# Guidelines for the Quality Assessment of Energy-Aware NAS Benchmarks

Nick Kocher*†, Christian Wassermann*, Leona Hennig‡, Jonas Seng§, Holger Hoos*¶, Kristian Kersting§, Marius Lindauer‡ and Matthias Müller*
*RWTH Aachen University
‡Leibniz University Hannover
§Technical University Darmstadt
¶Leiden University
†Corresponding Author: kocher@aim.rwth-aachen.de

***Abstract*—Neural Architecture Search (NAS) accelerates progress in deep learning through systematic refinement of model architectures. The downside is increasingly large energy consumption during the search process. Surrogate-based benchmarking mitigates the cost of full training by querying a pre-trained surrogate to obtain an estimate for the quality of the model. Specifically, energy-aware benchmarking aims to make it possible for NAS to favourably trade off model energy consumption against accuracy. Towards this end, we propose three design principles for such energy-aware benchmarks: (i) reliable power measurements, (ii) a wide range of GPU usage, and (iii) holistic cost reporting. We analyse EA-HAS-Bench based on these principles and find that the choice of GPU measurement API has a large impact on the quality of results. Using the Nvidia System Management Interface (SMI) on top of its underlying library influences the sampling rate during the initial data collection, returning faulty low-power estimations. This results in poor correlation with accurate measurements obtained from an external power meter. With this study, we bring to attention several key considerations when performing energy-aware surrogate-based benchmarking and derive first guidelines that can help design novel benchmarks. We show a narrow usage range of the four GPUs attached to our device, ranging from $146\,\mathrm{W}$ to $305\,\mathrm{W}$ in a single-GPU setting, and narrowing down even further when using all four GPUs. To improve holistic energy reporting, we propose calibration experiments over assumptions made in popular tools, such as Code Carbon, thus achieving reductions in the maximum inaccuracy from $10.3\,\%$ to $8.9\,\%$ without and to $6.6\,\%$ with prior estimation of the expected load on the device.**

***Index Terms*—Neural Architecture Search, Green AutoML, Energy reporting, Energy estimation**

## I. Introduction

Deep learning has accelerated progress in many machine-solvable tasks such as image recognition [1], image segmentation [2] and natural language processing [3]. The trend away from feature engineering towards architecture engineering has given rise to increasingly large neural network models. Starting from the breakthrough of AlexNet [4] at the ImageNet competition in 2012, various different architectures, including GoogLeNet [5], ResNet [6] and the Transformer [7], have introduced a plethora of ways to bundle and stack different layers of neural networks. In contrast to these manual engineering approaches, neural architecture search (NAS), a research area within automated machine learning (AutoML) [8], aims to automate the complex process of finding best-performing architecture for a given task [9]. NAS has contributed to frontier models in image classification [10] and natural language processing [11].

While architectures get more refined, their energy demand can quickly skyrocket during the long searches on large numbers of GPUs [12]. The increase in search cost, as well as overall high training and inference cost, sparked research efforts towards Green AI and Green AutoML [13], [14]. Beside environmental considerations, Green AI additionally advocates for more efficient AI research, in order to democratise research for smaller-scale and publicly funded institutions.

One way to mitigate high computational requirements for participation in research is through the use of more efficient benchmarking techniques. Surrogate-based benchmarks, such as NAS-Bench-201 [15], replace expensive function evaluations (*i.e.*, fully training a candidate neural network architecture) by proxy evaluations at almost negligible cost. Here, instead of a full training cycle, we only need to query the (pre-trained) surrogate model to retrieve (an estimate of) the performance associated with the given candidate architecture.

Even though only querying the surrogate helps decrease the overall cost of benchmarking new NAS approaches, we are ultimately interested in finding architectures that are themselves energy-efficient during training and inference. Hardware-aware NAS (HW-NAS) promises to identify architectures at the Pareto front of energy efficiency and accuracy [16]. This has important applications in the context of energy-constrained devices, such as battery-powered cars or low-power devices in the Internet of Things (IoT) [17], [18].

To facilitate the multi-objective search for architectures, HW-NAS benchmarks provide additional surrogates for specific hardware [19] and may resort to reusing performance metrics [15]. The Energy-aware Hyperparameter and Architecture Search Benchmark (EA-HAS-Bench) aims to construct a hardware-agnostic search space based on energy measured using Nvidia SMI [20]. As shown in Figure 1, this may lead to inaccurate measurements. For these hardware-agnostic energy-aware NAS benchmarks to work in complementarity with other hardware-aware NAS benchmarks, we define three key design principles:

**Conduct reliable power measurements.** Most modern NAS methods employ some form of multi-fidelity optimisation based on partial information (*i.e.,* based on training for a small number of epochs or on a subset of data). For these methods to work, we need energy measurements that are already reliable at low fidelity.

**Allow for a wide range of GPU usage.** The search space or the proposed datasets need to be diverse enough to include high- and low-power ML models on a single GPU. The benchmark should produce meaningful results for multiple different use cases. A search that only uses 100W on a 800W Nvidia H100 wastes expensive hardware resources, while architectures resulting from a 800W-search will be meaningless for smaller IoT devices.

**Report holistic model cost.** Energy costs of using the final model should include the cost for the complete training device, *i.e.*, not only processors, accelerators and memory, during training and inference. This is especially important for battery-constrained IoT devices, where reliable cost information is crucial for the battery life estimation of the device.

*Main contributions*

Our main contribution is a large-scale study of the EA-HAS-Bench data collection scheme, which we introduce in section V. Based on the previously introduced design principles and using different power measurement strategies, we show that:

1) Nvidia SMI produces **poor correlation** to external power meter measurements on a per-epoch basis. This often results in a mix of high- and low-power states as shown in Figure 1.
2) Measurements using SMI produce **insufficient samples in the low-power epochs** to correctly estimate the energy consumption during those epochs. In contrast to their published code, in their original work, Dou et al. used pyNVML to measure the GPU energy consumption [21]. With the pyNVML-based setup, we were able to reproduce the data collection without error.
3) EA-HAS-Bench has a low GPU usage range, but high correlation with the power meter at full fidelity.
4) Holistic energy measurements tools such as Code Carbon [22] underestimate the energy consumption during training, and we propose a method for an offline calibration of the non-measured consumption.

