Computer Standards & Interfaces 92 (2025) 103906

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Energy-efficient neural network training through runtime layer freezing, model quantization, and early stopping

Álvaro Domingo Reguero [a], Silverio Martínez-Fernández [a,*], Roberto Verdecchia [b]

[a] Universitat Politècnica de Catalunya, Spain
[b] University of Florence, Italy

ARTICLE INFO

Dataset link: https://doi.org/10.5281/zenodo.13371442

Keywords:
Green AI
Deep learning
Computer vision
Layer freezing
Model quantization
Early stopping

ABSTRACT

Background: In the last years, neural networks have been massively adopted by industry and research in a wide variety of contexts. Neural network milestones are generally reached by scaling up computation, completely disregarding the carbon footprint required for the associated computations. This trend has become unsustainable given the ever-growing use of deep learning, and could cause irreversible damage to the environment of our planet if it is not addressed soon.
Objective: In this study, we aim to analyze not only the effects of different energy saving methods for neural networks but also the effects of the moment of intervention, and what makes certain moments optimal.
Method: We developed a novel dataset by training convolutional neural networks in 12 different computer vision datasets and applying runtime decisions regarding layer freezing, model quantization and early stopping at different epochs in each run. We then fit an auto-regressive prediction model on the data collected capable to predict the accuracy and energy consumption achieved on future epochs for different methods. The predictions on accuracy and energy are used to estimate the optimal training path.
Results: Following the predictions of the model can save 56.5% of energy consumed while also increasing validation accuracy by 2.38% by avoiding overfitting.The prediction model developed can predict the validation accuracy with a 8.4% of error, the energy consumed with a 14.3% of error and the trade-off between both with a 8.9% of error.
Conclusions: This prediction model could potentially be used by the training algorithm to decide which methods apply to the model and at what moment in order to maximize the accuracy-energy trade-off.

1. Introduction

Since the resurgence of deep neural networks in 2012 due to its breakthrough for image classification [1], deep learning and other related Artificial Intelligence (AI) methods have lived an exponential growth in relevance up to this day. This includes academic interest *via* journal, conference and repository publications, but it is not limited to that. In the private sector, tons of new AI companies have been funded, existing ones have integrated this technology, and the newborn AI sector experienced an ever-growing increase of investments over the last few years [2,3]. The line of improvement of these technologies over the latest years has been scaling up the computation, either directly (parallelization or distributed training) or by changing parameters that indirectly increase it (larger datasets or bigger model architectures), with algorithmic improvements occurring sporadically [4]. The computation scaling trend has led to an exponential increase on the power used in the largest AI training runs by a factor of 10 yearly, yielding a growth of more than 300,000x from 2012 to 2018 [5]. Despite that this rate is starting to slow down in state-of-the-art models due to technical infeasibility to meet the computational requirements [6], most modern policies in AI development still focus on outspending rivals in computational power [7]. Amidst the compute-centered development trend, numerous AI researchers have expressed their concerns about the environmental implications of ever-growing energy-consuming algorithms [8].

In the field of green AI, most of the proposals either integrate compute awareness in the own learning algorithm *prior* to start training, or perform an optimization process *after* training the model. The goal of this study is to conceive a novel framework that allows the designer to pose different training modes, allowing the model to choose the most appropriate one *during* the training according to the predefined objectives and the collected data up to that moment. The framework is based on a monitoring system that collects the data and a control system that modifies the training at will in the middle of it. To develop a decision

* Corresponding author.
E-mail address: silverio.martinez@upc.edu (S. Martínez-Fernández).

https://doi.org/10.1016/j.csi.2024.103906
Received 4 March 2024; Received in revised form 18 July 2024; Accepted 4 August 2024
Available online 9 August 2024

criteria for choosing the method that best follows the objectives, we perform a statistical analysis of the impact that different methods have on the energy consumption and accuracy of a model when applied at different moments of the training phase. To collect the data, we train a Convolutional Neural Network (CNN) on 12 different computer vision datasets during 50 epochs, applying 3 methods (layer freezing, model quantization and early stopping) at 5 different moments each run of the training. The analysis includes the development of an auto-regressive prediction model based on a *what if* analysis and simulation.

The contributions of this work are:

- The creation of a dataset on mid-training decisions of different methods applied over CNNs at different epochs of the training with measures of accuracy and carbon footprint.
- An analysis on the effect of those decisions over the accuracy obtained and energy consumed at the end of the training.
- A data-driven auto-regressive prediction model for accuracy and energy based on a regression over the mentioned dataset.
- A technique to reduce the energy consumption without hindering the accuracy achieved by predicting the optimal decisions during the training based on the prediction models developed.

The rest of the paper is structured as follows: Section 2 describes the background knowledge needed to understand the concepts explained in the study, as well as presenting an overview of the related work. Section 3 formulates the problem to be solved and the goals of the project by defining the research questions (RQs), and the design of the study, including the variables taken considered and the methodology utilized to collect and analyze the results. Section 4 shows the results of the stated analysis and the answers to the RQs. Section 5 discusses the results obtained, their implications, and the future work the results entail. Finally, Section 6 presents the final conclusions of the study.

2. Background and related work

2.1. Background

In the sustainable computing research area, and more specifically on Green AI, new solutions for the sustainable development of AI are constantly searched and tested, motivated by the environmental concerns on the high energy demands of common ML algorithms and the current state of the global climate [9]. Those solutions can target either the training or the inference phase. The energy consumption on the training phase is critical [10]. That is why we opt to focus on methods to improve the energy efficiency for the training of NNs. Precisely, we consider three widespread methods utilized to alleviate the computational requirements during training, namely early stopping, layer freezing, and model quantization.

Early stopping is a regularization method mostly utilized in the training phase to prevent overfitting [11–13]. The method involves monitoring the performance of a model on a validation set during training, and stopping the training process when performance on the validation set begins to deteriorate, allowing to identify the optimal number of training epochs to reduce overfitting. Layer freezing [14–16] instead is a method adopted during neural network training which entails keeping layers unchanged (or "frozen"), while focusing computational efforts on fine-tuning the unfrozen layers. Finally, model quantization is a method entailing the precision reduction of the numbers used to represent parameters models, typically from 32-bit floating point to 16-bit or 8-bit integers, hence decreasing model size and speeding up inference times [17–19]. Model quantization can be applied during training or post-training, and often entails techniques to minimize the accuracy loss associated with lower precision.

In the current literature, algorithmic design techniques for Green AI typically involve taking into account the compute implications of the strategies to determine how to apply them in a deterministic and learnable fashion, such as using reinforcement learning or dynamic parameter adaptation [20]. We decided to take a different approach, namely leveraging the live monitorization of resources, a parallel branch of green AI which was to the best of our knowledge never utilized for energy-aware training algorithms. The major advantage of monitorization is that the algorithm cannot only obtain the current state of the model, but can access also the history of past states. This allows to forecast next states and, with the help of data from other executions, simulate different paths of training according to the different decisions made. To make this possible, we collected data from training multiple neural networks with different datasets of computer vision, applying the mentioned methods at different moments of the training phase.

2.2. Related work

Sustainable software had multiple and ambiguous definitions in its early stage, usually referring only to long-lasting software. The Karlskrona manifesto in 2015 defined sustainability in software by dividing it into five dimensions: environmental, social, economical, technical and individual sustainability [21]. In this work, we focus particularly on environmental sustainability, defined by Calero et al. as "how software product development, maintenance, and use affect energy consumption and the consumption of other natural resources. [...] This dimension is also known as Green Software" [22,23] or "Sustainability IN Software." [24].

In the context of Green AI [22,25], i.e., AI environmental sustainability, one the most relevant paper on the environmental sustainability of deep learning is the work done by Strubell et al. in 2019 [26], that spiked a major research interest in the topic.

Based on the observation that accuracy increased logarithmically with respect to model size [27], dataset size [28], or the number of hyperparameter tuning experiments [29], researches in the past presented light-weight architectures or algorithms that take less computational power to achieve similar accuracy metrics [30–32]. As reported in a dedicated survey [33], common green AI techniques are based on parameter pruning [34] and quantization [35], convolutional filter compression and matrix factorization [36], neural architecture search [37], and knowledge transfer and distillation [38].

As can be evinced by inspecting the literature, a large fraction of studies focus on the inference phase, e.g., the work of Li et al. [39], which showed that inference optimization methods should be followed by millions of inferences to compensate the energy spent in training. While millions of inferences are achieved by some widely utilized deployed models, there are still loads of NNs that do not reach that amount of usage, *e.g.*, test models that never get deployed, or the ones trained for a very specific few-use task. based on such related works, in this study, we opted to focus on the energy efficiency of the training phase.

Regarding papers that propose new methods to improve NN environmental sustainability, we find only few studies on improving the effect of already existing methods by an optimized application of them. As examples, Yang et al. proposed an algorithm for weight pruning that focuses on pruning first the layers that consume more energy, as they showed that traditional algorithms that reduce the total weight or number of operations of the model do not necessarily reduce the energy consumption [40]. Nonetheless, Yang et al. again only measure and reduce the energy consumed on inference time, ignoring the energy consumed during training. Wang et al. instead reviewed minibatching, layer freezing, and model quantization to reduce energy consumed throughout training [41], by proposing algorithm modifications to make them self-adaptive and learnable from the start of the training.

Other related literature explored how inference energy consumption can be optimized on edge devices [42], the impact that hyperparameter tuning can have on power consumption [43], knowledge distillation techniques to make neural machine translation more efficient [43], the impact of NN hardware deployment on energy, and energy-accuracy

trade-offs achievable via structure simplification [44]. As further discussed below, among such vast and heterogeneous amount of green AI literature, no study seemed to date to have focused on the research line we are considering in this work, namely achieving energy efficient model training *via* a mix of known green AI strategies, mid-training monitoring, and execution of runtime decisions.

Regarding the topic of green AI monitoring, numerous frameworks have been proposed to collect and estimate energy and carbon consumption of ML models (*Machine Learning Emissions Calculator* [45], *experiment-impact-tracker* [46], Green Algorithms [47], among others). Some of them include an algorithm to optimize energy consumption, like Perseus, which optimizes parallel GPU job scheduling by identifying straggler pipelines and critical paths of computation. Very few monitoring tools however include a prediction functionality as the one utilized in our study. One of the first approaches to energy prediction was NeuralPower, which proposed a polynomial regression using the model architecture as the predictor [48]. SyNERGY fine-grained this concept to predict energy consumption for each layer based on predicted Single-Instruction Multiple-Data operations and bus accesses, which at the same time are predicted using Multiply-Accumulate operations registered [49]. IrEne decomposes a model into a tree where each node is a ML primitive (such as layers) and its energy is predicted using data from previous executions on different models with similar nodes, to later combine all nodes to get total energy consumed [50]. Wang et al. developed time and energy prediction models for the inference phase using the computation graphs from the neural networks in order to find equivalent graph substitutions with lower power costs [51]. In contrast to the green AI monitoring research presented above, which focus on the inference phase, in this work we leverage monitoring strategy considering the training phase, which as additional novely is used in this work to optimize the energy consumed by training.

In contrast to all the work discussed so far, a framework that combines monitoring and prediction for the training phase is CarbonTracker, which enables self-reporting of real and forecasted compute time, energy consumption and carbon emissions based on measurements on GPU, CPU and DRAM [52]. Building upon such work, we explore the impact that a novel strategy based on forecasting the effects of different green AI techniques, and the runtime application of the techniques based on the computed forecasts, can have on concrete model training energy consumption.

In conclusion, to the best of our knowledge, no previous study has combined at runtime Green AI techniques like layer freezing, model quantization, and early stopping with monitoring and prediction capabilities. Therefore, as most prominent novelty of this work with respect to the related literature, this study presents the integration of these Green AI techniques with monitoring and prediction capabilities, enabling the execution of Green AI decisions during mid-training runtime for the first time.

3. Research methodology

3.1. Research goal

We define the research goal, following the Goal Question Metric (GQM) approach [53], of the project as:
Analyze *the mid-training decisions (layer freeze, model quantization, early stopping)*
with the purpose of *measuring their impact on energy consumption and accuracy*
with respect to *the training phase*
from the point of view of *the ML engineer*
in the context of *neural networks*.

