findings emphasize the potential of EC-NAS by leveraging multi-objective optimization algorithms, revealing a balance between energy usage and accuracy. This suggests the feasibility of identifying energy-lean architectures with little or no compromise in performance.

This report corresponds to *Deliverable D3.1 - Carbon footprint based model optimization tool* of the SustainML project. This deliverable covers an overview of the growing energy consumption problems in deep learning techniques and the importance of energy-efficient models for real-world applications. A summary of existing NAS methods, tabular benchmarks, and the current focus on performance as the primary objective for model development, and why energy efficiency and carbon footprint are crucial performance/efficiency criteria to consider. Moreover, an important concern when assessing model sustainability regarding energy costs is the ability to measure these costs effectively and accurately across different compute infrastructures. The tool we developed and employed in this work, Carbontracker, is also discussed in terms of its ability to educate AI practitioners about their use of computing resources and the potential impacts and trade-offs among different development practices.

The key contribution in this deliverable is based on [1] which was presented at the International Conference on Acoustics, Speech and Signal Processing (ICASSP-2024).

Contents

1 Introduction

NAS strategies explore predefined search spaces for potential model architectures, determining fitness based on validation/test set performance to discover the best-performing architecture [2]. Despite success in finding novel designs [3, 4, 5, 6], NAS's high computational cost, energy consumption, and significant carbon footprint are drawbacks [7, 8, 9, 10]. Tabular benchmarks and surrogate models have improved NAS evaluation efficiency [11, 12, 13, 14, 15, 5, 16], but the focus remains on performance over efficiency.

Building upon the foundational NAS-Bench-101 [15], we introduce our benchmark, EC-NAS [1], to emphasize the imperative of energy efficiency in NAS. Our adaptation of this dataset, initially computed using an exorbitant 100 TPU years equivalent of compute time, serves our broader goal of steering NAS methodologies, and model development in general, towards energy consumption awareness.

This deliverable highlights the use of energy consumption as an additional criterion in tabular NAS benchmarks to discover energy-efficient models for practical deployment and sustainability with the EC-NAS benchmark [1]. Results show good promise in revealing computationally efficient models that balance energy consumption and performance with minimal performance loss and smaller carbon footprints.

2 Architecture Space

Central to EC-NAS are architectures tailored for CIFAR-10 image classification [17]. We introduce additional objectives for emphasizing the significance of hardware-specific efficiency trends in deep learning models. The architectural space is confined to the topological space of cells, with each cell being a configurable feedforward network. In terms of cell encoding, these individual cells are represented as directed acyclic graphs (DAGs). Each DAG, $G(V, M)$, has $N = |V|$ vertices (or nodes) and edges described in a binary adjacency matrix $M \in \{0, 1\}^{N \times N}$. The set of operations (labels) that each node can realise is given by $\mathcal{L}' = \{\texttt{input}, \texttt{output}\} \cup \mathcal{L}$, where $\mathcal{L} = \{\texttt{3x3conv}, \texttt{1x1conv}, \texttt{3x3maxpool}\}$. Two of the N nodes are always fixed as `input` and `output` to the network. The remaining $N-2$ nodes can take up one of the labels in $\mathcal{L}$. The connections between nodes of the DAG are encoded in the upper-triangular adjacency matrix with no self-connections (zero main diagonal entries). For a given architecture, $\mathcal{A}$, every entry $\alpha_{i,j} \in M_{\mathcal{A}}$ denotes an edge, from node i to node j with operations $i, j \in \mathcal{L}$ and its labelled adjacency matrix, $L_{\mathcal{A}} \in M_{\mathcal{A}} \times \mathcal{L}'$. In aggregate, we consider DAGs where $|V| \leq 7$, encapsulating 423k unique architectures. As architectures are are examined with varying training budgets of 4, 12, 36, and 108 epochs, to explore potential trade-offs between performance and resource expenditure, EC-NAS contains $\approx$ 1.6M data points.