## II. Related work

With the remarkable success of deep learning techniques across a wide range of applications, the problem of finding optimal architectures for the task at hand has rapidly become important to solve. Pioneering approaches in NAS include the utilisation of Bayesian optimisation [23], reinforcement learning (RL) [12], [24] and evolutionary algorithms (EA) [25], [26], as well as the introduction of differentiable neural architectures that enable the optimisation of architectures using stochastic gradient descent (SGD) [27]–[31]. While RL and EA-based NAS methods often cover a more flexible macro design space, they come with high resource consumption. In contrast, gradient-based approaches focus on micro-level architecture decisions but are significantly more efficient; this is a consequence of treating the search space as continuous instead of discrete, thereby training only one super-network with continuous decisions on components. Thus, the problem of finding an optimal architecture can be decomposed into two distinct levels, where in the inner optimisation, network weights are trained with current architecture components, and in the outer optimisation, the architecture components are updated based on network performance. Overall efficacy is then further enhanced by approximations of the first-order solution of this bi-level optimisation problem. To improve the efficiency of NAS algorithms, recent approaches make use of training-free evaluation scores to guide the search through the space of architectures [32]–[34].

To foster the development, reproducibility, and fair comparison of NAS algorithms, a number of different benchmarks have been proposed. These NAS benchmarks define a search space over the architectures and train all (or a large subset of) candidate architectures to yield performance metrics such as validation and test accuracy. Popular examples of such benchmarks include NAS-Bench-101/201/301 [15], [35], [36] and JAHS Bench [37]. Since solving NAS problems comes with significant consumption of compute resources (and thus electric power), energy-aware NAS benchmarks have been proposed to obtain similar benefits as from existing NAS benchmarks in the more challenging setting of developing energy-efficient multi-fidelity NAS methods. Popular benchmarks include the Energy-aware Hyperparameter and Architecture Search Benchmark (EA-HAS-Bench) [20] and the Energy Consumption-aware NAS benchmark (EC-NAS) [38]. EA-HAS-Bench and EC-NAS aim to be hardware-agnostic. However, there are benchmarks that do consider hardware properties to foster the development of NAS algorithms that search for architectures tailored to certain hardware configurations [19], [39].

## III. Power measurement tools

In this section, we describe the different power measurement tools we used for our experiments.

### A. External power meter

To validate the node-internal power measurements, a calibrated ZES ZIMMER LMG450[1] power meter was connected to the four power supply units (PSUs) of the compute node. Experiments were performed on the same node running Rocky Linux 9.3 using two Intel(R) Xeon(R) Platinum 8 480 processors with 112 cores in total, 2 000 GB of memory, and four H100 GPUs attached to the node. The LMG450 has a measurement accuracy of $0.07\,\% + 0.04\,\%$ of the measuring range. We record the continuously integrated active power of all four channels as measured by the internal shunt current sensors. The sampling frequency was configured to $100\,\mathrm{ms}$,

[1] https://www.zes.com/en/Products/Precision-Power-Analyzers/LMG450

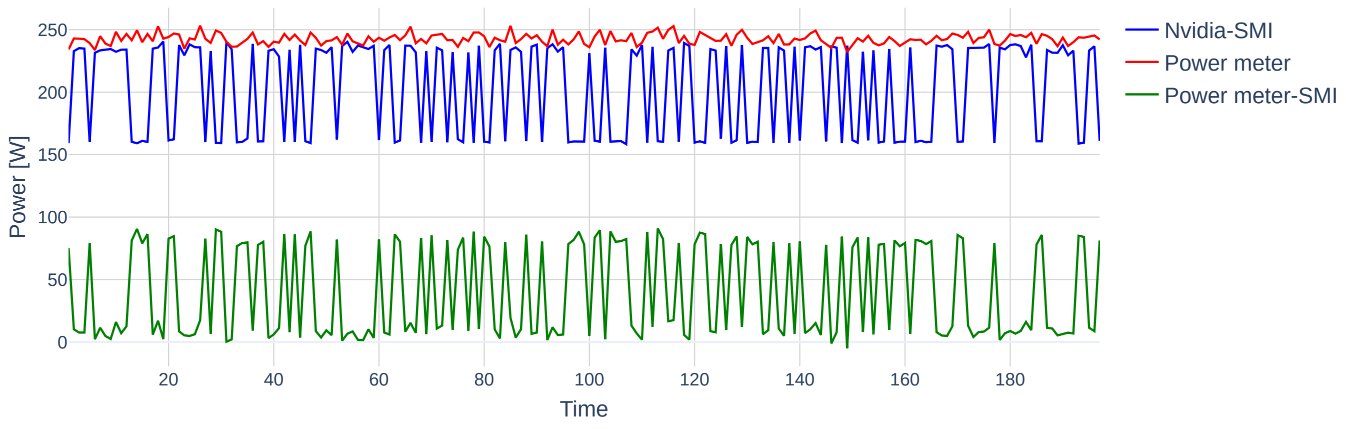

Fig. 1: Switching of low- and high-power states measured by SMI queries during training of a neural architecture on one GPU. The red line is the power measured by the external power meter. The blue line is the power measured by SMI and the green line is the difference between the two.

while the current and voltage ranges were set to $16\,\mathrm{A}$ and $250\,\mathrm{V}$, respectively.

### B. Nvidia SMI

The Nvidia SMI is a command-line utility providing management and monitoring functionality for GPU statistics. Power is measured through the <`power.draw`> call, which by default outputs the average power draw over $1\,\mathrm{s}$ estimated by the GPU itself [40], [41]. This can lead to significantly under-reported consumption, especially during short kernel executions [40]. Steady power loads have a margin of error of $5\%$ on the H100 [40]. The NVIDIA Management Library (NVML) is the library underlying SMI [21]. It also provides a Python binding through `pip` (called pyNVML) and is used for the tools such as Code Carbon; For more details, see subsection III-D.

### C. RAPL

Intel's Running Average Power Limit (RAPL) technology manages the power consumption of the underlying processor [42]. As such, it estimates the energy consumption of certain so-called power domains. On most modern Intel processors, starting from the Sandybridge architecture, (at least) the `PKG` and `DRAM` domains are available, which manage the entire CPU socket and the memory, respectively. For ease of understanding, we will call them CPU and memory consumption going forward.