This goal breaks down in two research questions (RQs):

RQ1: *Do the mid-training decisions (layer freeze, model quantization, number of epochs) have any effect on accuracy and energy consumption?*

The aim of RQ1 is to provide an initial analysis on the proposed design decisions, to ascertain their effects on the training of the NN and their worth as an alternative to be considered during the design of the training in the context of energy efficiency. The metrics have been chosen to follow the score defined as $Score = Accuracy/Energy$ presented in [54].

RQ2: *Can accuracy and energy consumption be accurately predicted throughout training for all the design decisions?*

The aim of RQ2 is to provide a reliable prediction model to try and maximize the score in advance, which requires an estimation of the accuracy and energy consumption. The data generated in RQ1 will be interpreted as a time series using the epochs as the temporal unit. This question involves simulating the training to predict the accuracy and energy of further epochs, as well as *what if* analysis to simulate the application of the mentioned methods.

3.2. Study design

With these RQs in mind, we propose a study design consisting of three stages, as shown in Fig. 1, which are:

1. Stage 1: Original data management. This stage consists of collecting, preprocessing, and saving a wide range of computer vision datasets. The output of this stage is a locally saved collection of datasets ready to be directly read and interpreted by the CNN models to train on them.
2. Stage 2: Model training and new data generation. This stage consists of training our CNN with the datasets collected while applying the mid-training design decisions proposed, and saving all the intermediate results. The output of this stage is a novel dataset on green CNN training evolution called `EAT-IT` (Energy Accuracy Trade-off - Interactive Training).
3. Stage 3: New data analysis. This stage consists of analyzing the newly created dataset, interpreted as different time series, to answer RQ1 and RQ2, and creating and evaluating a reliable prediction model for those time series. The output of this stage is in first place an understanding of the data collected in the previous stage and the relationship between the different variables following the research questions, and a usable auto-regressive prediction model for the variables of interest as part of the answer of RQ2 with its evaluation.

To guarantee **replicability**, all the data and scripts used are available in a **Zenodo repository**. [1] To guarantee reproducibility, the study design is composed by basic blocks, each one of them with one or multiple components that can be altered, removed or added without affecting the general structure of the study, allowing for alternative experiments and studies that could expand the knowledge and insights provided by this methodology.

3.3. Variables

In the following subsections we define the variables of our experimental design grouped into three categories, as summarized in Table 1.

3.3.1. Independent variables

In this study we define three independent variables, which are: training mode, epoch of intervention and number of epochs.

Training mode (TM) refers to variations on the setup of the training that have previously been proposed by other authors to modify the performance of the model, in comparison to a by-default "base" training. In this study, the alternative modes are chosen to be layer freezing

[1] https://doi.org/10.5281/zenodo.13371442.

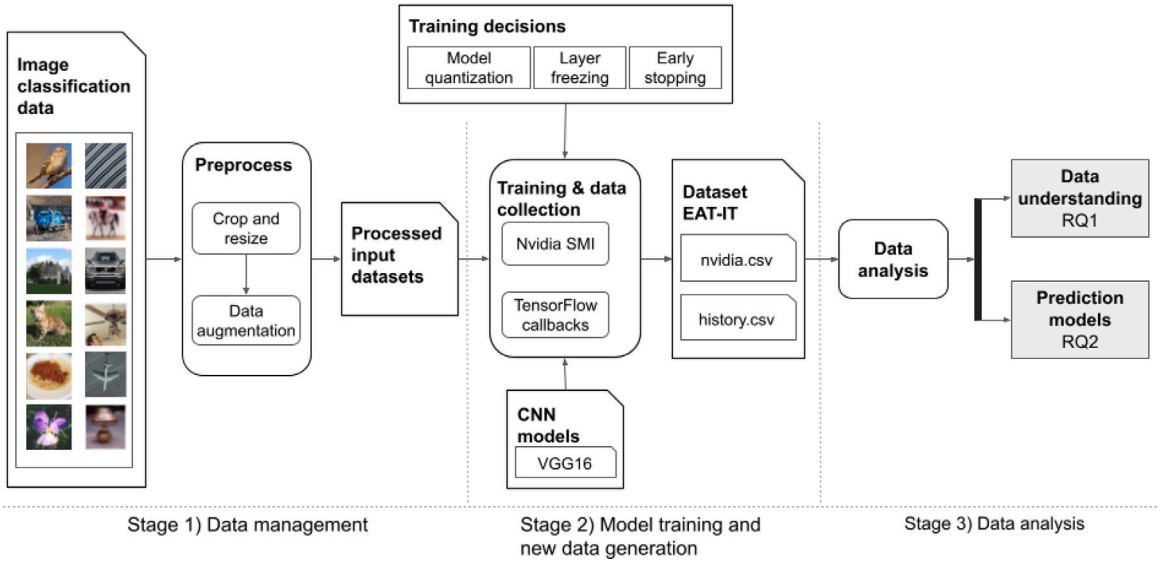


Fig. 1. Schema of the empirical study.

Table 1
Variables of the experiment and their classification.

Class	Name	Description	Scale	Operationalization
Independent	Training mode	The decisions made during the training	Nominal	{Base, layer freezing, model quantization}
	Epoch of intervention	The number of epochs used before taking a decision on training	Numerical	{0, 10, 20, 30, 40}
	Epoch of stopping	The total epochs of training the model	Numerical	[1, 50]
	Dataset	The input dataset used to train the models	Nominal	12 popular CV datasets
Dependent	Energy consumption	Net power supply consumed during the compute time	Numerical	Measured with Nvidia-SMI as kW h
	Accuracy	Validation accuracy obtained after training	Numerical	Measured with TensorFlow

and model quantization. The modes are experimental blocking factors, meaning that a model cannot be freezed and quantized at the same time, to avoid a possible interaction between the two modes [55]. In addition, to limit the number of possible interactions between variables, once a model changes its TM from base mode, it cannot change again neither to base mode nor to another TM, reflecting also that such situation rarely happens in common DL training practices.

The *base mode* is defined as having freezed the base layers of the model, training the top part of it, with a numerical precision of 16-bit floating point. This mode is the "by-default" mode. The base mode is designed to enforce the focus of the research design on fine tuning pretrained models, as it is a very extended practice in DL for saving resources, reducing environmental impact. Additionally, accuracy achieved in fine tuning pretrained models is correlated with accuracy achieved by training from scratch [56].

On the *Layer Freezing (LF) mode*, all layers but the last one are freezed, so that only the output layer can be trained.

On the *Model Quantization (MQ) mode*, all the model is quantized from the default 32 bits to the smaller 16 bits, including the computational results and the model weights, in contrast of the mixed precision, which only quantizes the computational results to the lower precision. 16 bits is chosen over mixed precision in this setup to force bigger energy savings, with a potentially higher accuracy downgrade.

The **epoch of intervention (EoI)** marks the moment at which a decision during the training of the model is made, being a decision a change on the TM. As said before, the *base mode* is the default one with which all trainings are expected to start, so the EoI marks when the TM changes from base to *LF mode* or to *MQ mode*. As only one change of mode is possible in this setup, each record can only have one EoI. This number is a positive integer for a model in a state where its training mode is different to the base one, and is set to zero if and only if the model is at base mode. If the training starts directly in a mode different than the base one, the EoI is defined to 1.

The **epoch of stopping** shows for how long the model has been training, adding up the epochs of base and modified training. This does not imply an actual stop of training, but a simulation of stopping, as all the intermediate results are saved and later analyzed as if the training had really been stopped at that point. In this experiment, the number of considered epochs goes up to 50 for every training.

We use a categorical variable to indicate which **dataset** is the model trained with. Although the effects of different datasets on the model performance is not on the focus of this study, it is still a variable that can affect the results. This variable can take up to 12 values, corresponding to the 12 datasets used to train this model. More information about the datasets can be seen in Table 2.

3.3.2. Dependent variables

As defined in Section 3.1, the focus of this study is the trade-off between energy consumption and accuracy, so the dependent variables measured in this experiment aim to assess those attributes: Energy consumption measured in kW h, and validation accuracy of the models trained.

3.4. Data collection

The starting point for the data chosen to train the CNN in our study is a collection of 12 computer visions datasets used by Kornblith et al. in [56] to test transfer learning and fine tuning over models pretrained on ImageNet. This dataset perfectly fits our study, as the models used here will also be fine-tuned over pretrained weights on the same dataset. The images from the 12 datasets are cropped if the labels include bounding boxes, and resized to a resolution of 32 × 32 pixels to match the lowest resolution of the datasets, corresponding to the resolution of *cifar10* and *cifar100*. This also saves computational power, and although the accuracy obtained is lower than the possible with complete resolution, it is still a predictor of it [57].

Fig. 2. Visualization of the training tree recreated, where each ball represents an epoch and each bifurcation a cloning of the model.

The model chosen to be trained is VGG16, as it is composed of fewer layers than similar architectures while still achievig very good results in benchmarks such as ImageNet [58]. In this study, the convolutional layers of the model are imported already pretrained on ImageNet, as transfer learning allows to achieve good performance way faster than training from zero. The pretrained layers are frozen, to which two fully-connected layers of 4096 channels are added, apart from the output layer with a variable number of channels depending on the dataset, to match the original architecture proposed. The chosen software in which develop the training is the Python library TensorFlow, as it includes a pretrained version of VGG16 as well as most of the datasets used.

The training decisions to be made are four:

- Keeping the model untouched as previously defined, which is said to be the "base" training mode.
- Layer freezing freezes the two fully connected layers, leaving only the last one as trainable.
- Model quantization is performed over all the weights of the model, as well as for the intermediate computations and results. Model quantization is mutually blocking with layer freezing, so they will never be applied together over the same training.
- Early stopping is mocked by selecting the metrics achieved at the desired epoch during the analysis, although the training is not stopped.

The models are trained for a total of $N = 50$ epochs, as that is far beyond the convergence point of the validation accuracy of the models. Such convergence point of validation accuracy is between 10–15 epochs on average, as seen in Fig. 13. The training mode can be changed at regular intervals of $s = 10$ epochs starting at 0 (a whole training in that mode). This generates 5 decision points at which $m = 2$ decisions can be taken, apart from letting the model in the current mode (early stopping is not performed in this phase), forming a decision tree. As the training decisions are blocking, and once taken the model cannot revert back to base mode, the decision tree has $m\lceil\frac{N}{s}\rceil + 1 = 11$ leaf nodes. This tree can be visualized at Fig. 2. Each model training on each dataset follows this tree structure, called from now on training tree, meaning that only one model is created at first, cloned at each decision node for the different branches, avoiding redundant computation for different paths that start with the same decisions. That makes each training on a dataset consist of 350 epochs following the training tree structure, in comparison with 550 epochs that would be needed to train each of the 11 leaf nodes of decision. The datasets are trained on a random order, and the training tree is explored following a Depth First Search strategy to save space, running the leaf branches first in every bifurcation.

Table 2
Datasets used and their characteristics.

Dataset	num_classes	train_size	test_size	from_tensorflow
birdsnap	500	37 354	2500	No
caltech101	102	3059	6085	Yes
cars196	196	8144	8041	Yes
cifar10	10	50 000	10 000	Yes
cifar100	100	50 000	10 000	Yes
dtd	47	1880	1880	Yes
food101	101	75 750	25 250	Yes
oxford_flowers102	102	1020	6149	Yes
oxford_iiit_pet	37	3680	3669	Yes
sun397	397	76 127	21 750	Yes
visual_domain_decathlon/aircraft	100	3334	3333	Yes
voc	20	2501	4952	Yes

The training is performed on the high performance servers provided by /rdlab, a Research and Development Lab founded by the UPC to foster the research on computer science.[2] A 250 m^2 TIER II+ ANSI/TIA-942 certified space, with a dual cooling system, redundant power generator and a 24×7 monitoring service. Accuracy after each epoch is registered using already implemented TensorFlow callbacks due to the simplicity of its integration, and energy consumption is measured with Nvidia-SMI, due to its ease of use and accuracy due to being implemented and accessing actual measurements on the Nvidia GPUs. Nvidia-SMI queries the power draw to sensors built in the GPU itself with a sampling frequency of 1 s [59]. As it is an energy profiler, Nvidia-SMI gives estimations. These estimations correlate with real consumption [60], and have been correctly used in previous studies [61].

