ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

Making Convolutional Neural Networks Energy-Efficient: An Introduction

Noémie Draguet, Benoît Frénay*

University of Namur - NaDI - PReCISE - HuMaLearn
Rue Grangagnage, 21, 5000 Namur - Belgium

Abstract. As convolutional neural networks (CNNs) have become mainstream for object recognition and image classification, the environmental impact caused by their high energy consumption (EC) is non negligible. This paper examines techniques that have the ability to reduce the EC of CNNs. It also highlights the inconsistency of metrics that are used for estimating or measuring EC, which reduces the comparability of these techniques. This review aims to shed light on the current situation and to provide a basis for future research in green machine learning.

1 Introduction

As machine learning and artificial intelligence become increasingly popular, these technologies start having growing consequences on the environment due to their substantial energy consumption (EC). Among various models are convolutional neural networks (CNNs) that perform well in object recognition and image classification. However, they usually require a lot of computational power to achieve satisfying results given their high dimensional inputs, their large size and their substantial number of parameters. For example, 0.19 joules are necessary for making an inference with ResNet50 on Nvidia RTX 2080 Ti [1]. So far, the literature has focused on improving the accuracy of those models, or on decreasing their size [2], inference time [3] and EC for performance or portability purposes [4]. Furthermore, the decrease in energy resulting from the proposed methods is usually not measured with a same metric, which makes the comparison of different techniques difficult and the literature inconsistent regarding EC. This paper gives an overview of existing methods that have for purpose or side effect to reduce the EC of CNNs, which can help introducing junior researchers to this subject and might open doors for future research in this area.

Section 2 discusses metrics to measure or approximate the EC of a model and gives recommendations about their use. Section 3 introduces families of methods that reduce the EC of CNNs and shows the results obtained with these methods. Finally, Section 4 concludes and gives perspectives for future research.

2 Metrics for Energy Consumption

Before introducing techniques that can reduce the EC of CNNs, it is essential to first discuss how to measure it. Indeed, works in the literature use multiple

*Supported by SPWR under grant n°2010235 - ARIAC by DIGITALWALLONIA4.AI.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

metrics as proxies for EC. EC can be measured or inferred in three different ways: from hardware measurements, with proxies or with a model.

Consumed energy is sometimes inferred from actual hardware measurements. However, most of the time, energy is approximated by proxies such as the number of parameters of the model, the average number of operations (OPS) per input or the runtime for inference. These approximations do not always reflect the actual energy that was consumed [4]. For example, using scalable-effort classifiers results in a 2.79x decrease in average OPS per input, but only in a 2.3x and 1.5x decrease in EC and runtime, respectively [5]. The difference between these measures is mostly due to implementation overheads. Measuring the actual EC of a CNN is complicated, which is why Cai et al. [6] present an approach called *NeuralPower* that predicts the EC of different CNN architectures without needing to train the models. A model is trained to predict the EC, power and runtime of each layer of a CNN, based on its architecture, and the results are then generalized to the network level. Other models or frameworks, such as the Synopsys Power compiler, also exist to evaluate EC. In conclusion, several metrics or methods can be used to measure EC, and their choice can influence the conclusions that are drawn. For measuring actual EC and its impact on the environment, it is preferable to directly use hardware measurements.

3 Methods for Reducing Energy Consumption

This section presents techniques that have for purpose or side effect to reduce the EC of CNNs, along with their results. Most of the presented methods reduce EC during inference and do not specifically address training. This is most likely because reducing EC for training is even more challenging than for inference. The ignored additional EC induced by a potentially more complex training might reduce the benefits of the presented techniques. This is however out of scope of the present work, since training EC is not discussed in the reviewed papers.

3.1 Pruning and network sparsification techniques

Pruning methods and network sparsification techniques are the most explored in the literature. They simplify the network by removing or reducing some of its components (weights, neurons, filters or number of layers). The first paper introducing pruning "Optimal brain damage" [7] dates back to 1989 and removed weights from the artificial neural network using second-derivative information. At that time, the focus of pruning was not to reduce EC and this aspect was therefore not studied. The objective was rather to achieve better generalization, to reduce the need of training examples and to improve the learning speed.

A few years later, Iandola et al. [2] demonstrated how to use sparsification techniques to reduce the number of parameters of CNNs. The paper focuses on reducing the size of a given CNN architecture without loosing accuracy. Their SqueezeNet architecture achieves AlexNet-level accuracy on ImageNet with 50x less parameters by using three distinct strategies: (1) "replace 3x3 filters with 1x1 filters", (2) "decrease the number of input channels to 3x3 filters" and (3)

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

"downsample late in the network so that convolution layers have large activation maps". The two first strategies aim to reduce the number of parameters in the network, while the third one helps to maximize accuracy under the constraint of a limited number of parameters. While demonstrating significant reduction of the number of network parameters, the paper does not mention EC.

Another example of pruning by Wen et al. [8] shows an average speedup of 5.1x and 3.1x for convolutional layer computations in the AlexNet architecture on CPUs and GPUs, respectively. The method obtains a more compact architecture from a large model, by adjusting filters, channels, filter shapes and depth of the network. This is achieved by leveraging group Lasso regularization during training. This study focuses on computational speed rather than on EC in itself, which makes it slightly less relevant for the present work on EC reduction.