These domains are accessible through model-specific registers (MSRs), typically exposed in the power cap interface of the Linux kernel. We use the PyRAPL library [43] to import the estimations into the benchmark directly through the provided Python API. Other tools, such as PAPI [44] and LIKWID [45], are popular alternatives to obtain high-level access to the power cap interface. RAPL is generally considered to be accurate on server architectures developed after Sandybridge; the main reason for this is the switch from power modeling [42] using performance counters to a measurement-based approach [46]. Note that to reproduce results from our study, elevated access or ownership of the power cap interface may be required.

### D. Code Carbon

Reliable tools for quantifying and reporting emissions from machine learning models are essential for both research and environmental impact assessments [47], [48]. Code Carbon is such an energy reporting tool specifically designed to track carbon emissions from computational processes by monitoring energy use and regional energy mix in $gCO_2eq$, grams of $CO_2$ equivalents [22]. Grams of $CO_2eq$ quantify the carbon footprint of ML models by accounting for their energy consumption and grid carbon intensity, using the global warming potential (GWP) to standardise emissions from different greenhouse gases. For electricity consumption during computing, this measure is simply based on the proportional amount of $CO_2$ emitted on the power grid, given an appropriate regional energy mix, in our case $43\%$ fossil fuel, resulting in a carbon intensity of $381\,\mathrm{gCO_2/kWh}$. [2]

We use Code Carbon as a proxy for for measuring energy consumption, as it is a widely adopted tool; yet, its reliability requires further assessment [49]. Its main routine is split into two parts. In the first stage, the energy consumption of the executed code block is modelled as

$$E_{codecarbon} = (E_{NVML} + E_{CPU} + \hat{E}_{memory}) \cdot \mathrm{PUE}, \quad (1)$$

where memory consumption $\hat{E}_{memory}$ is estimated using $3\,\mathrm{W}$ per $8\,\mathrm{GB}$ of memory. $E_{NVML}$ and $E_{CPU}$ are the energy consumptions reported by NVML and for the CPU domain in RAPL, respectively. The node energy consumption is then scaled by the power usage effectiveness (PUE). The PUE represents a factor incorporating the additional energy used to operate a compute cluster; an industry average PUE of $1.58$ has been reported for 2020 [50]. In the second stage,

[2]https://github.com/bundesAPI/smard-api

energy consumption is transformed to $CO_2eq$ emissions by multiplying the average carbon intensity ($CO_2eq$ per kWh) of the local power grid with the energy consumption. In our study, we manually extract the energy consumption prior to scaling with the PUE, since it is constant across all experiments and not used by SMI or the external power meter.

## IV. Fair measurements

The main objective of using a power meter in this context is to empirically evaluate how to perform measurements at scale and to improve existing measurement tools. The main complication of using the power meter in our setup is that it measures the energy consumption of the entire node and not merely that attributable to the model training on the GPU. To be able to obtain a fair comparison between SMI and the power meter, the latter needs to be calibrated to a base power consumption; everything above this base level would then be counted towards actual consumption by the benchmark. In Equation 2 below, we present a naïve way to think about capturable power consumption $P_{naive}$ on a compute node. We can obtain $P_{idle}$ from $P_{naive}$ by measuring an idle run using the power meter and subtracting RAPL and SMI measurements. During the actual experiments, this information could then be used to obtain the GPU power consumption as measured by the power meter.

$$P_{naive} = P_{CPU} + P_{Memory} + P_{GPU} + P_{idle} \tag{2}$$

For the idle run, $P_{idle}$ was calculated as 783 W. However, when running a stress test such as Firestarter [51] to maximise load on the CPU and memory, we obtained $P_{idle} = 941\,\mathrm{W}$. This leads us to conclude that there is an uncaptured energy-using component on a node during load; this is made explicit in Equation 3 below. We thus need to determine a $P_{busy}$ that approximates the uncaptured power consumption from the CPU and memory to obtain unbiased GPU consumption during the experiments; see Equation 4.

$$\begin{aligned} P_{total} &= P_{naive} + P_{uncaptured} && (3)\\ &= P_{CPU} + P_{Memory} + P_{GPU} + P_{busy} && (4) \end{aligned}$$

Thankfully, initial experiments show that calculating large prime numbers with a $P_{busy}$ of 811 W is a good lower bound for the busy power consumption.

Our procedure for calculating the GPU power consumption as measured by the power meter thus looks as follows:

$$P_{GPU} = P_{total} - P_{CPU} - P_{Memory} - P_{busy} \tag{5}$$

When comparing the power meter to the measurements from SMI, we use this adjusted power consumption.

## V. Validation Study

The EA-HAS-Bench(mark) provides surrogate models for a complex RegNet search space. RegNets are a variation of ResNets, where residual blocks are replaced by recurrent neural networks, such as LSTMs [52]. The data for the surrogates is gathered by randomly sampling both traditional hyperparameters, as well as architectural details, such as the number of residual blocks from a large search space, and then training the resulting model while measuring the energy consumption using SMI. As claimed by Dou et al., these surrogates show a Pearson correlation coefficient of 0.89 with the collected data. Validating the surrogates is beyond the scope of our study; instead, we are interested in the collected raw energy data. Specifically, we performed a large-scale study of 500 sampled architectures to understand the accuracy of the above described data collection scheme underlying the benchmark.

### A. *Setup of validation experiments*

We mimic the procedure by randomly sampling architectures from the RegNet [52] search space and then sequentially train the resulting models on Tiny Imagenet. Tiny Imagenet is a smaller version of the original Imagenet dataset [53] that contains 1 000 000 images of 200 classes with 500 images per class. Images are downsized to $64 \times 64 \times 3$. This is the largest dataset used in the EA-HAS-Benchmark. We modified the training code of the benchmark to obtain the energy consumption of the entire node from the power meter in addition to the already contained SMI samples. The benchmark samples SMI with a rate of 10 Hz (one sample each 100 ms) from a separate thread. While this is configurable, we used the default of 10 Hz, as the power meter was also storing samples at this frequency.

The power meter output was recorded on a separate external machine, and results were stored after experiments had been concluded. The measurements started and stopped before and after each training run. Therefore, additionally measuring energy with the power meter did not induce any load on the CPU while the architectures were being trained. To obtain the adjusted energy consumption in postprocessing, we also sampled the RAPL values at the end of each epoch using PyRAPL, as outlined in section IV. For this validation study, we only used one out of four available GPUs.