The data is saved in a dataset called `EAT-IT` consisting of 3 files, public to use in the replication package. Two files correspond to the two sources of data collection: TensorFlow callbacks (`history.csv`) and Nvidia-SMI (`monitor.csv`). The third one is an auxiliary table explaining information about the datasets chosen to be used, which is presented for completeness in Table 2.

3.5. Data analysis

Below, we describe the process to analyze RQ1 and RQ2. For the tests described here and the results reported in Section 4, the level of significance is defined to be $\alpha = 0.05$. The code is available in the replication package highlighted in the data availability statement and the end of the introduction.

[2] https://rdlab.cs.upc.edu/, last visited 11/09/2023.

Table 3
Tests performed for RQ1 with their null hypotheses.

Variables	Null hypothesis	Statistical test	Results
Accuracy on three TM	There is no difference between the accuracies obtained by the three different TMs	Kruskal–Wallis	Table 4
Accuracy on base and LF modes	There is no difference between the accuracies obtained under base mode and under LF mode	Wilcoxon	Table 4
Accuracy on base and MQ modes	There is no difference between the accuracies obtained under base mode and under LF mode	Wilcoxon	Table 4
Accuracy on epoch of stopping	There is no difference between the accuracies obtained at each epoch of the training	Kendall	Fig. 6
Energy on three TM	There is no difference between the energy consumed by the three different TMs	Kruskal–Wallis	Table 5
Energy on base and LF modes	There is no difference between the energy consumed under base mode and under LF mode	Wilcoxon	Table 5
Energy on base and MQ modes	There is no difference between the energy consumed under base mode and under LF mode	Wilcoxon	Table 5
Energy on epoch of stopping	There is no difference between the energy consumed up to each epoch of the training	Kendall	Fig. 9

3.5.1. Data analysis for RQ1

For RQ1, the analysis is divided in two parts, corresponding to the two dependent variables we are studying, namely **accuracy and energy consumption**.

In order to compare accuracy over different factors, directly using the accuracy can be problematic, as an additive increase should be interpreted differently depending on the baseline: an increase of 1% in accuracy is different over a base 50% or over a base 99%. For that reason, in RQ1, the accuracy is transformed into a derived metric as in [56]: a logit function $logit(p) = sigmoid^{-1}(p) = log(p/(1-p))$, that is, the log odds. To eliminate the effect of varying difficulty across datasets, the metrics of accuracy and energy are adjusted by subtracting the difference between the group mean and the dataset mean. That is, if x_{md} denotes a metric of the model m in the dataset d, the centered metric is computed as $x'_{md} = x_{md} - \overline{x_d} + \overline{x}$ [62]. This way, all datasets have the same average log odds, making it simpler to compare the variance caused by the effects of a modified training across different datasets with a different complexity.

To test if layer freezing or model quantization affect the energy consumption and accuracy, the data collected at the end of each training phase is selected, and the results from the base training are jointly compared to that ones of the two methods, independently for each epoch of intervention, with a Kruskal–Wallis test [63]. If this test retrieves a significant difference of medians, the different treatments are compared individually to the base training with the Wilcoxon test [64].

To test if early stopping affects performance, data from each of the 50 epochs of the training is extracted (simulating that the training had stopped at that moment) and compared to the data at the end of the 50 epochs by computing the Kendall correlation coefficient [65].

All these tests performed in RQ1 are summarized in more detail in Table 3.

3.5.2. Data analysis for RQ2

For RQ2, in which we create models to predict accuracy and energy consumption and validate their accuracy, we follow the pipeline summarized in Fig. 3.

Predicting accuracy. The prediction model on accuracy is an auto-regressive model over previous accuracies and training modes fitted using panel data analysis, treating each dataset as a different individual that might have specific parameters while still sharing common ones with all the group. Pooled Ordinary Least Squares, fixed effects and random effects models are tested and compared to select the model which best fits our data [66]. Pooled OLS model ignores all differences of individual-level characteristics by pooling all time series and performing the same linear regression over all the different entities, fixed effects model estimates individual-specific intercepts to account for the differences between individuals assuming that those are fixed over time, and random effects model assumes that the individual-level effects are random and uncorrelated with the independent variables. Pooled OLS represents a nested model of fixed effects, so they can be compared with an ANOVA or a Wald test [67]. Fixed effects is compared to random effects via Hausman test [68], considering the former as consistent but less efficient, and the latter as more efficient but whose consistency needs to be tested.

To know how many previous values (lags) have to be considered in order to make reliable predictions without overfitting, different models are fitted, every one with a different amount of values lagged. For a model or order n, the n coefficients corresponding to the n lagged values are tested to be significantly different to 0 with a t-test. As multiple tests are being performed, increasingly more as n grows, the probability of a type I error (a false positive) increases. To control it, the Bonferroni correction method is applied, in which the significance level gets divided by the number of tests being performed [69]. If on a model of order n all coefficients are significantly different to 0, and on the model of order $n+1$ they are not, then n is the optimal number of lagged values to use for the prediction. This methodology is performed first to determined the lags needed on accuracy. Then, this number is fixed and the same methodology is performed to determine the lags needed on the TM.

The model precision is then measured using 12-fold cross-validation, dividing the data according to the 12 datasets in which the model has been trained. The data collected from the 11 training datasets is used to estimate *a priori* the regression coefficients for the lagged accuracy and TM values. The individual-specific coefficients for each dataset can be discarded as it is not relevant. The model then traverses the training tree of the twelfth dataset performing inferences on the next epoch of the tree. The data collected on this dataset up to the inference epoch is used to estimate the individual-specific parameter. This parameter is simply an additive correction over the prediction obtained using the common ones. It can then be estimated by the average difference between the accuracies predicted before each epoch without any correction and the real accuracies measured after the epoch. This factor is the one that, when added to a predicted accuracy, minimizes the expected error of it with the real measured one. For that, the model needs $n+1$ starting values of accuracy, being n the number of lagged values of accuracy with which the auto-regressive model is fitted, and one extra to estimate the individual-specific parameter.

Predicting energy consumption. As, on the same dataset and under the same training mode, the computational effort of each epoch is the same, the energy consumption of each one should be independent from any other factor of previous ones, without defining any temporal

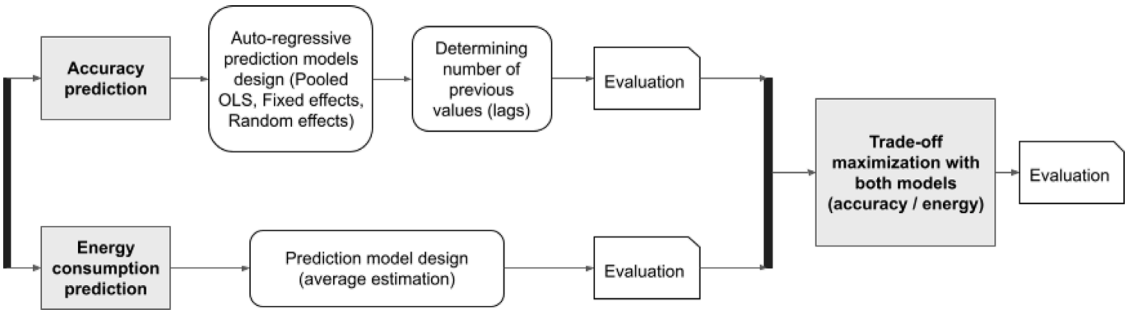


Fig. 3. Pipeline of the design and evaluation of the prediction models in RQ2.

pattern. This is tested with the Ljung–Box test [70], under the null hypothesis that there is no autocorrelation on the series of energy consumed at each epoch. If it is not rejected, temporal independence of energy consumed between epochs is assumed, and energy consumption is assumed then to be drawn from some unknown random distribution around an unknown mean. Under that assumption, the best prediction possible for future epochs is just the average energy consumed in previous epochs on the same dataset and TM. In order to predict the energy that would consume the next epoch on a different training mode that the current, two smaller regression models are fitted. These models use as the predictor the average energy consumption on each dataset under base mode, and as the response the average energy consumption on each dataset for layer freezing and model quantization modes each. In the event of a change of TM, the corresponding model is used to estimate the average energy consumption under that TM from the average registered energy consumption under base mode up to that epoch.

The energy prediction model is evaluated on a similar 12-fold cross-validation way as the accuracy prediction model. 11 datasets are used to estimate the regression over TM averages, and inferences are performed on the twelfth one traversing the training tree. The model uses $n+1$ starting values of energy consumption to be the same as the accuracy prediction model.

Maximizing trade-off of both accuracy and energy consumption. The predictions to the score $Score = Accuracy/Energy$ are just performed using the two previous models, dividing the estimated accuracy of the next epoch by the sum of the already consumed energy and the estimated for the next epoch. The precision of this model is measured with the same 12-fold cross-validation technique as both the previous models, using also $n+1$ starting values of accuracy and energy consumption.

This model is then used to minimize energy consumption by choosing the best TM in which train the next epoch and the best epoch to stop in order to find the optimal predicted trade-off score. As, across epochs, the accuracy is characterized as an increasing function with diminishing returns until stabilization, and energy consumption as a constantly increasing function, the score is characterized as a decreasing function that approaches 0, decreasing rapidly at first and slowing over time. The trade-off between accuracy and energy is then defined to be optimum when the rate of increase of both variables is equal, that is, when the local derivative of the score function is -1. To reduce the effect of outliers of accuracy or energy on the calculation of the local score derivative, the measured score up to the inference epoch is concatenated to the predicted score for the next 10 epochs and fit to a function of the form $Score \approx \frac{c}{epoch+b}$, being c and b constants estimated via non-linear least squares. This function is the one used to estimate the optimal epoch from its derivative. To decide which TM should the next epoch be trained on, the optimal score (being the one obtained at the optimal epoch) obtained with all possible TMs is computed, and the mode with the best one is selected.

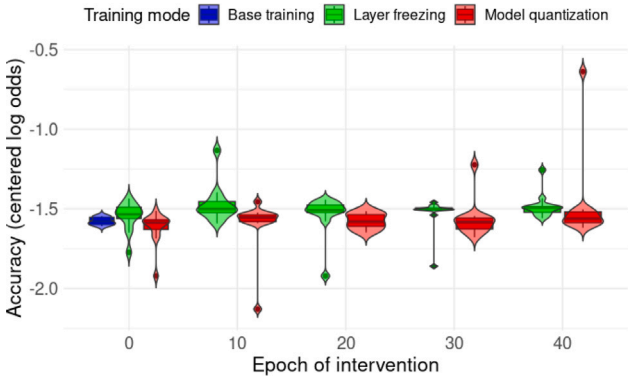


Fig. 4. Violin plot for centered log odds for each epoch of intervention.

4. Results

In this section we discuss the quantitative results in response to the RQs and hypotheses presented in the previous section.

4.1. Do the training decisions have any effect on accuracy and power consumption? (RQ1)

4.1.1. Effects on accuracy

Fig. 4 shows the violin plot of the correlation between the TM (base, layer freezing or model quantization) and the epoch of intervention for each mode with the accuracy (processed as explained in 3.4) obtained at the end of the training across all datasets. Table 4 shows the p-values for the different tests performed. The first one is the Kruskal–Wallis between the 3 training modes. As can be seen, for all 5 epochs of intervention tested the p-value is below the significance level, which leads to reject the null hypothesis and conclude that at least one group is different to the rest.

As a follow-up to this result, the training decisions are compared individually to the base training with Wilcoxon tests. Results of the tests between LF and base mode show a p-value under the significance level for each EoI tested, which leads to reject the null hypothesis in favor of the alternative one, meaning that, no matter the moment where the layers are freezed, it has a significant positive effect on accuracy. On the other hand, no p-value of the Wilcoxon tests between the MQ and the base mode fall below the significance level, so the null hypothesis cannot be rejected. Therefore, there is no evidence that supports the hypothesis that model quantization affects the accuracy of the model at the end of the training no matter the EoI.

Fig. 5 shows the box plot of the correlation between the epoch at which the training has been stopped with the accuracy achieved at that point of the training. Accuracy shows a Kendall's τ coefficient of 0.1594 with a p-value on the order of 1×10^{-53}, which marks that the estimated coefficient as significantly different to 0. As can be seen on the box

Table 4
p-values for accuracy tests.