We measure the energy consumption of training these architectures on the CIFAR-10 dataset, follow- ing the protocols to NAS-Bench-101. The in-house SLURM cluster, powered by an NVIDIA Quadro RTX 6000 GPU and two Intel CPUs, provides a representative environment for model development. The vast architecture space, however, introduces chal- lenges in the direct energy estimation. Our remedy to this is a surrogate model approach, wherein we derived insights to guide a multi-layer perceptron (MLP) model by training using a representative subset of architectures. This surrogate model adeptly predicts energy consumption patterns, bridging computational demand and energy efficiency.

2.1 Surrogate Energy Predictor

The MLP-based surrogate energy model takes the graph-encoded architecture and the number of parameters as input, predicting energy consumption for a given number of epochs. This surrogate model

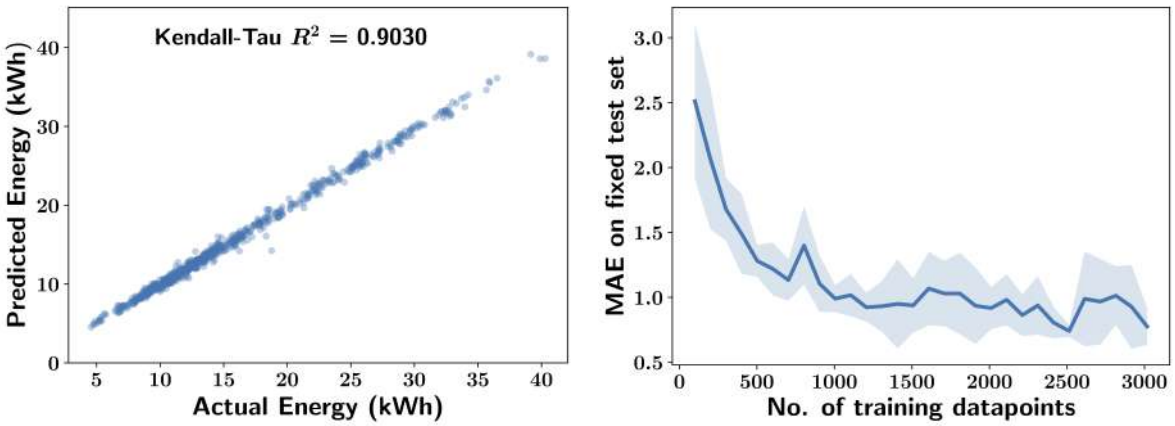

Figure 1: Scatter plot depicting the Kendall-Tau correlation co- efficient between predicted and actual energy consumption (left) and the influence of training data size on test accuracy (right). Error bars are based on 10 random initializations.

is similar to current surrogate NAS methods that have efficiently facilitated NAS evaluations on larger search spaces and provided more accurate performance estimates than tabular benchmarks [14]. Surrogate models are used as one-time compute methods for cost-effective evaluation of NAS methods within extensive search spaces and do not apply to other search spaces.

The MLP-based surrogate model used for predicting the training energy consumption of the 7V space, E, is given as: $f_\theta(\cdot) : \mathbf{x} \in \mathbb{R}^F \rightarrow E \in \mathbb{R}$, where θ are the trainable parameters and $\mathbf{x}$ consists of the F features obtained from architecture specifications. We populate $\mathbf{x}$ with the upper triangular entries of the adjacency matrix, operations mapped to categorical variables and the total number of parameters. For the 7V space, this results in $\mathbf{x} \in \mathbb{R}^{36}$.

The surrogate energy model is implemented as a simple four-layered MLP with gelu activation functions and trained using actual energy measurements from 4310 randomly sampled architectures in the 7V space. The model is implemented in Pytorch and trained on a single NVIDIA RTX 3060 GPU. The training, validation, and test split has a ratio of [0.7,0.1,0.2], resulting in [3020,430,860] data points, respectively. Using the Adam optimiser, the MLP is trained for 200 epochs with an initial learning rate of 5×10^{-3} to minimise the L1-norm loss function between the predicted and actual energy measurements.