### B. *Validation results*

Epochs measured in this way exhibit a poor energy correlation with the power meter, with a Pearson correlation coefficient of 0.64. When aggregating epochs for the full training of a sampled model, the coefficient improves to 0.99. The comparison between full training and per-epoch measurements is highlighted in the first row of Table I. Throughout the training of the models in this study, the per-epoch measurements are split between low- and high-power states on the GPU, with a plateau of non-measured power states in the empirical cumulative distribution function (eCDF); see Figure 2 as an example. In this specific case, the model had no measured power consumption between 130 W and 150 W. While the eCDF of the power consumption per epoch appears to be strictly monotonically increasing across all training runs, we observed such a split behaviour for almost all of the individual training runs. The reason for this behaviour is the low sampling rate in 39% of the sampled epochs. In these

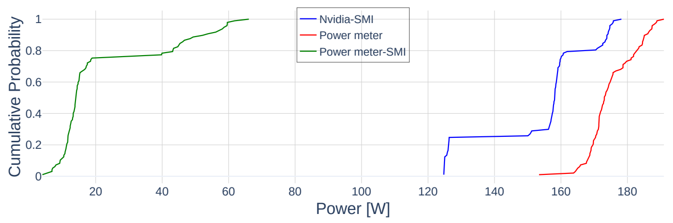

Fig. 2: eCDF of the power meter measurements and the SMI sampled power estimates for an example model training on one GPU.

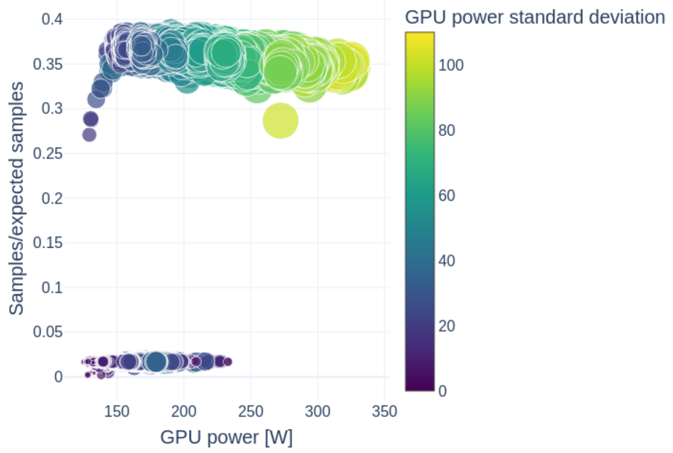

Fig. 3: GPU power vs the ratio of taken samples and expected samples. We see that there are no high power states with a low sample ratio.

epochs, there are no more than 10 power samples measured during an average training time of 11.8 s.

The divide between low and high sampling rates is visualised in Figure 3. In fact, we did not observe any high GPU power epochs with a low number of taken samples in the total number of 53 850 measured epochs. When leaving these epochs out of consideration, the Pearson correlation coefficient increased to 0.95; see the second row of Table I. Overall, the Pearson correlation coefficient between the number of samples per epoch and reported energy usage per epoch was calculated as 0.57. Correcting for low number of samples turned the correlation around to -0.39. Ideally, these values should be uncorrelated. Explaining the reasons for this undesired behaviour is beyond the scope of our study, although we observed in subsequent experiments that it is due to the use of SMI on top of pyNVML.

## VI. Main study

In this section, we first discuss the setup for three different experiments we conducted as a follow-up to the initial study, and then present and discuss results from these three experiments, particularly focusing on the three design principles introduced in section I.

| | Epoch | | | Full Training | | |
|---|---|---|---|---|---|---|
| | Spearman[†] | Pearson[†] | KS[*] | Spearman | Pearson | KS |
| Single GPU | | | | | | |
| SMI | 0.58 | 0.64 | ✗ | 0.99 | 0.99 | ✗ |
| SMI (corrected) | 0.99 | 0.95 | ✓ | 0.99 | 0.99 | ✗ |
| NVML | 0.98 | 0.99 | ✓ | 0.99 | 0.99 | ✓ |
| Code Carbon | 0.99 | 0.99 | ✗ | 0.99 | 0.99 | ✓ |
| Multi GPU | | | | | | |
| SMI | 0.88 | 0.97 | ✗ | 0.99 | 0.99 | ✓ |
| Code Carbon | 0.93 | 0.99 | ✗ | 0.99 | 0.99 | ✓ |

[*] KS: Kolmogorov–Smirnov test, $\alpha = 0.05$, [†] correlation coefficients

TABLE I: Correlation of measurements obtained via different tools with the power meter measurements.

### A. Setup of experiments

All three experiments used the same general protocol as the initial study and were executed on the same hardware. The follow-up experiments, in which we sampled 20 architectures per experiment, enrich the data collection by using additional power measurement tools and training procedures. In all of the experiments, we measured the energy consumption using Code Carbon in addition to the power meter to understand the quality of the energy cost reporting. The custom measuring interval of $100\,\mathrm{ms}$ in Code Carbon deviates from the default of $15\,\mathrm{s}$ to match the power meter and the benchmark sampling speed. In the second small-scale experiment, all four H100s were used and the model was trained in a distributed fashion. In the third experiment, we replaced SMI with pyNVML to analyse the low sampling rates reported in the previous section. pyNVML was used in a previous version of the EA-HAS-Bench to perform the data collection, and was, to the best of our knowledge, the method used in the original paper of Dou et al. [20]. We performed the third experiment twice, once only with pyNVML enabled, to confirm the cause of the behaviour from SMI showcased in the previous section, and a second time also with the power meter and Code Carbon enabled.

### B. Reliable power measurements

We now discuss the results from measuring energy consumption using different tools during the training of RegNets as part of the data collection phase of an energy-aware benchmark. In Table I, we present the main aggregated statistics from all experiments. The table includes the Spearman and Pearson correlation coefficients between the energy measured by the tested tool and the power meter, as well as the result of Kolmogorov-Smirnov (KS) test with $\alpha = 0.05$. Our null hypothesis was that the measurements obtained via a given tool follow the same distribution as the ground truth from the power meter.