EoI	3-way Kruskal	Wilcoxon base vs. LF	Wilcoxon base vs. MQ
0	0.0339	0.0376	0.443
10	5.38×10^{-4}	2.01×10^{-4}	0.0597
20	0.00232	0.00182	0.843
30	0.00122	2.01×10^{-4}	0.590
40	2.51×10^{-4}	8.88×10^{-6}	0.160

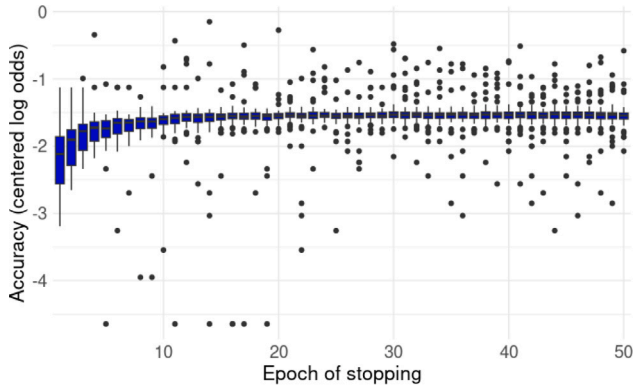


Fig. 5. Box plot for centered log odds at each epoch of the training.

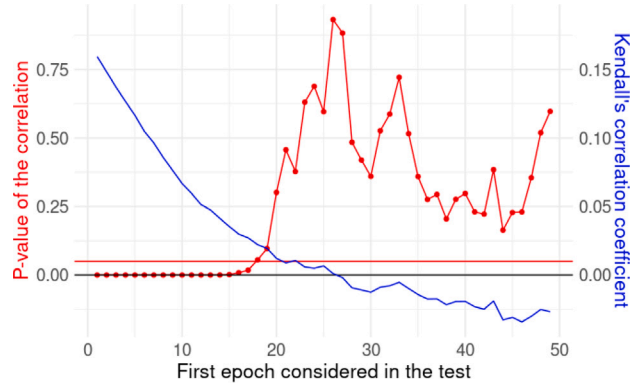


Fig. 6. Kendall's τ coefficient for each epoch of start and the centered log-odds, with its *p*-value and the significance threshold in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

plot, accuracy seems to stabilize at a certain point during training. To find the stabilization point, Kendall's τ has been estimated multiple times, each one taking into account one epoch less at the beginning, so that it arrives a moment when the plot of the accuracy of the selected epochs is not significantly different than a flat line. Fig. 6 shows the estimated correlation coefficients with their *p*-values against the epoch at which data has started being considered. The first value to get above the significance level is starting at 18 epochs, with a τ of 0.022 and a *p*-value of 0.555, being all following values even higher. This means that stopping at epoch 17 or before has a significant negative impact on the accuracy achieved, but every epoch after that has no significant difference to the accuracy reached at the end of all 50 epochs.

4.1.2. Effects on energy consumption

Analogously, Fig. 7 shows violin plot of the correlation between the training mode and the EoI for each mode with the total energy consumed at the end of the training, and Table 5 the *p*-values for the different tests done. This time, *p*-values of the Kruskal–Wallis tests are below the significance level only if the EoI is 20 or lower, not being able to find significant difference in energy consumed when the intervention it is 30 or higher.

Table 5
p-values for energy tests.

EoI	3-way Kruskal	Wilcoxon base vs. LF	Wilcoxon base vs. MQ
0	0.0384	0.0242	0.932
10	4.92×10^{-4}	7.17×10^{-5}	0.443
20	0.0140	0.0121	0.932
30	0.107	0.0780	0.932
40	0.881	0.671	0.977

Wilcoxon tests between base and MQ mode do not identify a significant difference between the energy consumed of the two modes, no matter when the quantization is conducted. Nevertheless, the tests between LF and base mode show the same pattern as the Kruskal–Wallis test, rejecting the null hypothesis of equal energy consumption only for an EoI of 20 or lower. An exponential function has been fitted to the 5 *p*-values with respect to their EoI via non-linear least squares. The interpolated function represents the estimated *p*-value that a Wilcoxon test between a supposed energy consumed for base and for layer freezing training would retrieve for each EoI. This estimated *p*-value surpasses the significance level at a value of 27.88 epochs before intervention, which leads to the estimation that if a significant negative difference in energy consumption is sought, layer freezing must be performed at epoch 27 of before.

Fig. 8 shows the box plot for the centered energy consumed up to each epoch. This retrieves a Kendall's τ coefficient of 0.882 and a *p*-value smaller than the numerical precision of the software ($2.2{\times}10^{-308}$), which means strong evidence to support the correlation between the epoch of stopping and the energy consumed. Moreover, when considering progressively less epochs, the correlation coefficient decreases under 0.3 as can be seen in Fig. 9, but the *p*-value always remains way under the significance level, with the maximum *p*-value reached being just 2.04×10^{-7} when just considering the last two epochs. This means that there is a significant negative difference on the total energy consumed at the end of the training and at every epoch before.

Key findings for RQ1 (Training Decisions):

Finding 1.1: Layer freezing has a positive effect on final accuracy achieved, no matter at which epoch it has been performed. Layer freezing also has a negative effect on total energy consumed as long as it is performed at epoch 27 or before.

Finding 1.2: Model quantization has no effect on model accuracy nor energy consumption, no matter at which epoch model quantization is performed.

Finding 1.3: Early stopping has a negative effect on final accuracy as long as it is performed at epoch 17 or before, and after that the accuracy achieved does not significantly change. Early stopping also has a negative effect on the total energy consumed no matter at which epoch it is performed.

4.2. Can accuracy and energy consumption be accurately predicted throughout training for all the design decisions? (RQ2)

4.2.1. Accuracy prediction

Different panel data analysis methods are used in order to create a prediction model for the accuracy. To test which method fits better the data, first only the base mode training data is selected to generate a single, unbranched time series for each individual (here represented by the dataset), and the independent variable is lagged once to get the dependent one.

First, pooled OLS and fixed effects models are compared via an *F*-test, which retrieves a *p*-value smaller than 1×10^{-50}, rejecting the null

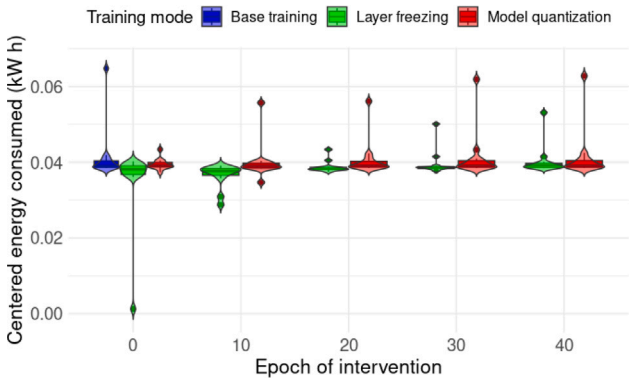


Fig. 7. Violin plot for centered energy consumption for each epoch of intervention.

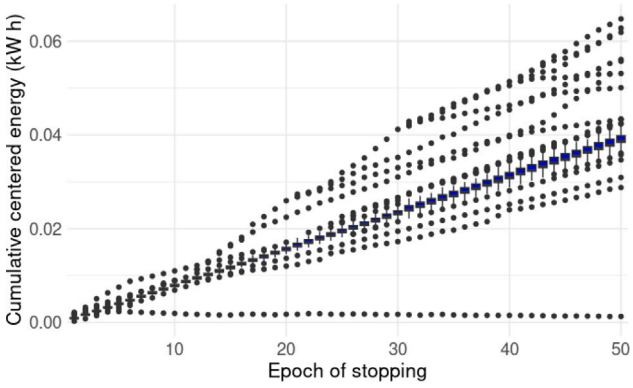


Fig. 8. Box plot for centered energy consumed up to each epoch of the training.

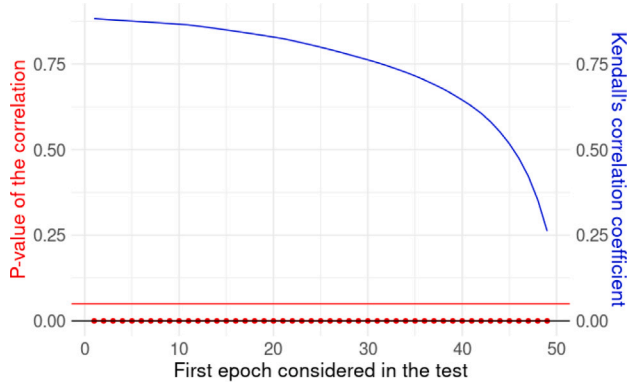


Fig. 9. Kendall's τ coefficient for each epoch of start and energy consumed, with its p-value and the significance threshold in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

hypothesis that both models are equally good favoring fixed effects. On top of that, a Wald test is performed on the coefficients added on the fixed effects model with respect to pooled OLS, with the null hypothesis that they are all equal to 0. This test retrieves a p-value of virtually 0, rejecting that hypothesis and again proving a difference between both models. Then, this model is compared to a random effects model via a Hausman test, which retrieves a p-value smaller than 1×10^{-50}, ruling the random effects inconsistent in favor of the fixed effects. All those tests are repeated using more lagged values on the data, and all of them give the same general results. More details on the models and tests can be seen on Tables 6 and 7. From the collected results we can conclude that fixed effects is the method that better explains the data.

Once the model is chosen, all the data can be taken into consideration, adding an independent variable to codify the training mode.

n models are fitted, each one with $i = 1, \ldots, n$ lagged values of accuracy and TM. The i coefficients corresponding to the lags on accuracy are tested to be significantly different from 0 with a t-test. The first model with at least one coefficient not proven to be different from 0 is the one that uses $n = 5$ lagged values, whose coefficient for the last one is of 0.0126 with a correspondent p-value of 0.297, and a Bonferroni-corrected significance level of 0.01. On top of that, an ANOVA test is performed between every two consecutive models, that is, one fitted with n lagged values and the other one with $n + 1$. The first test that does not give a significant difference between the residual error of two contiguous models is for $n = 4$, with an F-statistic of 0.616 and a p-value of 0.605. This test is then also unable to reject the hypothesis that the addition of the fifth lagged value does not significantly increase the precision of prediction.

Upon a further inspection on the models fitted, the coefficients corresponding to the dummy variable for the MQ mode are never significantly different from 0, neither on the current epoch's mode nor on the lagged modes. This is consistent with the results for RQ1, which proved that there is no difference between the results with the base TM and the MQ one. For that reason, that variable is dropped and the data collected with that mode is treated in the same way as the one collected with the base mode.

After fixing the number of lags on accuracy on $n = 4$, four more models are fitted, using $i = 1, \ldots, 4$ previous values of TM, and the i coefficients are again tested to be significantly different from 0 with a t-test. For each of the models, only the last coefficient from the TM (corresponding to the mode of i epochs before the current one) is significantly different from 0 after the Bonferroni correction, being the previous $i - 1$ not significant, so the only model with all coefficients significant is the one that only uses $i = 1$ lagged value of TM

To sum up, the model reporting the best prediction precision is fixed effects, using 4 lagged values of accuracy and one of training mode as the predictors, without considering quantization mode. This model is evaluated via 12-fold cross-validation over the 12 datasets on which the CNN was trained, as explained in Section 3. The model achieves a sum of squared errors (SSE) of 0.2161 and a mean (MSE) of 5.425e−5, with a mean relative error (MRE) of 8.77%. A visualization of the predictions can be seen in Annex C

4.2.2. Energy consumption prediction

To test the hypothesis of independence, that is, that each sample on energy consumed is independent from the previous one, the autocorrelation coefficients from the energy consumed on single-mode trainings are inspected. All trainings with a change of mode in the middle of it are discarded for this test, as the subsequent change of energy consumption can give place to an undesired correlation not caused by an actual temporal pattern on the data. Therefore, for each dataset 3 series are tested, one for each training mode (base, LF and MQ). The first 10 correlation coefficients from 2 datasets can be seen on Fig. 10, as well as the 95% confidence bands for those being significantly different than 0. Those correspond with the datasets with more and less significant coefficients. It can be seen that only a few of the coefficients surpass the confidence bands, meaning that all the rest are not significantly different from 0 so there is no evidence of autocorrelation. The ones that surpass the bands only do it slightly, so the results can be attributed to spurious results, given the high number of coefficients that are being tested (30 per dataset for 12 datasets).