The resulting surrogate dataset closely approximates the actual training energy costs, with a Kendall-Tau correlation of 0.9030 between actual and predicted energy measurements on the test set (left) Figure 1. A high correlation is expected, considering the search space exhibits a high degree of locality, especially for smaller models. The mean absolute error of predicted and actual energy measurements plateaus when trained with about 3000 architectures (right), justifying its use for predicting the remaining space.

2.2 Dataset Analysis and Hardware Consistency

Understanding architectural characteristics and the trade-offs they introduce is crucial. This involves studying operations, their impacts on efficiency and performance, as well as the overarching influence of hardware on energy costs. Training time and energy consumption trends naturally increase with model size. However, gains in performance tend to plateau for models characterized by larger DAGs. Interestingly, while parameter variation across model sizes remains minimal, training time and energy consumption show more significant variability for more extensive models. These findings highlight the multifaceted factors affecting performance and efficiency.

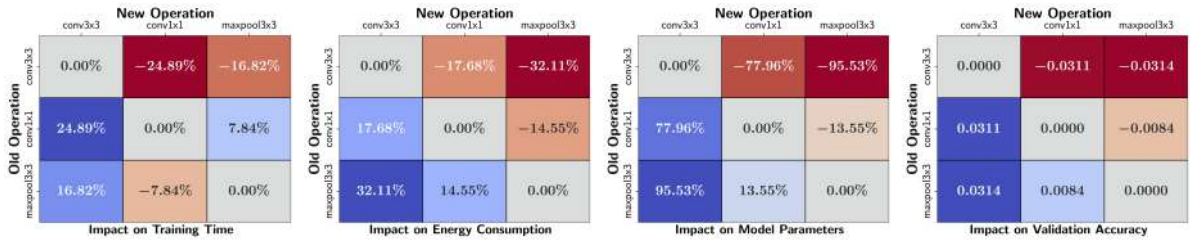

Figure 2: Aggregated impact of swapping one operator for another on energy consumption, training time, validation accuracy, and parameter count. The figure illustrates how changing a single operator can affect the different aspects of model performance, emphasizing the importance of selecting the appropriate operators to balance energy efficiency and performance.

Different operations can also have a profound impact on performance. For instance, specific operation replacements significantly boost validation accuracy while increasing energy consumption without increasing training time. This complex relationship between training time, energy consumption and performance underscore the importance of a comprehensive approach in NAS. The impact of swapping one operation for another on various metrics, including energy consumption, training time, validation accuracy, and parameter count, is captured in Figure 2.

Furthermore, we probed the energy consumption patterns of models characterized by DAGs with $|V| \leq 4$, spanning various GPUs. This exploration, depicted in Figure 3, confirms the flexibility of the benchmark across different hardware environments. This adaptability paves the way for advanced NAS strategies, notably for multi-objective optimization (MOO). It signifies a paradigm shift towards a balanced pursuit of performance and energy efficiency, echoing the call for sustainable computing.

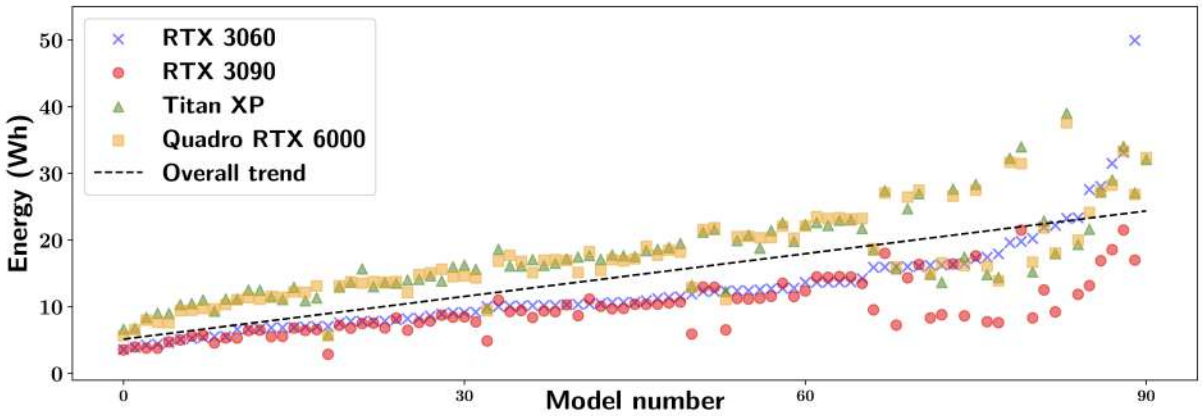

Figure 3: Energy consumption of models with DAGs where $|V| \leq 4$ on different GPUs. Models are organized by their average energy consumption for clarity.