We tested SMI and Code Carbon for both single-GPU and multi-GPU training, and we present aggregated results for both single epochs and the full training cycle. This includes the results from the preliminary study. Additionally, the results for the single-GPU measurements from SMI corrected for sufficiently sampled epochs and NVML are depicted. For the Code Carbon measurements, we observe an overall high correlation

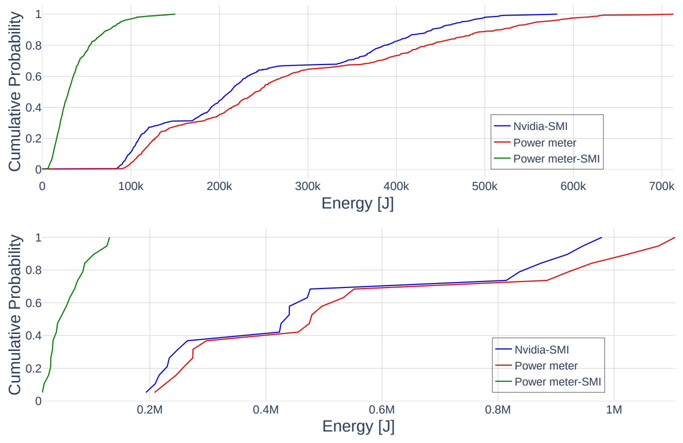

Fig. 4: Top: eCDF of the aggregated single-GPU training for SMI and power meter measurements. Bottom: eCDF of the aggregated multi-GPU training for SMI and power meter measurements.

with the power meter measurements for single epochs, both during the single- and multi-GPU run. This is very similar to the results obtained via NVML. With Code Carbon using the pyNVML API, this further provides evidence towards SMI being the cause for the previously described low sample rate. The measurement quality for SMI improved significantly compared to the preliminary results, when sampling during a multi-GPU run. The Pearson correlation coefficient jumps to 0.97 for single epochs. Only the NVML and corrected SMI distributions pass the Kolmogorov-Smirnov test and are thus statistically indifferent from those obtained from the power meter. The measurements on full training runs all correlate well with the power meter measurements.

The Kolmogorov-Smirnov tests for full training show that all measurements but "SMI on a single GPU" are not significantly differently distributed from the power meter measurement. We note that there are many more full training cycles for the single-GPU-SMI setting, since the large-scale study was performed exclusively using this setting. The similarity between the SMI measurements on single- and on multi-GPU is also highlighted in Figure 4. Here, we visualise the eCDFs over the full training energy costs for the single- and multi-GPU scenarios, respectively. According to both distribution functions, SMI reports consistently less energy than the power meter. The median training cost doubles from $67\,\mathrm{Wh}$ to $132\,\mathrm{Wh}$, while the median inaccuracy from SMI increases from $8.3\,\mathrm{Wh}$ to $12.2\,\mathrm{Wh}$.

### C. Range of GPU usage

The range of different power draws on the GPU is relatively narrow, with $146\,\mathrm{W}$ to $305\,\mathrm{W}$ compared to the base consumption of the GPU of $75\,\mathrm{W}$ and the maximum consumption of $800\,\mathrm{W}$. The GPU was never fully used, with maximum usage around $40\%$. This is a sign that the RegNet search space is not ideal for the H100 GPU. Testing on additional hardware is out of scope of this study, but would be worthwhile investigating in future work. Figure 5 shows the linear correlation between the energy and usage reported by SMI for all epochs measured during the single-GPU measurements. Brighter colours and larger circles indicate higher standard deviations during measurements. We observe that the higher variance also increases almost linearly with the measured power consumption. The multi-GPU measurements on the righthand side of Figure 5 demonstrate different behaviour. The power draw of all four GPUs is uncorrelated to the GPU usage, with a Pearson correlation coefficient of -0.03. The power consumption in this setting is only correlated with the memory usage of the GPUs.

### D. Reporting of holistic energy costs

Dou et al. [20] claim that EA-HAS-Bench is "*aware of the overall search energy cost*". The statistical indifference between the distributions of (corrected) SMI/NVML measurements and the power meter was shown in Table I. We briefly bring attention back to the calculations we had to perform to extract only the GPU power from the power meter, as described in Equation 5 in section IV. While GPU measurements are accurate, the associated consumption does not comprise the majority of the energy cost during training, even during multi-GPU training. Therefore, we analysed whether the accurate measurements could be supplemented by publicly available cost reporting tools, such as Code Carbon, to make the benchmark aware of the *overall* energy cost. In subsection III-D, we described the calculation procedure for Code Carbon. To summarise, Code Carbon uses NVML, RAPL and an estimate of the memory power consumption, in our case $750\,\mathrm{W}$. With the real memory power consumption of our node being $12\,\mathrm{W}$, the overestimation of memory consumption by RAPL helps compensate for the lack of off-socket power consumption on the node. In Table I we see that the measurements from Code Carbon are linearly correlated with those obtained from the power meter. In Figure 6, we visualise for one example model training the constant power difference between Code Carbon and the power meter. This is again highlighted in Figure 7, this time aggregated across the entirety of the second experiment. Compared to the SMI data, we observe a relatively constant underestimation of the power consumption by Code Carbon.

Instead of assuming large amounts of available memory, we base the calculations on empirically evaluated data. To this end, we propose to use a sensible range of estimations between the idle power consumption and the busy power consumption from Firestarter. The range of estimations should be provided by system administrators through the baseboard management controller available on most modern servers. These systems have relatively low sampling rates, usually once every 5 minutes; however, a constant load does not require high sampling rates. If more precise bounds are needed, calibrating with a custom load, such as our prime number calculation, would be required.