To further test the presence of autocorrelation on the series, especially for those which had significant coefficients, the Ljung–Box test is applied, which compares the whole group of correlation coefficients to 0. The p-values of the tests can be seen on Fig. 11. As can be observed, for most of the datasets and modes, the test did not find sufficient evidence to reject the null hypothesis of all the correlation coefficients being equal to zero, except for 2 series of the 30 tested, which are

Table 6
Details on prediction models for different lagged values.

Lagged values	Model	Statistic	DF	p-value	SSE	RSE	R^2
1	Pooled OLS	16 181	586	$<10^{-307}$	0.08306	0.01191	0.9950
	Fixed effects	16 181	575	7.6×10^{-135}	0.04895	0.009227	0.9970
	Random effects	116 390	1	$<10^{-307}$	0.08306	0.01191	0.9950
2	Pooled OLS	62 187	573	$<10^{-307}$	0.07467	0.01142	0.9954
	Fixed effects	19 830	562	2.6×10^{-91}	0.03542	0.007939	0.9978
	Random effects	124 375	2	$<10^{-307}$	0.07467	0.01142	0.9954
3	Pooled OLS	52 496	560	$<10^{-307}$	0.05660	0.01005	0.9965
	Fixed effects	18 696	549	4.2×10^{-51}	0.03343	0.007804	0.9979
	Random effects	157 489	3	$<10^{-307}$	0.05660	0.01005	0.9965

Table 7
Details on prediction model tests for different lagged values.

Lagged values	Test	Statistic	DF	p-value
1	F-test OLS vs. FE	56.62	11	4.1×10^{-59}
	Wald test over FE	670.1	11	$<10^{-307}$
	Hausman test FE vs. RE	615.3	2	3.1×10^{-88}
2	F-test OLS vs. FE	36.42	11	1.1×10^{-83}
	Wald test over FE	400.6	11	$<10^{-307}$
	Hausman test FE vs. RE	396.6	2	2.5×10^{-134}
3	F-test OLS vs. FE	34.58	11	6.1×10^{-56}
	Wald test over FE	473	11	$<10^{-307}$
	Hausman test FE vs. RE	375.7	2	4.0×10^{-81}

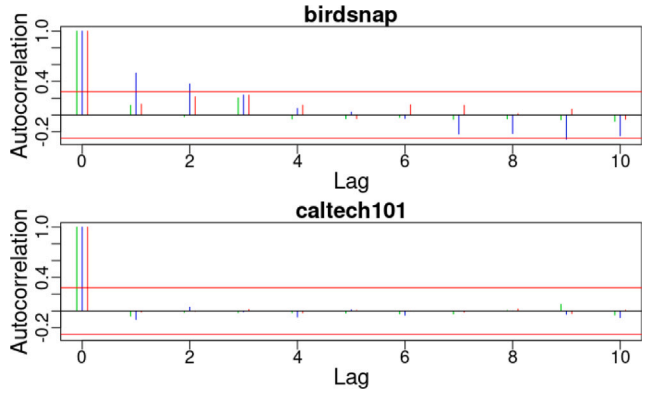


Fig. 10. Autocorrelation coefficients for single-mode trainings for two datasets, one with significant coefficients and one without.

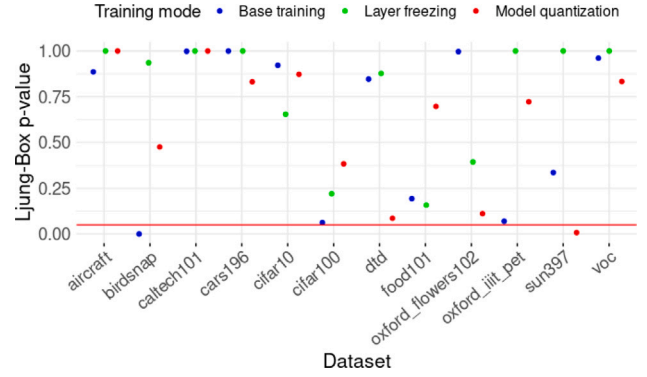


Fig. 11. Ljung–Box test p-value for single-mode trainings for each dataset.

considered outliers as no evidence was found on the other 28 training branches. So, despite of the mixed results on the Ljung–Box test, the independence hypothesis is accepted.

Fig. 12 shows the average energy per epoch consumed in the alternative modes against the base one, with the identity line in blue. A point over that line represents an increase on energy when changing to that mode, while a point under it represents a decrease. It is worth noticing that there is a clear distinction of five datasets that consume a higher amount of energy (precisely *cifar10*, *cifar100*, *sun397*, *food101* and *birdsnap*, in ascending order, with the two first ones overlapping and seeming only one point) and seven that consume much less. For the five high-consuming datasets, all data points are below the identity line, indicating a reduction of energy. On the other hand, some of the low-consuming datasets present a small increase of energy for some modes. Although counter-intuitive and against the general pattern and against the conclusion drawn for RQ1 (in the case of layer freezing mode), this can happen just due to smaller numbers having a bigger variance.

The figure also shows the regression lines fitting the 12 points for each alternative mode, with the intercept fixed to 0 as it was deemed insignificant for both model quantization and layer freezing modes (with p-values of 0.433 and 0.599 respectively). The lines have a slope of 0.956 for MQ and 0.836 for LF mode. This means that the predicted average energy consume per epoch for a dataset on MQ mode will be 95.6% the recorded one on base mode, and 83.6% for LF mode. When compared to the identity line via a t-test, both are proven to be significantly different, with p-values of 0.0146 and 0.00855 respectively. This means that there is enough evidence to reject the null hypotheses that the average energy consumed per epoch is the same for the base mode and for the other ones.

Like for accuracy, the prediction precision is tested via 12-fold cross-validation, estimating the regression lines with the data from 11 dataset trainings and performing inference on the last one. Whenever there are previous epochs with the same training mode, the predicted value is the average energy consumption of those epochs. On the event of a training mode change, the average consumption on the base mode on that branch and the average change estimated with the 11 other datasets are used to retrieve the prediction. This model achieved a SSE of 1.008×10^{-4}, a MSE of 2.508×10^{-8}, and a MRE of 14.3%.

The final prediction model for energy consumption is then the average registered energy consumption for the epochs in the same TM as the one to be predicted and, in the case that there are none, that same average multiplied by a precomputed factor to account for the change in TM. This model achieves a prediction precision of 85.7%. Visualizations of the predictions by this model can also be seen in Annex C

4.2.3. Trade-off maximization

Finally, the trade-off score defined as the ratio between accuracy and energy is predicted using both previous models jointly. The predicted accuracy for next epoch can be used as it is, but the predicted energy must be added to the cumulative energy consumed up to each epoch following its branch from the training tree. This prediction model is tested using the same cross-validation methodology as for the previous ones, and yields a SSE of 59723.5, a MSE of 14.99084 and a MRE of 8.95%.

This score model is used to predict the optimal branch of the training tree, as explained in Section 3. From epoch 6 onwards, as the model needs 5 epochs to perform predictions on accuracy, the score

from the next 10 epochs is predicted in an auto-regressive fashion, fit to a function of form $Score \approx \frac{c}{epoch+b}$ via non-linear least squares, and the optimal epoch is predicted by the point where the local derivative of this function equals −1. If this optimal epoch is the current one or any previous one, the training is stopped. For any bifurcation node in the training tree (that is, every 10 epochs), the score achieved at the optimal epoch for each TM is compared, and the branch selected is the one with the TM that has the highest optimal score. The optimal node from the 12 training trees is selected following this methodology, using for each one the data from the 11 other datasets as training for the model. On average, the optimal node consumed 56.5% less energy than the final node (50 epochs) on the base TM, achieving a relative 2.38% increase on validation accuracy by avoiding overfitting.

Findings for RQ2 (accuracy and energy consumption prediction):

Finding 2.1: Validation accuracy can be predicted using a fixed effects model, using the previous 4 values and the previous training mode, with an accuracy of 91.6%.

Finding 2.2: Energy consumption can be predicted by retrieving the average energy consumption measured up to that point, or a compensated average using a precomputed relationship across training modes, with an accuracy of 85.7%.

Finding 2.3: Trade-off score can be predicted by using the mentioned accuracy and energy predictions, with an accuracy of 91.1%.

Finding 2.4: Trade-off score can be maximized by predicting this score for future epochs for different TMs, choosing the one with highest score at the optimal epoch and stopping at such epoch, defining it as the point where the derivative of the trade-off score equals to −1. This method can reduce energy consumption by 56.5% and increase validation accuracy by 2.38%.

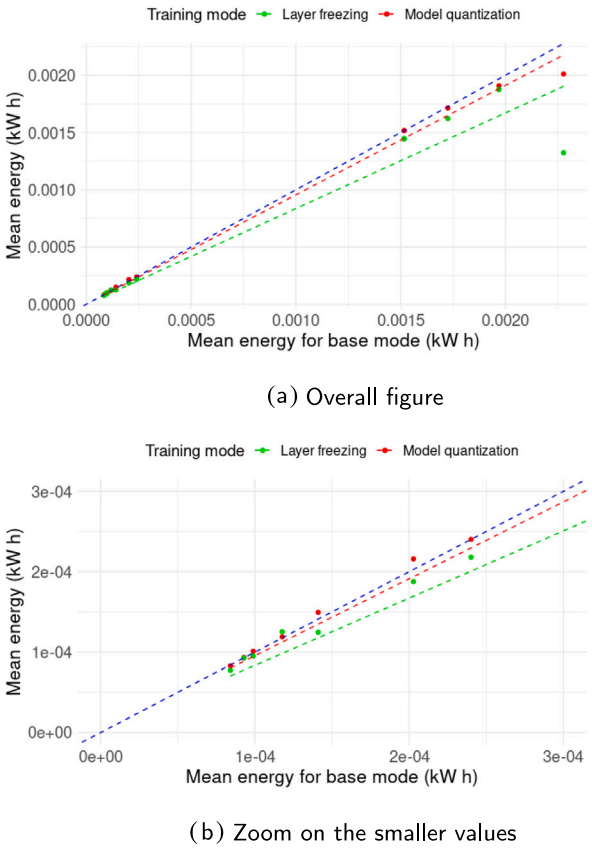


Fig. 12. Average energy consumed per epoch for base mode against freeze and quantization modes, with the identity line in blue and the regression lines for the 2 alternative modes, with the intercept fixed to 0. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5. Discussions

In this section we will analyze and interpret the findings of the study, its implications, the limitations that could threaten the validity of it, and the ways we mitigated the threats, paving the way for future works related to improving the energy efficiency of neural networks.

5.1. Implications

In this study, we empirically investigated the impact that layer freezing, model quantization, and early stopping can have when applied at different moments during the training phase. It is clear that there is plenty of room for energy savings in traditional DL training, without compromising accuracy or even improving it. Our study also provides a framework for reducing the environmental impact of AI model training through the intelligent application of the mentioned techniques layer freezing, model quantization, and early stopping. By empirically demonstrating the benefits of using past and future data for decision-making, we offer a method that not only maintains but can also improve model accuracy while significantly reducing energy consumption. This addresses the critical environmental impact of the growing carbon footprint of AI training by offering a practical, evidence-based approach that AI practitioners can adopt.

Early stopping was an already widespread technique utilized to reduce the time of training, as it was also well known that the rate of increase in validation accuracy slows down over time, or even declines due to overfitting later in the training. However, it is usually performed following heuristics defined prior the start of the training by using past data and is applied automatically. We showed the improvement of using past *and future* data with the help of simulation and *what if* analysis in order to take a more informed and accurate decision.

Similarly layer freezing was often applied on two phases, one of full training and one of fine tuning, with lengths defined *a priori* based on estimations. The ability of deciding when to perform it by using past and also future data has been proven to at times even improve the accuracy obtained with a base training while at the same time reducing energy consumption. In addition, layer freezing also reduces time and power requirements, making its application beneficial even from a deployment perspective that disregards green AI concepts.

Model quantization did not show any significant impact under our research setup. Nevertheless there are other setups to be tested, e.g. having a different number of trainable layers or training a different NN architecture. There are also plenty of other methods that could benefit from the possibility of performing them intelligently based on previously collected data that allows to perform predictions into the future. This could be extended not only to other training methods, but also to other phases of the DL deployment, such as hyperparameter search. In fact, hyperparameter search usually involves retraining the same model over and over by slightly tweaking some variables to find the best configuration in a trial-and-error fashion, with very few heuristics to finalize variables without actually testing them first.