3 Leveraging `EC-NAS` in NAS Strategies

Tabular benchmarks like `EC-NAS` offer insights into energy consumption alongside traditional performance measures, facilitating the exploration of energy-efficient architectures using multi-objective optimization (MOO) whilst emphasizing the rising need for sustainable computing. In the context of NAS, MOO has emerged as an instrumental approach for handling potentially conflicting objectives. We utilize the `EC-NAS` benchmark to apply diverse MOO algorithms, encompassing our own simple evolutionary MOO

Method	Arch.	$T(s)\downarrow$	$P_v\uparrow$	E(kWh)$\downarrow$	$\lvert\theta\rvert$(M)$\downarrow$
SH-EMOA	$\mathcal{A}_{\mathbf{r}_0}$	15.22 ±2.98	0.52 ±0.02	0.01 ±0.00	5.98 ±0.10
	$\mathcal{A}_{\mathbf{r}_1}$	1034.35 ±358.15	0.91 ±0.03	0.27 ±0.11	6.55 ±0.44
	$\mathcal{A}_{\mathbf{r}_k}$	226.28 ±114.03	0.85 ±0.04	0.04 ±0.03	6.27 ±0.43
RS	$\mathcal{A}_{\mathbf{r}_0}$	14.23 ±0.00	0.52 ±0.00	0.01 ±0.00	5.95 ±0.00
	$\mathcal{A}_{\mathbf{r}_1}$	1649.11 ±342.02	0.94 ±0.00	0.41 ±0.09	7.05 ±0.18
	$\mathcal{A}_{\mathbf{r}_k}$	310.93 ±56.50	0.89 ±0.03	0.07 ±0.03	6.51 ±0.38
MSE-HVI	$\mathcal{A}_{\mathbf{r}_0}$	14.23 ±0.00	0.52 ±0.00	0.01 ±0.00	5.95 ±0.00
	$\mathcal{A}_{\mathbf{r}_1}$	1112.13 ±642.49	0.92 ±0.02	0.25 ±0.14	6.80 ±0.33
	$\mathcal{A}_{\mathbf{r}_k}$	191.09 ±93.13	0.83 ±0.03	0.02 ±0.02	6.01 ±0.18
SEMOA	$\mathcal{A}_{\mathbf{r}_0}$	14.23 ±0.00	0.52 ±0.00	0.01 ±0.00	5.95 ±0.00
	$\mathcal{A}_{\mathbf{r}_1}$	2555.95 ±202.42	0.94 ±0.00	0.62 ±0.08	7.26 ±0.15
	$\mathcal{A}_{\mathbf{r}_k}$	306.9 ±41.86	0.92 ±0.01	0.07 ±0.01	6.43 ±0.06

Table 1: Average performance and resource consumption for models. Architectures $\mathcal{A}_{\mathbf{r}_0}$, $\mathcal{A}_{\mathbf{r}_1}$, and $\mathcal{A}_{\mathbf{r}_k}$ correspond to the two extrema and the knee point, respectively.

algorithm (SEMOA) based on [18] and other prominent algorithms such as SH-EMOA and MS-EHVI from [19]. These methodologies are assessed against the conventional random rearch (RS) technique.

Our exploration within `EC-NAS` span both single-objective optimization (SOO) and MOO. We execute algorithms across various training epoch budgets $e \in \{4, 12, 36, 108\}$ over 100 evolutions with a population size of 10. For SOO, 1000 evolutions were designated to equate the discovery potential. Results, averaged over 10 trials, followed the methodology of [19].