We highlight the results of our bounded approach compared to Code Carbon in Figure 8. The base Code Carbon approach

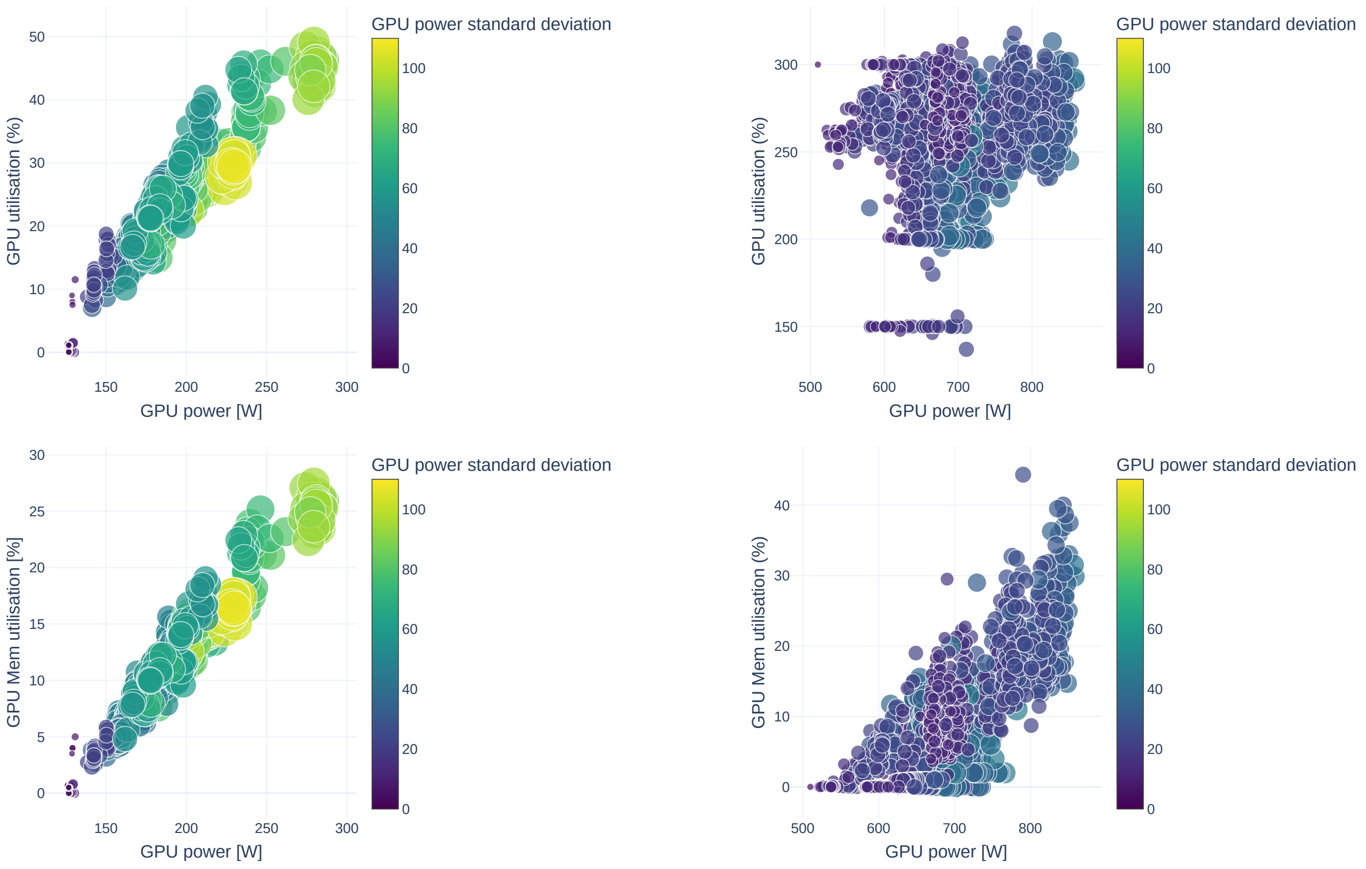

Fig. 5: Top left: GPU power consumption vs *GPU utilisation* for *single-GPU* training. Bottom left: GPU power consumption vs *GPU memory utilisation* for *single-GPU* training. Top right: GPU power consumption vs *GPU utilisation* for *multi-GPU* training. Bottom right: GPU power consumption vs *GPU memory utilisation* for *multi-GPU* training. All data is aggregated across epochs. Brighter colours indicate higher standard deviation.

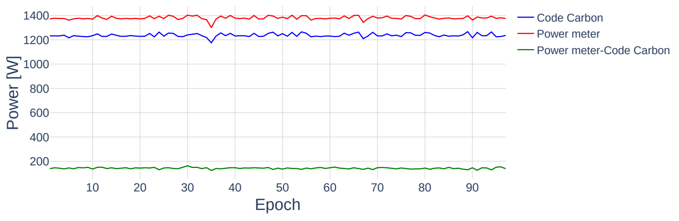

Fig. 6: Power measurements across time for Code Carbon and the power meter for an example model training on one GPU.

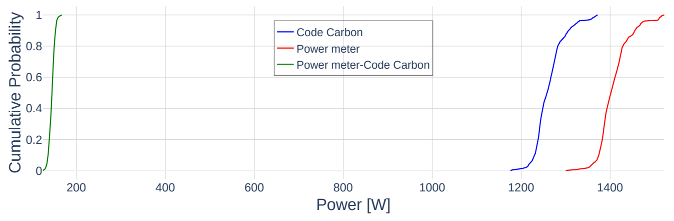

Fig. 7: eCDF plot for the epoch power measurements aggregated across all training runs for Code Carbon and the power meter.

is $10.3\%$ off the power meter, while the bounds given are $8.9$ to $-3.6\,\%$. The load estimation via prime number calculation already improves the upper bound to $6.6\%$. Measurements also do not deviate much from the median. The narrow range of error implies a constant off-socket load for EA-HAS-Bench, which could be estimated through small preliminary experiments. If a non-constant load is encountered, the upper and lower bound still guarantee a valid estimation. Precisely defining what load gives the best bounds for this correction will be explored within future work.

## VII. Conclusions, limitations and future work

In this work, we proposed design principles for energy-aware neural architecture search benchmarks and empirically evaluated the recently published EA-HAS-Bench based on them. Such benchmarks should be built upon reliable power measurements, allow for a wide range of device usage, and report the holistic energy cost for the model. Our study encompasses multiple power measurement tools, including Nvidia SMI, NVML and Code Carbon, which we compared