As general conclusion, each one with their own characteristics, there are multiple methods that can be used in order to limit the environmental footprint of training DL models. We encourage AI practitioners to consider possible methods to do so during the design of their work, focusing on early stopping, the simplest of the methods and the one that reported bigger energy savings, with no drawbacks in accuracy if applied properly. We also recommend to include an auto-regressive prediction model on the performance with the available data, and enable it to decide the optimal moment to apply the design decisions.

With that purpose, we encourage AI researchers to continue on this line of investigation, by not only analyzing the effects of different energy saving methods but also the effects of the moment of intervention, and what makes certain moments optimal, in order to facilitate this task for practitioners. Those methods should be defined and included in the training process design prior to its start. The intelligent decisions on how to apply those methods are not possible without an energy profiling tool. Recent studies have shown the inaccuracies of current system-level profilers [71], and given the environmental concerns raised in the last years related to AI, including a reliable energy monitorization method into the most popular frameworks is an overdue task. For that reason, we call for the industry, especially the flagships development endeavours such as TensorFlow and PyTorch, to include reliable energy profiling tools, preferably with prediction or simulation capabilities.

Even though this work focuses on the computer vision datasets, any other energy-consuming AI field could apply the guidelines proposed in this study: Prior to starting the training, find appropriate methods to reduce the energy consumption of the utilized algorithms and investigate the effects they have on the energy reduction and model accuracy. During the execution, monitor constantly the energy and accuracy and enable a prediction model when possible. Use the predicted energy and accuracy to apply the best possible strategy to the algorithm in order to utilize the pareto optimal solution considering both energy efficiency and accuracy. The possible decisions do not have to limit to only the epoch at which the method is applied, but can also define other hyperparameters of such method, as the already mentioned number of layers to freeze or the numerical precision of the weights, related to the methods used in this work.

5.1.1. Energy consumed for this study

In order to comply with and promote the energy transparency defended by multiple green AI researchers, we want to report the energy consumption and carbon emissions caused by this study. As documented in the dataset `EAT-IT` reports, all the trainings and inferences performed to collect the data sum up to 3.24 kWh of energy consumed, according to the Nvidia-SMI measurements. The energy consumed is equivalent to approximately 3 h electric consumption in an average household in the USA. The documented energy consumed for this study underlines the energy efficiency of the computer where the training was performed, as collecting the data required 24.15 h compute time with an average of 12% of GPU utilization. The economical cost of this computation was of 368€, a low budget compared to typical works in AI, that makes this study easily replicable to other research groups with limited budget. We conclude that the carbon footprint of this work can be considered as relatively low, and we expect that it can be compensated by the potential savings in energy that can be achieved by applying the findings of this study in practice.

5.2. Threats to validity

As all works, this study has limitations that might have hindered the reliability of our findings. It is vital to acknowledge such threats to assert the strengths and weaknesses of the study design, and to address them with the intent of mitigating them. That way, we can strengthen the confidence in conclusions reached and point the direction onto future work to try to overcome such limitations. As recently pointed out, threats to validity and their mitigation should not be considered only as an afterthought, but should be actively dealt with during the research design [72]. As further clarified below, while reported towards the end of the study, threats were considered from the first research stages of this work.

The main limitation of the study was the amount of compute power at our disposition, limiting the amount of data we could create for our dataset `EAT-IT`. We tried to mitigate this threat by reducing the amount of redundant data with the design of a training tree, although we could still only manage to get one repetition per dataset on our budget. The most threatened analysis by this fact would be on the effect of different datasets on the measurements, and for that reason we kept it out of the focus of the study and did not report it, although we took it into account when analyzing the effect of the other independent variables to mitigate potential threats to conclusion validity. We opted for using 12 different datasets instead of repeating 12 times the experiment on the same dataset to encompass a wide variety of fields and tasks and strengthen the external validity. For the analysis on the effect of the other independent variables, it could be seen as having 12 repetitions of the same experiment, albeit this would be assuming no interaction of treatments between dataset and the other variables, which has not been tested. Still, each repetition of such experiment contains 9 training branches and 350 epochs in total, which we considered a considerable amount of data to guarantee internal validity, especially for the conclusions drawn on the energy consumption, as it has been shown to be temporally independent. In any case, further replication work should be performed by utilizing different datasets to strengthen the external validity of this study.

Other limitations on the study design caused by the computational budget were the number of models that we tested, the number of layers on the model that we trained, and the resolution of the input images. We mitigated this threat to external validity by referring to previous researches that proved that accuracy achieved in low resolution, fine tuning, or only one model can be used as a predictor for accuracy achieved in under other settings. Still, the conclusions of this work should be taken cautiously when applying them to other setups. We encourage the replication of the study in other settings, by expanding the building blocks of the study design, varying datasets and tasks, in order to further strengthen the external validity of this investigation.

Regarding the prediction model developed, we chose the models that suited the best the nature of the data we were observing and measuring, but there are still plenty of other prediction methods that could be used. This could lead to a potential model selection bias, which would suppose a threat to the statistical conclusion validity. We tried to minimize this threat by correctly justifying the decisions made on the design of the prediction models and how the selected methods were suitable to the data. We still tried to propose a variety of methods and test them all to get a representative sample of models, as proposing pooled OLS, fixed effects and random effects models to predict the accuracy. We also tested possibly preconceived ideas about the data, like the temporal independence of the energy consumption, to avoid confirmation bias in the model selection. Replication of the study could be performed testing different prediction methods to potentially achieve more precise predictions, which could lead to bigger savings in energy consumption.

6. Conclusions

In this work, we studied three methods to improve the energy efficiency of neural network training, namely, early stopping, layer freezing, and model quantization. We not only analyzed the effect of them in the final accuracy and energy consumption achieved by models, but also the variation of this effect depending on when it is performed during the training phase. On top of that, we studied how predictable this effect is, even when utilizing data collected by trainings a different dataset. For this purpose, we created a novel dataset containing information on the learning curves of a convolutional neural network trained on 12 different image datasets, including power measurements at the temporal granularity of one second. The analysis is performed with the intention to show the potential of including these results, or the ones from equivalent analysis, into a DL pipeline following the guidelines presented in Section 5. Estimating in advance the accuracy obtained and the energy needed for different training

modes and trying to optimize it can potentially lead to models more environmentally sustainable with marginal decreases of accuracy.

Using the data collected, we studied not only the effects that layer freezing, model quantization and early stopping have on the model accuracy and energy consumption, but how the timing of the application of these methods during the training process influences such impact. We showed that layer freezing affects accuracy at any point of the training, although for it to reduce the energy consumed it must be applied before epoch 27 out of 50. Similarly, early stopping can reduce drastically the energy consumption proportional to the amount of epochs saved, and does not affect accuracy after epoch 17 out of 50. We did not find a significant difference on either accuracy or energy consumption when applying model quantization. Although those findings might vary depending on the setup used, they give an intuition on the behavior of these methods.

An essential part of the framework is to predict future accuracy and energy consumed whether the different methods are applied or not during all the training. We showed that accuracy can be treated as panel data and predicted by fitting a fixed effects model on it, reaching more than 90% of precision in the predictions. On the other hand, energy measures can be treated as temporally independent and predicted by reporting its average, reaching more than 85% of precision. The trade-off score defined as the ratio between accuracy and energy consumed can be predicted by using the previous models reaching more than 90% of precision. This score can be maximized by predicting it for future epochs for different TMs, estimating the point where its derivative equals to −1 for each TM, choosing the one with the highest score at that point to continue the training and stopping at that point. We also showed that using this method can reduce the energy consumed to less than half and have marginal increases in accuracy with respect to the baseline.

Since the inception of the work, we designed the study on building blocks composed by different elements that can be changed or expanded, in order to enable software reuse of our replication package in possible future work. More datasets, models, training decisions, and measuring instruments can be tested to extend the findings and capabilities of the interactive training framework. We discussed the effect on the elements chosen for those building blocks on this study in the previous section. Another direction of work is in developing a system to fully include monitoring, prediction and intelligent decisions into the DL development process, to be capable to use these findings to reduce environmental impact of current DL projects.

CRediT authorship contribution statement

Álvaro Domingo Reguero: Writing – original draft, Visualization, Methodology, Investigation, Data curation, Conceptualization. **Silverio Martínez-Fernández:** Writing – review & editing, Supervision, Resources, Project administration, Conceptualization. **Roberto Verdecchia:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

For everyone interested in reproducing or building upon our study, we make the entirety of the data collected, the source code developed for the study, and the scripts utilized to analyze the results publicly available at the following Zenodo repository: https://doi.org/10.5281/zenodo.13371442.

Acknowledgments

This work has been supported by the GAISSA project (TED2021-130923B-I00, which is funded by MCIN/AEI/ 10.13039/501100011033 and by the European Union "NextGenerationEU"/PRTR). The authors also thank the "Beatriz Galindo" Spanish Program (BEAGAL18/00064), and the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE0000001 - program "RESTART"). We thank Santiago del Rey for having reviewed our work and for being supportive and useful to enhance it.

Appendix A. Visualization of data collected

In Fig. 13 we provide some visualizations of the novel dataset created. Each visualization is divided in 12 for each of the 12 original datasets, represented individually, and codifies the other 3 independent variables as follows: The training mode is codified by color (blue for base mode, green for layer freezing mode, blue for model quantization mode), the epoch is codified in the x axis, and the epoch of intervention is codified as where the branch of the data point starts. That way, the 12 training trees are reconstructed, and showing in the y axes one of the dependent variables of interest. The energy values are the ones measured by Nvidia-SMI. It is necessary to note that in a printed version those graphics could not look optimal, but they are properly vectorized so that they can be seen in high resolution on a digital version zoomed in.

Appendix B. Distribution tests for energy measures

In Fig. 14 we provide figures related with the tests for the distribution of the energy measures, as stated in Section 4. Apart from just testing temporal independence, those figures also account for tests of normality. The hypothesis of the data following a normal distribution was rejected, although that does not suppose a problem to the rest of the study. Again, blue for base mode, green for layer freezing mode, blue for model quantization mode.

Appendix C. Visualization of the predictions

In Fig. 15 we provide a visualization for the three auto-regressive prediction models developed. Simulating a use case scenario, when wanting to train on a new dataset, the training tree would not be computed entirely, just one branch of it. For that reason, here we show only one branch in every plot, which also helps the visualization. We only show training branches with an EoI of 20 or 30 to favor visualization as the interventions are made close to the middle of the training. We show branches diverging to both LF and MQ mode. For each combination of EoI and training mode, we show the corresponding branches of two datasets selected at random.

Each one of the plots follows the same methodology as explained in Section 4.2, trying to simulate the real use case scenario: For predictions over one dataset, the training data points available before starting are the ones corresponding to the other 11 datasets. For each epoch of the training branch to predict, the data points from that branch of previous epochs to the one being predicted are also available for the model to elaborate its predictions. Dots represent ground truth and lines represent the prediction for that epoch. Note that there is no prediction for the first epochs of each branch, as not enough data has been collected to perform a prediction.