For MOO, validation accuracy P_v and the training energy cost, E (in kWh), were chosen as the dual objectives and, for SOO, simply the performance metric. Given its indifference to parallel computing, energy consumption was chosen over training time. Inverse objectives were used for optimization in maximization tasks (e.g., $1 - P_v$). Balancing energy efficiency with performance presents a layered challenge in NAS. Figure 4 elucidates the architectural intricacies and the prowess of various MOO algorithms in identifying energy-conservative neural architectures.

Figure 4 (left) evaluates the architecture discovery efficacy of MOO algorithms, presenting the median solutions achieved over multiple runs. SEMOA, in particular, showcases an even distribution of models attributed to its ability to exploit model locality. In contrast, SH-EMOA and MSE-HVI display a more substantial variation, highlighting the robust search space exploration of SEMOA.

The Pareto front, as depicted in Figure 4 (center), highlights extrema (r_0, r_1) and the knee point (r_k), which represents an optimal trade-off between objectives. The extrema prioritize energy efficiency or validation accuracy, while the knee point achieves a balanced feature distribution. MOO algorithms' capability to navigate the NAS space effectively is evident in their identification of architectures that balance competing objectives.

4 Energy Consumption Awareness

NAS for efficient architectures has primarily focused on optimising run-time or floating point operations (FPOs) [20]. However, FPOs may not fully represent model efficiency [21, 22, 23]. Recently, energy consumption-optimised hyperparameter selection was studied outside NAS settings for large language models [24]. Energy consumption during model training encompasses aspects not covered by standard

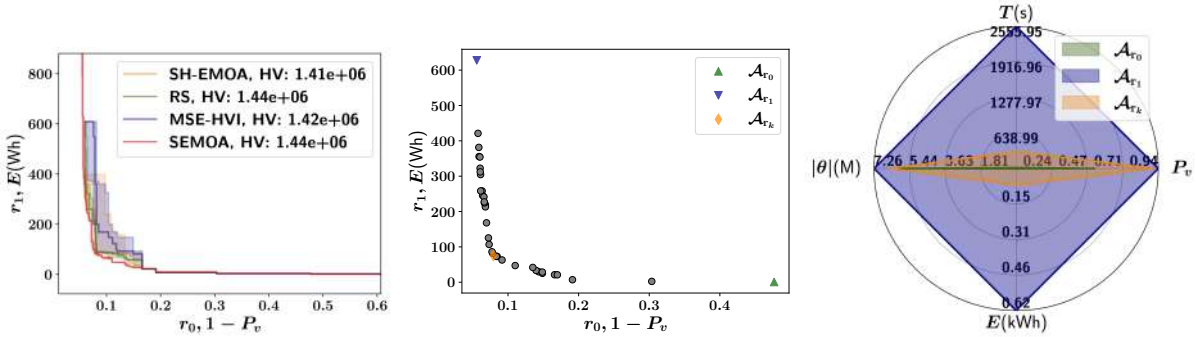

Figure 4: (Left) The attainment curve showing median solutions for 10 random initializations on the surrogate 7V space from `EC-NAS` dataset. (Center) A representation of the Pareto front for one run of SEMOA. (Right) Summary of metrics for the extrema and knee point architectures for one SEMOA run.

resource constraints like FPOs, computational time, and the number of parameters. For instance, given the variability in computational time, owing to diverse factors like parallel infrastructure, this metric can occasionally be misleading. Energy consumption, in contrast, lends itself as a more con- sistent and comprehensive measure, factoring in software and hardware variations, making it a suitable additional objective for NAS.

Roughly 75% of total energy costs when training neural networks come from hardware accelerators like GPUs, and TPUs [9, 25]. The rest is mainly due to CPUs and DRAM, with supporting infrastructure accounted for by power usage effectiveness (PUE). Open-source tools like experiment-impact-tracker [21], Carbontracker [9], and CodeCarbon [26] help track energy consumption. In EC-NAS, the energy consumption is estimated with Carbontracker [9], monitoring GPUs, CPUs, and DRAM to determine total energy costs, E (kWh), carbon footprint ($kgCO_2eq$), and total computation time, T(s) summarised in Table 2.