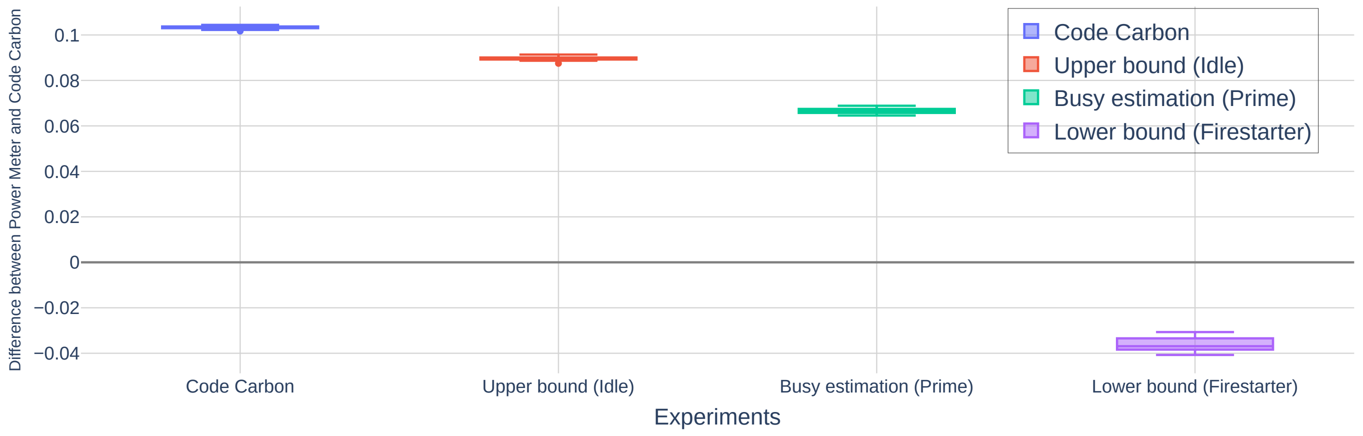

Fig. 8: Boxplot with differences in energy measurements between Code Carbon and the power meter for single-GPU training.

to an external power meter during the data collection phase of EA-HAS-Bench.

For this comparison, we determined the GPU power consumption from the power meter with the help of calibration experiments and Intel's RAPL interface. We also identified a range of off-socket power consumptions caused by different loads on the CPU and memory. The measurements made by Nvidia SMI showed poor correlation to the power meter; we believe that this is caused by the sampling procedure using SMI itself, producing sparsely sampled epochs. The faulty behaviour was validated by a complementary study using the underlying NVML library, in which the power measurements thus obtained were observed to show high correlation with those from the power meter.

We observed poor usage of the GPU during single-GPU training for the RegNet search space sampled in EA-HAS-Bench, with only up to $40\,\%$ of the GPU being used at any given time. For single-GPU training, power consumption was found to correlate linearly with GPU usage and GPU memory usage, while for multi-GPU training, power consumption appeared to be only correlated to GPU memory usage.

We additionally proposed a method to improve the Code Carbon holistic energy reporting by compensating for the overestimation of memory consumption. To this end, we reuse the bounds we calculated for off-socket power consumption. This provides a solid foundation for the reported energy costs based on empirically measured values rather than biased assumptions about memory consumption. In practice, these corrections do not require an external power meter, if the server supports power reporting through the baseboard management controller. Interested parties should contact their local system administrator and advocate for better reporting standards.

There are several limiting factors to our work. We only evaluate on one type of GPU attached to one specific server. This inherently introduces some bias towards high-end GPUs, such as the NVDIA H100 we tested. We sample a relatively low number of architectures during our main experiments. While the number of epochs sampled stays statistically relevant, we cannot guarantee the statistical significance for full training cycles. We argue that the large-scale validation experiment showing high correlation between precisely measured full-training energy consumption and the readings from the power meter justifies keeping the budget low for the main experiments.

In the future, energy-aware NAS benchmarks should sample from a more device-agnostic search space and provide transferability towards hardware-constrained devices. Furthermore, finding the off-socket load on a device a priori for tighter bounds during energy cost reporting is an interesting direction for further investigation. We believe that energy-aware neural architecture search benchmarks should be based on trusted data. With this study, we hope to have contributed to the establishment of best practices for measuring energy-aware benchmarks. More importantly, we enhance the reproducibility of the analysed EA-HAS-Bench and base holistic energy cost reporting tools, such as Code Carbon, on evidence-based foundations rather than assumptions. We are hopeful this will lead to better energy-aware benchmarks in the future.

## Acknowledgment

We thank Anja Jankovic for her feedback throughout the writing process, and Hadar Shavit for his help with designing an efficient experimental setup. This work was supported by the German Federal Ministry of the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (GreenAutoML4FAS project no. 67KI32007A). Further, this work was supported by the Hessian Ministry of Higher Education, Research, Science and the Arts (HMWK; projects "The Third Wave of AI") and the National High-Performance Computing project for Computational Engineering Sciences (NHR4CES). This work was performed as part of the Helmholtz School for Data Science in Life, Earth and Energy (HDS-LEE) and received funding from the Helmholtz Association of German Research Centres. Holger Hoos gratefully acknowledges support through an Alexander-von-Humboldt Professorship in Artificial Intelligence.

## References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[3] S. Hochreiter and J. Schmidthuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, 2012, pp. 1097–1105.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 5998–6008.

[8] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated Machine Learning: Methods, Systems, Challenges*. Springer Nature, 2019.

[9] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: a survey," *Journal of Machine Learning Research*, vol. 20, no. 1, p. 1997–2017, 2019.

[10] H. Pham, Z. Dai, Q. Xie, and Q. V. Le, "Meta pseudo labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 11 552–11 563.

[11] D. So, W. Mańke, H. Liu, Z. Dai, N. Shazeer, and Q. V. Le, "Searching for efficient transformers for language modeling," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 6010–6022.

[12] B. Zoph and Q. Le, "Neural architecture search with reinforcement learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[13] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green AI," *Communications of the ACM*, vol. 63, no. 12, p. 54–63, 2020.

[14] T. Tornede, A. Tornede, J. Hanselle, F. Mohr, M. Wever, and E. Hüllermeier, "Towards Green Automated Machine Learning: Status Quo and Future Directions," *Journal of Artificial Intelligence Research*, vol. 77, pp. 427–457, 2023.

[15] X. Dong and Y. Yang, "NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[16] H. Benmeziane, K. El Maghraoui, H. Ouarnoughi, S. Niar, M. Wistuba, and N. Wang, "Hardware-aware neural architecture search: Survey and taxonomy," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2021, pp. 4322–4329.

[17] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. IEEE*, vol. 107.

[18] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE Netw.*, vol. 32.