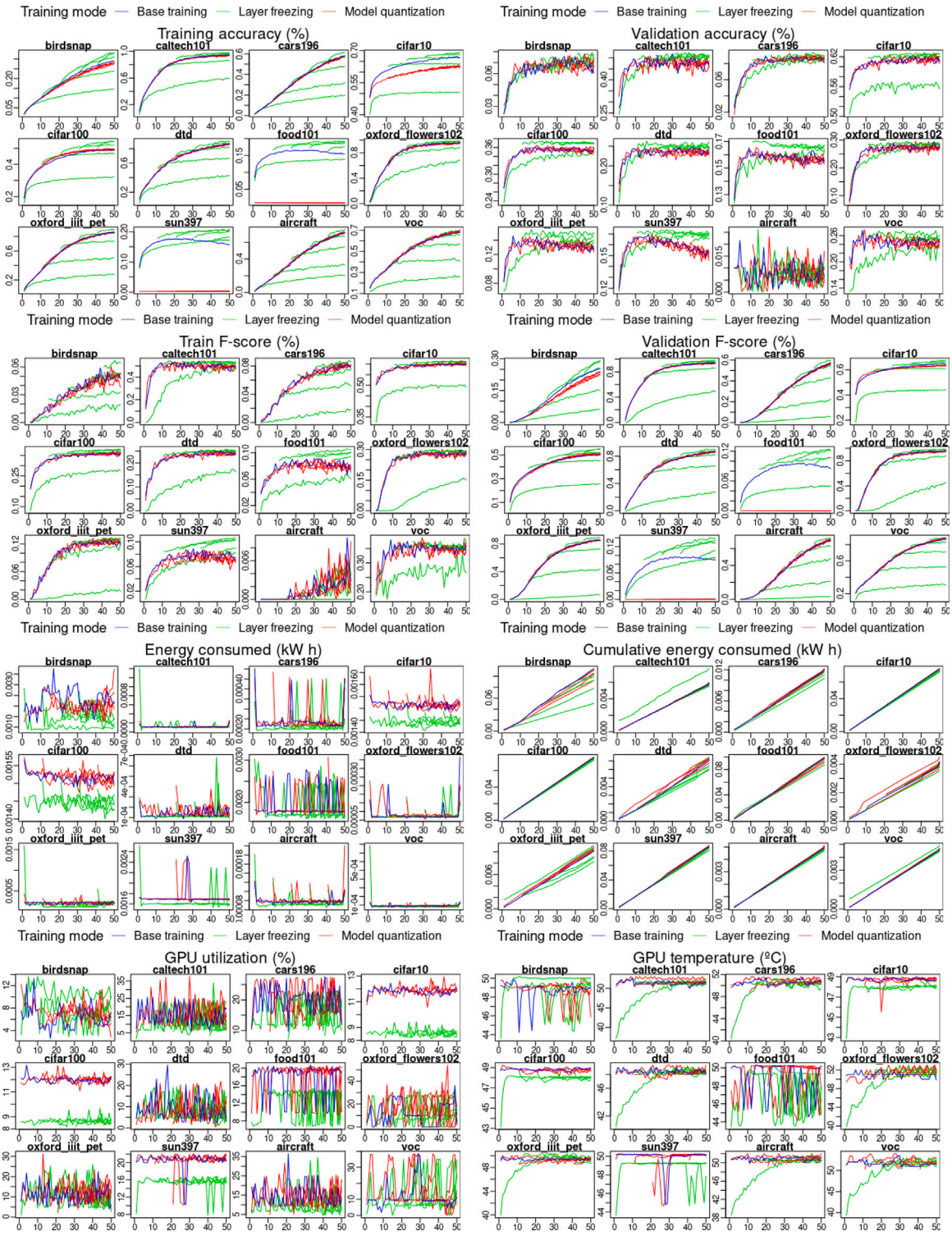
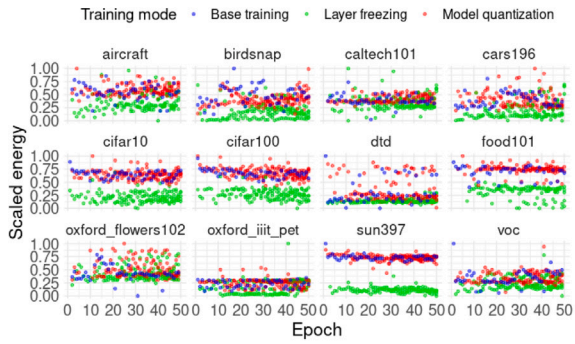
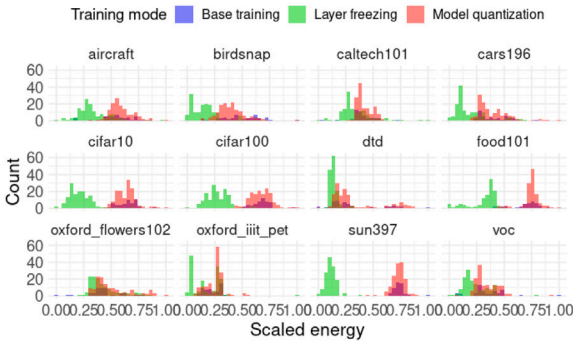


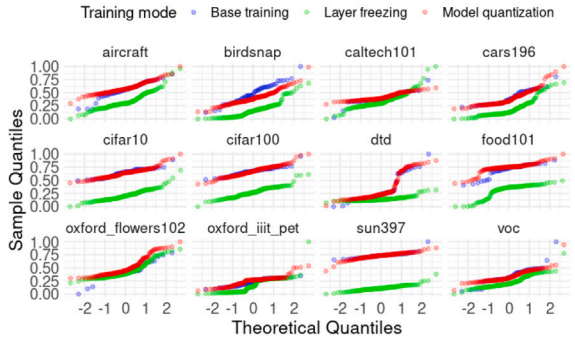
Fig. 13. Visualization of the training tree for different dependent variables. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



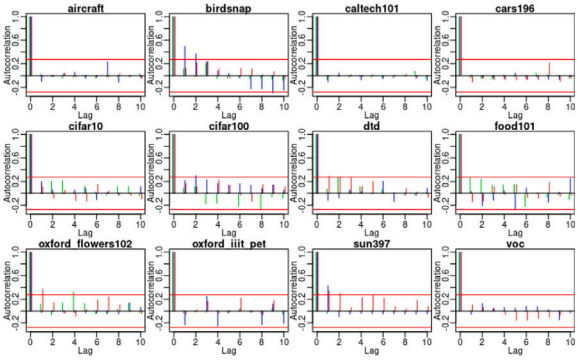
(a) Scaled energy consumed per epoch without outliers for each dataset



(b) Histograms of the scaled energy consumed per epoch without outliers for each dataset



(c) QQplots of the scaled energy consumed without outliers for each dataset



(d) P-values of the Shapiro-Wilk test for every dataset

Fig. 14. Visualization of various tests regarding the distribution of the energy consumed per epoch. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

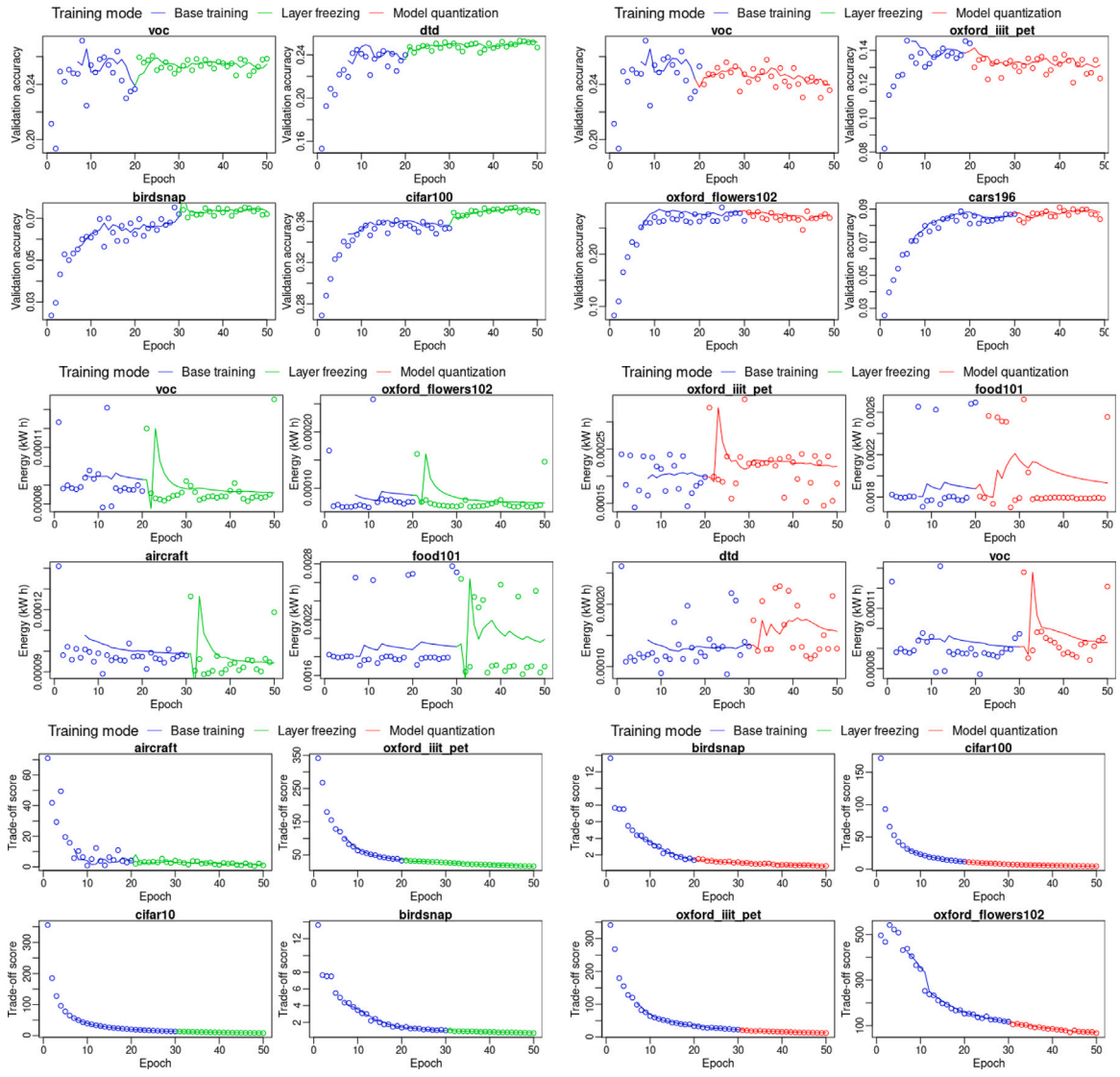


Fig. 15. Visualization of the predictions of the three models on various training branches.

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: F. Pereira, C. Burges, L. Bottou, K. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 25, Curran Associates, Inc., 2012, URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[2] D. Zhang, S. Mishra, E. Brynjolfsson, J. Etchemendy, D. Ganguli, B. Grosz, T. Lyons, J. Manyika, J.C. Niebles, M. Sellitto, Y. Shoham, J. Clark, R. Perrault, The AI index 2021 annual report, 2021, arXiv:2103.06312.

[3] D. Zhang, N. Maslej, E. Brynjolfsson, J. Etchemendy, T. Lyons, J. Manyika, H. Ngo, J.C. Niebles, M. Sellitto, E. Sakhaee, Y. Shoham, J. Clark, R. Perrault, The AI index 2022 annual report, 2022, arXiv:2205.03468.

[4] J. Hestness, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M.M.A. Patwary, Y. Yang, Y. Zhou, Deep learning scaling is predictable, empirically, 2017, arXiv:1712.00409.

[5] D. Amodei, D. Hernandez, AI and compute, 2018, URL: https://openai.com/research/ai-and-compute.

[6] N.C. Thompson, K. Greenewald, K. Lee, G.F. Manso, The computational limits of deep learning, 2022, arXiv:2007.05558.

[7] A.J. Lohn, M. Musser, AI and Compute: How Much Longer Can Computing Power Drive Artificial Intelligence Progress?, Technical Report, Center for Security and Emerging Technology, 2022, http://dx.doi.org/10.51593/2021CA009.

[8] C.-J. Wu, R. Raghavendra, U. Gupta, B. Acun, N. Ardalani, K. Maeng, G. Chang, F. Aga, J. Huang, C. Bai, M. Gschwind, A. Gupta, M. Ott, A. Melnikov, S. Candido, D. Brooks, G. Chauhan, B. Lee, H.-H. Lee, B. Akyildiz, M. Balandat, J. Spisak, R. Jain, M. Rabbat, K. Hazelwood, Sustainable AI: Environmental implications, challenges and opportunities, in: D. Marculescu, Y. Chi, C. Wu (Eds.), Proceedings of Machine Learning and Systems, Vol. 4, 2022, pp. 795–813, URL: https://proceedings.mlsys.org/paper_files/paper/2022/file/462211f67c7d858f663355eff93b745e-Paper.pdf.

[9] R. Verdecchia, J. Sallou, L. Cruz, A systematic review of green AI, in: Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Wiley Online Library, 2023, e1507.

[10] A.S. Luccioni, S. Viguier, A.-L. Ligozat, Estimating the carbon footprint of BLOOM, a 176B parameter language model, 2022, arXiv:2211.02001.