Metrics	Unit of measurement	Notation
Model parameters	Million (M)	$\|\theta\|$
Test/Train/Eval. time	Seconds (s)	$T(s)$
Test/Train/Val. Acc.	$\mathbb{R} \in [0; 1]$	P_v
Energy consumption	Kilowatt-hour (kWh)	E(kWh)
Power consumption	Joule (J), Watt (W)	E(J), E(W)
Carbon footprint	$kgCO_2eq$	–
Carbon intensity	g/kWh	–

Table 2: Metrics reported in `EC-NAS-Bench`.

5 Discussion

5.1 Carbontracker

CarbonTracker [9] is a Python tool designed to estimate and track the carbon emissions associated with the usage of computational resources, particularly in machine learning and other data-intensive applications. It helps users become more aware of the environmental impact of their computational workloads and make more sustainable choices. An overview of its functionalities are described below:

- **Emission Estimation**: CarbonTracker estimates the carbon emissions produced by running a specific computational job. This estimation is based on the energy consumption of the hardware used (such as CPUs and GPUs) and the carbon intensity of the electricity grid where the hardware is located.
- **Usage Monitoring**: The tool can monitor the usage of computational resources in real-time, allowing users to track the energy consumption of their jobs as they run.
- **Geographical Awareness**: CarbonTracker takes into account the geographical location of the data center or computing facility to provide more accurate carbon emission estimates. This is important because the carbon intensity of electricity can vary significantly between different regions.
- **Reporting**: After a job completes, CarbonTracker generates detailed reports that include the total energy consumed, the carbon emissions produced, and other relevant metrics. These reports can help users understand the environmental impact of their work and identify opportunities for improvement.
- **Optimization Suggestions**: By analyzing the data collected, CarbonTracker can provide suggestions on how to optimize resource usage and reduce carbon emissions. This might include recommendations on more efficient coding practices, alternative algorithms, or more sustainable computing resources.
- **Integration with Workflows**: CarbonTracker is designed to be easily integrated into existing machine learning workflows. It can be used with popular machine learning frameworks like TensorFlow and PyTorch, allowing users to seamlessly incorporate carbon tracking into their projects.

In addition, Carbontracker can also be used in non-intrusive mode that can be useful to integrate with non-GPU workflows.

```
carbontracker your_script
```

5.2 Measurements from Carbontracker

Our measurements account for the energy consumption of Graphics Processing Units (GPUs), Central Processing Units (CPUs), and Dynamic Random Access Memory (DRAM), with the CPU energy usage inclusive of DRAM power consumption. Energy usage data is collected and logged at 10-second intervals, and this information is averaged over the duration of model training. The total energy consumed is then calculated and reported in kilowatt-hours (kWh), where $1\text{kWh} = 3.6 \cdot 10^6$Joules (J). In addition, we assess the emission of greenhouse gases (GHG) in terms of carbon dioxide equivalents (CO_2eq), calculated by applying the carbon intensity metric, which denotes the CO_2eq emitted per kWh of electricity generated. This carbon intensity data is updated every 15 minutes during model training from a designated provider.

Space	Red. GPU days	Red. kWh	Red. kgCO_2eq
4V	3.758	48.931	6.327
5V	121.109	1970.495	252.571
7V	14037.058	259840.907	–

Table 3: Estimated reduction in actual resource costs when creating `EC-NAS` dataset for the 4V and 5V using linear scaling and 7V space using the surrogate model.

However, considering only the direct energy consumption of these components does not fully capture the carbon footprint of model training, as it overlooks the energy consumption of auxiliary infrastructure,

such as data centers. To address this, we refine our estimations of energy usage and carbon footprint by incorporating the 2020 global average Power Usage Effectiveness (PUE) of data centers, which stands at 1.59, as reported in [27].

5.3 Surrogate model adaptability

Our surrogate energy model is promising in predicting energy consumption within our current search space. We have also adapted the surrogate model to the OFA search space, achieving comparable results in terms of energy consumption prediction. This suggests the potential for the surrogate model to be generalized and applied to other search spaces, broadening its applicability and usefulness in future research. Estimates for reduction in compute costs for the EC-NAS benchmark datasets are presented in Table 3.