[19] C. Li, Z. Yu, Y. Fu, Y. Zhang, Y. Zhao, H. You, Q. Yu, Y. Wang, C. Hao, and Y. Lin, "HW-NAS-Bench: Hardware-Aware Neural Architecture Search Benchmark," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[20] S. Dou, X. Jiang, C. R. Zhao, and D. Li, "EA-HAS-bench: Energy-aware hyperparameter and architecture search benchmark," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[21] NVIDIA Corporation, "NVIDIA Management Library," 2025. [Online]. Available: https://developer.nvidia.com/management-library-nvml

[22] B. C. et Al, "mlco2/codecarbon: v2.4.1," 2024. [Online]. Available: https://github.com/mlco2/codecarbon

[23] L. Zimmer, M. Lindauer, and F. Hutter, "Auto-pytorch tabular: Multi-fidelity metalearning for efficient and robust autodl," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 3079 – 3090, 2021.

[24] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameters sharing," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018, pp. 4095–4104.

[25] T. Elsken, J. H. Metzen, and F. Hutter, "Efficient multi-objective neural architecture search via lamarckian evolution," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[26] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin, "Large-scale evolution of image classifiers," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2017, pp. 2902–2911.

[27] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable Architecture Search," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[28] X. Chu, X. Wang, B. Zhang, S. Lu, X. Wei, and J. Yan, "DARTS-: Robustly Stepping out of Performance Collapse Without Indicators," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[29] X. Chu, T. Zhou, B. Zhang, and J. Li, "Fair DARTS: Eliminating Unfair Advantages in Differentiable Architecture Search," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 465–480.

[30] X. Chen, L. Xie, J. Wu, and Q. Tian, "Progressive differentiable architecture search: Bridging the depth gap between search and evaluation," *Proceedings of the International Conference on Computer Vision (ICCV)*, pp. 1294–1303, 2019.

[31] A. Zela, T. Elsken, T. Saikia, Y. Marrakchi, T. Brox, and F. Hutter, "Understanding and robustifying differentiable architecture search," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[32] G. Li, D.-T. Hoang, K. Bhardwaj, M. Lin, Z. Wang, and R. Marculescu, "Zero-shot neural architecture search: Challenges, solutions, and opportunities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 7618–7635, 2023.

[33] Y. Qiao, H. Xu, Y. Zhang, and S. Huang, "MicroNAS: Zero-Shot Neural Architecture Search for MCUs," *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition*, 2024.

[34] J. Zhang, L. Zhang, Y. Wang, J. Wang, X. Wei, and W. Liu, "An efficient multi-objective evolutionary zero-shot neural architecture search framework for image classification," *International Journal of Neural Systems*, vol. 33, no. 05, p. 2350016, 2023.

[35] C. Ying, A. Klein, E. Christiansen, E. Real, K. Murphy, and F. Hutter, "NAS-Bench-101: Towards reproducible neural architecture search," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[36] J. N. Siems, L. Zimmer, A. Zela, J. Lukasik, M. Keuper, and F. Hutter, "NAS-Bench-301 and the Case for Surrogate Benchmarks for Neural Architecture Search," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[37] A. Bansal, D. Stoll, M. Janowski, A. Zela, and F. Hutter, "JAHS-Bench-201: A Foundation For Research On Joint Architecture And Hyperparameter Search," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[38] P. Bakhtiarifard, C. Igel, and R. Selvan, "EC-NAS: Energy Consumption Aware Tabular Benchmarks for Neural Architecture Search," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, p. 5660–5664.

[39] Y. Fu, Y. Zhang, Y. Zhang, D. Cox, and Y. Lin, "Auto-NBA: Efficient and Effective Search Over the Joint Space of Networks, Bitwidths, and Accelerators," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2021, pp. 3505–3517.

[40] Z. Yang, K. Adamek, and W. Armour, "Accurate and convenient energy measurements for gpus: A detailed study of nvidia gpu's built-in power sensor," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2024, p. 1–17.

[41] NVIDIA Corporation, "NVIDIA System Management Interface Documentation," 2025. [Online]. Available: https://docs.nvidia.com/deploy/nvidia-smi/index.html

[42] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le, "RAPL: memory power estimation and capping," in *Proceedings of*

*the ACM/IEEE International Symposium on Low Power Electronics and Design*, 2010, p. 189–194.

[43] R. Rouvoy, "pyRAPL," 2023. [Online]. Available: https://github.com/powerapi-ng/pyRAPL/

[44] H. McCraw, J. Ralph, A. Danalis, and J. Dongarra, "Power monitoring with papi for extreme scale architectures and dataflow-based programming models," in *Proceedings of the International Conference on Cluster Computing*, 2014, pp. 385–391.

[45] J. Treibig, G. Hager, and G. Wellein, "LIKWID: A Lightweight Performance-Oriented Tool Suite for x86 Multicore Environments," in *Proceedings of the International Conference on Parallel Processing Workshops*, 2010, pp. 207–216.

[46] S. Desrochers, C. Paradis, and V. M. Weaver, "A Validation of DRAM RAPL Power Measurements," in *Proceedings of the Second International Symposium on Memory Systems*, 2016, p. 455–470.

[47] A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres, "Quantifying the carbon emissions of machine learning," *CoRR*, vol. abs/1910.09700, 2019.

[48] K. Lottick, S. Susai, S. A. Friedler, and J. P. Wilson, "Energy usage reports: Environmental awareness as part of algorithmic accountability," *CoRR*, vol. abs/1911.08354, 2019.

[49] L. B. Heguerte, A. Bugeau, and L. Lannelongue, "How to estimate carbon footprint when training deep learning models? A guide and review," *CoRR*, vol. abs/2306.08323, 2023.

[50] D. Patterson, J. Gonzalez, U. Hölzle, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. R. So, M. Texier, and J. Dean, "The carbon footprint of machine learning training will plateau, then shrink," *Computer*, vol. 55, no. 7, pp. 18–28, 2022.

[51] D. Hackenberg, R. Oldenburg, D. Molka, and R. Schöne, "Introducing FIRESTARTER: A processor stress test utility," in *Proceedings of the International Green Computing Conference*, 2013, pp. 1–9.

[52] J. Xu, Y. Pan, X. Pan, S. Hoi, Z. Yi, and Z. Xu, "RegNet: Self-Regulated Network for Image Classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 11, pp. 9562–9567, 2023.

[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.