[11] L. Prechelt, Early stopping-but when? in: Neural Networks: Tricks of the Trade, Springer, 2002, pp. 55–69.

[12] R. Caruana, S. Lawrence, C. Giles, Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping, Adv. Neural Inf. Process. Syst. 13 (2000).

[13] Y. Yao, L. Rosasco, A. Caponnetto, On early stopping in gradient descent learning, Constr. Approx. 26 (2) (2007) 289–315.

[14] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? Adv. Neural Inf. Process. Syst. 27 (2014).

[15] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy layer-wise training of deep networks, Adv. Neural Inf. Process. Syst. 19 (2006).

[16] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.

[17] S. Han, H. Mao, W.J. Dally, Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, in: International Conference on Learning Representations, 2015.

[18] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, D. Kalenichenko, Quantization and training of neural networks for efficient integer-arithmetic-only inference, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2704–2713.

[19] M. Courbariaux, Y. Bengio, J.-P. David, Binaryconnect: Training deep neural networks with binary weights during propagations, Adv. Neural Inf. Process. Syst. 28 (2015).

[20] H. Järvenpää, P. Lago, J. Bogner, G. Lewis, H. Muccini, I. Ozkaya, A synthesis of green architectural tactics for ML-enabled systems, 2023, arXiv:2312.09610.

[21] C. Becker, R. Chitchyan, L. Duboc, S. Easterbrook, B. Penzenstadler, N. Seyff, C.C. Venters, Sustainability design and software: The karlskrona manifesto, in: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Vol. 2, IEEE, 2015, pp. 467–476.

[22] R. Verdecchia, P. Lago, C. Ebert, C. De Vries, Green IT and green software, IEEE Softw. 38 (6) (2021) 7–15.

[23] C. Calero, M. Piattini, Green in Software Engineering, vol. 3, Springer, 2015.

[24] C. Calero, M.Á. Moraga, F. García, Software, sustainability, and UN sustainable development goals, IT Prof. 24 (1) (2022) 41–48.

[25] R. Schwartz, J. Dodge, N.A. Smith, O. Etzioni, Green AI, Commun. ACM 63 (12) (2020) 54–63, http://dx.doi.org/10.1145/3381831.

[26] E. Strubell, A. Ganesh, A. McCallum, Energy and policy considerations for deep learning in NLP, 2019, arXiv:1906.02243.

[27] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, K. Murphy, Speed/accuracy trade-offs for modern convolutional object detectors, 2017, arXiv:1611.10012.

[28] C. Sun, A. Shrivastava, S. Singh, A. Gupta, Revisiting unreasonable effectiveness of data in deep learning era, 2017, arXiv:1707.02968.

[29] J. Dodge, S. Gururangan, D. Card, R. Schwartz, N.A. Smith, Show your work: Improved reporting of experimental results, 2019, arXiv:1909.03004.

[30] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, 2016, arXiv:1506.02640.

[31] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv:1704.04861.

[32] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi, XNOR-Net: ImageNet classification using binary convolutional neural networks, 2016, arXiv:1603.05279.

[33] A. Goel, C. Tung, Y.-H. Lu, G.K. Thiruvathukal, A survey of methods for low-power deep learning and computer vision, 2020, arXiv:2003.11066.

[34] S. Han, J. Pool, J. Tran, W. Dally, Learning both weights and connections for efficient neural network, in: C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 28, Curran Associates, Inc., 2015, URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf.

[35] Q. Hu, Y. Guo, M. Cordy, X. Xie, W. Ma, M. Papadakis, Y. Le Traon, Towards understanding model quantization for reliable deep neural network deployment, in: 2023 IEEE/ACM 2nd International Conference on AI Engineering–Software Engineering for AI , CAIN, IEEE, 2023, pp. 56–67.

[36] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (3) (2009) 455–500, http://dx.doi.org/10.1137/07070111X, arXiv:https://doi.org/10.1137/07070111X.

[37] T. Yarally, L. Cruz, D. Feitosa, J. Sallou, A. Van Deursen, Uncovering energy-efficient practices in deep learning training: Preliminary steps towards green ai, in: 2023 IEEE/ACM 2nd International Conference on AI Engineering–Software Engineering for AI, CAIN, IEEE, 2023, pp. 25–36.

[38] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, 2015, arXiv preprint arXiv:1503.02531.

[39] B. Li, X. Jiang, D. Bai, Y. Zhang, N. Zheng, X. Dong, L. Liu, Y. Yang, D. Li, Full-cycle energy consumption benchmark for low-carbon computer vision, 2021, arXiv:2108.13465.

[40] T.-J. Yang, Y.-H. Chen, V. Sze, Designing energy-efficient convolutional neural networks using energy-aware pruning, 2017, arXiv:1611.05128.

[41] Y. Wang, Z. Jiang, X. Chen, P. Xu, Y. Zhao, Y. Lin, Z. Wang, E2-train: Training state-of-the-art cnns with over 80% energy savings, Adv. Neural Inf. Process. Syst. 32 (2019).

[42] W.A. Hanafy, T. Molom-Ochir, R. Shenoy, Design considerations for energy-efficient inference on edge devices, in: Proceedings of the Twelfth ACM International Conference on Future Energy Systems, 2021, pp. 302–308.

[43] L.H.P. de Chavannes, M.G.K. Kongsbak, T. Rantzau, L. Derczynski, Hyperparameter power impact in transformer language model training, in: Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, 2021, pp. 96–118.

[44] B. Zhang, A. Davoodi, Y.H. Hu, Exploring energy and accuracy tradeoff in structure simplification of trained deep neural networks, IEEE J. Emerg. Sel. Top. Circuits Syst. 8 (4) (2018) 836–848.

[45] A. Lacoste, A. Luccioni, V. Schmidt, T. Dandres, Quantifying the carbon emissions of machine learning, 2019, arXiv:1910.09700.

[46] P. Henderson, J. Hu, J. Romoff, E. Brunskill, D. Jurafsky, J. Pineau, Towards the systematic reporting of the energy and carbon footprints of machine learning, 2022, arXiv:2002.05651.

[47] L. Lannelongue, J. Grealey, M. Inouye, Green algorithms: Quantifying the carbon footprint of computation, 2020, arXiv:2007.07610.

[48] E. Cai, D.-C. Juan, D. Stamoulis, D. Marculescu, NeuralPower: Predict and deploy energy-efficient convolutional neural networks, 2017, arXiv:1710.05420.

[49] C.F. Rodrigues, G. Riley, M. Luján, SyNERGY: An energy measurement and prediction framework for convolutional neural networks on jetson TX1, in: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA, The Steering Committee of The World Congress in Computer Science, Computer . . . , 2018, pp. 375–382.

[50] Q. Cao, Y.K. Lal, H. Trivedi, A. Balasubramanian, N. Balasubramanian, IrEne: Interpretable energy prediction for transformers, 2021, arXiv preprint arXiv:2106.01199.

[51] Y. Wang, R. Ge, S. Qiu, Energy-aware DNN graph optimization, 2020, arXiv preprint arXiv:2005.05837.

[52] L.F.W. Anthony, B. Kanding, R. Selvan, Carbontracker: Tracking and predicting the carbon footprint of training deep learning models, 2020, arXiv:2007.03051.

[53] V.R. Basili, G. Caldiera, D.H. Rombach, The Goal Question Metric Approach, vol. I, John Wiley & Sons, 1994.

[54] S. Alyamkin, M. Ardi, A.C. Berg, A. Brighton, B. Chen, Y. Chen, H.-P. Cheng, Z. Fan, C. Feng, B. Fu, K. Gauen, A. Goel, A. Goncharenko, X. Guo, S. Ha, A. Howard, X. Hu, Y. Huang, D. Kang, J. Kim, J.G. Ko, A. Kondratyev, J. Lee, S. Lee, S. Lee, Z. Li, Z. Liang, J. Liu, X. Liu, Y. Lu, Y.-H. Lu, D. Malik, H.H. Nguyen, E. Park, D. Repin, L. Shen, T. Sheng, F. Sun, D. Svitov, G.K. Thiruvathukal, B. Zhang, J. Zhang, X. Zhang, S. Zhuo, Low-power computer vision: Status, challenges, opportunities, 2019, http://dx.doi.org/10.48550/ARXIV.1904.07714, URL: https://arxiv.org/abs/1904.07714.

[55] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer Science & Business Media, 2012.

[56] S. Kornblith, J. Shlens, Q.V. Le, Do better ImageNet models transfer better?, 2018, http://dx.doi.org/10.48550/ARXIV.1805.08974, URL: https://arxiv.org/abs/1805.08974.

[57] M. Koziarski, B. Cyganek, Impact of low resolution on image recognition with deep neural networks: An experimental study, Int. J. Appl. Math. Comput. Sci. 28 (4) (2018) 735–744, http://dx.doi.org/10.2478/amcs-2018-0056.

[58] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, http://dx.doi.org/10.48550/ARXIV.1409.1556, URL: https://arxiv.org/abs/1409.1556.

[59] N. Corporation, Nvidia-smi - NVIDIA system management interface program, 2016, URL: https://developer.download.nvidia.com/compute/DCGM/docs/nvidia-smi-367.38.pdf. Version 367.38.

[60] L. Cruz, Tools to measure software energy consumption from your computer, 2021, http://dx.doi.org/10.6084/m9.figshare.19145549.v1, Blog post, http://luiscruz.github.io/2021/07/20/measuring-energy.html.

[61] Y. Xu, S. Martínez-Fernández, M. Martinez, X. Franch, Energy efficiency of training neural network architectures: An empirical study, 2023, arXiv:2302.00967, URL: https://arxiv.org/abs/2302.00967.

[62] D. Cousineau, Confidence intervals in within-subject designs: A simpler solution to Loftus and Masson's method, Tutor. Quant. Methods Psychol. 1 (1) (2005) 42–45, http://dx.doi.org/10.20982/tqmp.01.1.p042, URL: http://www.tqmp.org/RegularArticles/vol01-1/p042/p042.pdf.

[63] W.H. Kruskal, W.A. Wallis, Use of ranks in one-criterion variance analysis, J. Amer. Statist. Assoc. 47 (260) (1952) 583–621, http://dx.doi.org/10.1080/01621459.1952.10483441, arXiv:https://www.tandfonline.com/doi/pdf/10.1080/01621459.1952.10483441, URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1952.10483441.

[64] F. Wilcoxon, Individual comparisons by ranking methods, in: S. Kotz, N.L. Johnson (Eds.), Breakthroughs in Statistics: Methodology and Distribution, Springer New York, New York, NY, 1992, pp. 196–202, http://dx.doi.org/10.1007/978-1-4612-4380-9_16.

[65] M.G. Kendal, A new measure of rank correlation, Biometrika 30 (1–2) (1938) 81–93, http://dx.doi.org/10.1093/biomet/30.1-2.81, arXiv:https://academic.oup.com/biomet/article-pdf/30/1-2/81/423380/30-1-2-81.pdf.

[66] J.M. Wooldridge, Econometric Analysis of Cross Section and Panel Data, MIT Press, 2010.

[67] A. Wald, Tests of statistical hypotheses concerning several parameters when the number of observations is large, Trans. Amer. Math. Soc. 54 (3) (1943) 426–482, URL: http://www.jstor.org/stable/1990256.

[68] J.A. Hausman, Specification tests in econometrics, Econometrica 46 (6) (1978) 1251–1271, URL: http://www.jstor.org/stable/1913827.

[69] C.E. Bonferroni, Teoria statistica delle classi e calcolo delle probabilità / Carlo E. Bonferroni, Pubblicazioni Del R. Istituto Superiore Di Scienze Economiche E Commerciali Di Firenze 8, Seeber, Firenze, 1936.

[70] G.M. Ljung, G.E.P. Box, On a measure of lack of fit in time series models, Biometrika 65 (2) (1978) 297–303, http://dx.doi.org/10.1093/biomet/65.2.297, arXiv:https://academic.oup.com/biomet/article-pdf/65/2/297/649058/65-2-297.pdf.

[71] Q. Cao, A. Balasubramanian, N. Balasubramanian, Towards accurate and reliable energy measurement of NLP models, 2020, arXiv:2010.05248.

[72] R. Verdecchia, E. Engström, P. Lago, P. Runeson, Q. Song, Threats to validity in software engineering research: A critical reflection, Inf. Softw. Technol. 164 (2023) 107329.