While a comprehensive investigation of the surrogate model's performance in different search spaces is beyond the scope of this work, it is worth noting that the model could potentially serve as a valuable tool for researchers seeking to optimize energy consumption and other efficiency metrics across various architectural search spaces. Further studies focusing on the adaptability and performance of surrogate models in diverse search spaces will undoubtedly contribute to developing more efficient and environmentally sustainable AI models.

Hardware accelerators have become increasingly efficient and widely adopted for edge computing and similar applications. These specialized devices offer significant performance improvements and energy efficiency, allowing faster processing and lower power consumption than traditional computing platforms. However, deriving general development principles and design directions from these accelerators can be challenging due to their highly specialized nature. Moreover, measuring energy efficiency on such devices tends to be hardware-specific, with results that may need to be more easily transferable or applicable to other platforms. Despite these challenges, we acknowledge the importance and necessity of using hardware accelerators for specific applications and recognize the value of development further to improve energy efficiency and performance on these specialized devices.

5.4 Insights from ECNAS experiments

While we recognize the value of conducting NAS experiments on multiple datasets, the current computational and environmental constraints limit our ability to do so comprehensively. We believe that the insights gained from our existing experiments on CIFAR10 are relevant to the use-case of image classification in WP6, and indicative of our method's potential. Further justification is provided below:

1. **Resource Limitations:** Conducting NAS experiments across multiple datasets is highly resource-intensive. Each NAS run involves training numerous candidate architectures, which requires substantial computational resources and time. Given our current computational infrastructure, it is impractical to perform these extensive experiments across multiple datasets within a reasonable timeframe.
2. **Environmental Considerations:** The energy consumption associated with large-scale NAS experiments has significant environmental impacts. Running multiple extensive experiments would exponentially increase our carbon footprint, which we are striving to minimize with our tabular benchmarks for NAS.
3. **Transferability and Generalization:** Our single-dataset experiments have already provided valuable insights into the efficiency and effectiveness of the proposed ECNAS method. The observed architectural patterns and hyperparameter sensitivities indicate that the discovered architectures have a high potential for transferability and generalization to other datasets.

4. **Benchmark Comparisons:** Our results have been benchmarked against existing state-of-the-art methods on the chosen dataset, demonstrating competitive performance. These benchmarks provide a strong indication of our method's robustness and effectiveness.

We acknowledge the importance of comprehensive multi-dataset evaluations and plan to explore this in future work.

6 Conclusion

Our work presented EC-NAS, a novel neural architecture search (NAS) benchmark that incorporates energy consumption awareness, expanding upon the traditional focus on performance metrics. Combining this additional objective with multi-objective optimisation (MOO) strategies allows energy-efficient architectures to be identified while maintaining competitive performance. Adopting an energy-aware approach in NAS is crucial for mitigating the environmental impact of training large-scale neural networks, as hardware accelerators contribute significantly to energy consumption. EC-NAS enables researchers to explore the trade-offs between performance, energy consumption, and other standard metrics, supporting informed decision-making in architecture selection and ultimately promoting the development of more sustainable AI systems.

References

[1] Pedram Bakhtiarifard, Christian Igel, and Raghavendra Selvan. "EC-NAS: Energy Consumption Aware Tabular Benchmarks for Neural Architecture Search". In: *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2024, pp. 5660–5664. DOI: 10.1109/ICASSP48485.2024.10448303.

[2] Pengzhen Ren et al. *A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions*. 2020. DOI: 10.48550/ARXIV.2006.02903. URL: https://arxiv.org/abs/2006.02903.

[3] Chenxi Liu et al. *Progressive Neural Architecture Search*. 2017. DOI: 10.48550/ARXIV.1712.00559. URL: https://arxiv.org/abs/1712.00559.

[4] Hanxiao Liu, Karen Simonyan, and Yiming Yang. *DARTS: Differentiable Architecture Search*. 2018. DOI: 10.48550/ARXIV.1806.09055. URL: https://arxiv.org/abs/1806.09055.

[5] Ming Lin et al. "Zen-NAS: A Zero-Shot NAS for High-Performance Deep Image Recognition". In: *International Conference on Computer Vision (ICCV)*. 2021.

[6] Bowen Baker et al. *Accelerating Neural Architecture Search using Performance Prediction*. 2017. DOI: 10.48550/ARXIV.1705.10823. URL: https://arxiv.org/abs/1705.10823.

[7] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: (2019). DOI: 10.48550/ARXIV.1905.11946. URL: https://arxiv.org/abs/1905.11946.

[8] Roy Schwartz et al. *Green AI*. 2019. DOI: 10.48550/ARXIV.1907.10597. URL: https://arxiv.org/abs/1907.10597.

[9] Lasse F. Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan. *Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models*. ICML Workshop on Challenges in Deploying and monitoring Machine Learning Systems. arXiv:2007.03051. July 2020.

[10] Jaime Sevilla et al. "Compute trends across three eras of machine learning". In: *arXiv preprint arXiv:2202.05924* (2022).

[11] Aaron Klein and Frank Hutter. *Tabular Benchmarks for Joint Architecture and Hyperparameter Optimization*. 2019. DOI: 10.48550/ARXIV.1905.04970. URL: https://arxiv.org/abs/1905.04970.

[12] Xuanyi Dong and Yi Yang. "NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search". In: *International Conference on Learning Representations (ICLR)*. 2019.

[13] Wei Wen et al. *Neural Predictor for Neural Architecture Search*. 2019. DOI: 10.48550/ARXIV.1912.00848. URL: https://arxiv.org/abs/1912.00848.

[14] Arber Zela et al. "Surrogate NAS Benchmarks: Going Beyond the Limited Search Spaces of Tabular NAS Benchmarks". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=OnpFa95RVqs.

[15] Chris Ying et al. "NAS-Bench-101: Towards Reproducible Neural Architecture Search". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, Sept. 2019, pp. 7105–7114. URL: http://proceedings.mlr.press/v97/ying19a.html.

[16] Arber Zela, Julien Siems, and Frank Hutter. *NAS-Bench-1Shot1: Benchmarking and Dissecting One-shot Neural Architecture Search*. 2020. DOI: 10.48550/ARXIV.2001.10422. URL: https://arxiv.org/abs/2001.10422.

[17] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[18] Oswin Krause, Tobias Glasmachers, and Christian Igel. "Multi-objective Optimization with Unbounded Solution Sets". In: *NeurIPS Workshop on Bayesian Optimization (BayesOpt 2016)*. 2016.

[19] Sergio Izquierdo et al. "Bag of baselines for multi-objective joint neural architecture search and hyperparameter optimization". In: *8th ICML Workshop on Automated Machine Learning (AutoML)*. 2021.

[20] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International conference on machine learning*. PMLR. 2019, pp. 6105–6114.
[21] Peter Henderson et al. "Towards the systematic reporting of the energy and carbon footprints of machine learning". In: *Journal of Machine Learning Research* 21.248 (2020), pp. 1–43.
[22] Andrew G Howard et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: *arXiv preprint arXiv:1704.04861* (2017).
[23] Yunho Jeon and Junmo Kim. "Constructing fast network through deconstruction of convolution". In: *Advances in Neural Information Processing Systems* 31 (2018).
[24] Lucas Høyberg Puvis de Chavannes et al. "Hyperparameter Power Impact in Transformer Language Model Training". In: *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*. Virtual: Association for Computational Linguistics, Nov. 2021, pp. 96–118. DOI: 10.18653/v1/2021.sustainlp-1.12. URL: https://aclanthology.org/2021.sustainlp-1.12.
[25] Jesse Dodge et al. "Measuring the carbon intensity of ai in cloud instances". In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. 2022, pp. 1877–1894.
[26] Victor Schmidt et al. "CodeCarbon: Estimate and Track Carbon Emissions from Machine Learning Computing". In: (2021). DOI: 10.5281/zenodo.4658424.
[27] Rhonda Ascierto and Andy Lawrence. *Uptime Institute global data center survey 2020*. Tech. rep. Uptime Institute, July 2020.