While the three aforementioned papers successfully reduce the size of the model, this does not necessarily imply a significant decrease in EC, as previously mentioned. Yang et al. [4] introduce energy-aware pruning, which consists in pruning the weights layer by layer while starting with the layer that consumes the most energy. The first step is to determine which layers of the network require the most energy. This is determined by a framework that can estimate the energy consumption of a CNN for inference, based on two components: computation and data movement. The energy of each multiply-accumulate operation and each memory access are extrapolated from actual hardware measurements, which is more relevant compared to metrics that only take into account the size of the network or the number of operations executed. Layers with higher EC are given priority as the pruning process becomes progressively more challenging. Following the established order, each layer is pruned and its weights are locally fine-tuned. After all the layers have gone through the process, the network is globally fine-tuned using back-propagation. The method of Yang et al. [4] reduces the EC (as estimated by the above framework) of AlexNet and GoogleNet by respectively 3.7x and 1.6x with less than 1% top-5 accuracy loss.

3.2 Reduced precision networks

Another way to decrease the EC of CNNs is to reduce the precision of their components (representing them with fewer bits), although this reduction remains mostly theoretical, as current commercial GPUs are only able to work with 16 or 32 bit arithmetic. It is rather useful to store models more efficiently. For the sake of completeness and because reduced precision has been used in a high number of works, two examples of applications are provided in this section.

First, Hubara et al. [9] present a method to train Quantized Neural Networks (QNNs). In QNNs, weights and activations are represented using only a few bits, which simplifies operations and aims to reduce power consumption. The quantization process can be applied to both weights and activations or to only one of these components, and the quantized components can be used either during both training and testing phases or only during testing. For the MNIST, CIFAR-10, SVHN and ImageNet datasets, the accuracy of QNNs approaches the accuracy without quantization. Even though the exact impact of this technique

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

on EC is not mentioned in [9], the reduction in memory access cost due to the smaller memory size is expected to reduce EC with the appropriate hardware.

Second, Courbariaux et al. [10] introduce a method called BinaryConnect. This technique is not directly aimed at reducing EC but rather focuses on increasing the computational speed of CNNs. Binary weights are used instead of real-valued weights for the training to simplify the computing operations. The performance of BinaryConnect in accuracy is slightly higher than the performance of CNNs using real-valued weights for the MNIST, CIFAR-10 or SVHN datasets, as the reduced precision acts as a regularizer for these tasks.

Reduced precision networks must be used with appropriate hardware that enables low-precision values to not loose its benefits. In this vein, half-precision floats can be leveraged by PyTorch using *automatic mixed precision*.

3.3 Conditional computation

Conditional computation leverages the fact that some inputs are easier to classify than others. It consists in learning policies that either determine when an input should exit the network or which network is the most appropriate.

Venkataramani et al. [5] introduce the notion of "scalable-effort classifiers", which consist of an ensemble of classifiers of varying complexities. The objective is to use simple classifiers for inputs that are easy to classify, and to use more complex ones when it is not the case. The input progresses through several stages, each composed of two biased classifiers designed to detect a single class more effectively. A stage is only added to the system if the cost it incurs is smaller than if all instances had proceeded to the next stage. After each stage, a decision is made about the termination of the classification process. Even though this work does not specifically address CNNs, it focuses on energy efficiency by taking into account the average number of OPS, the energy and the runtime. Results differ across datasets but overall, they show a 2.79x decrease in average OPS per input, which leads to a 2.3x and 1.5x reduction in EC (from hardware measures) and runtime, respectively.

Panda et al. [11] use conditional deep learning (CDL) to design energy-efficient CNNs. Linear classifiers are trained to decide whether the classification process should continue or be terminated after a convolutional layer. A classifier is added after a convolutional layer only if it improves the overall efficiency of the model, taking into account the EC of additional computations performed by this classifier. The constructed linear classifiers leverage convolutional layer features to determine the difficulty of inputs and decide if the next layer should be activated. The linear classifier at each stage determines a class label for the input and associates a confidence value to it. A threshold (*activation value*) is set and the classification process is terminated if the confidence value is above this activation value for exactly one class label. Otherwise, the input is deemed as too difficult and moves on to the next stage. This method shows a 1.91x decrease in average number of OPS for MNIST, corresponding to a 1.84x reduction in EC (estimated with Synopsys Power). This is tested on two different CNN architectures, and shows bigger benefits when used with the most complex one.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

Bolukbasi et al. [3] show two different techniques of conditional computing. The first one is quite similar to Panda et al. [11] and trains a decision function (or policy) that determines whether an input should exit the classification process or not. The second method by Bolukbasi et al. [3] organizes a set of pre-trained CNNs in an acyclic graph and trains an exit policy (or decision function) at each node of the graph. These CNNs present different levels of accuracy and computational time. The input is first evaluated by the simplest CNN (with a smaller computational time), and the policy determines whether the process should terminate after this evaluation or the instance has to be evaluated by a more complex network. The process goes on until a CNN yields a satisfactory output. For the ImageNet 2012 dataset and a variety of network architectures, Bolukbasi et al. obtain a speedup of up to 2.8x compared to the performance of the initial architecture, with less than 1% top-5 accuracy loss.

Stamoulis et al. [12] simultaneously address two issues encountered when designing adaptive systems of CNNs: network selection and network design. When building adaptive systems of CNNs (as in the previous example), the assembled CNNs are usually treated as black boxes with a predefined architecture. The goal of Stamoulis et al. is to optimize the hyper-parameters of each CNN with regard to energy, accuracy and communication constraints. This reduces the EC for image classification on a mobile device by up to 6x compared to adaptive systems of CNNs that consider them as black boxes with a fixed architecture.

3.4 Spatially adaptive networks

Spatially adaptive networks are less common and are explored to a smaller extent for reducing the EC of neural networks. This method is similar to conditional computing in many ways. Bengio et al. [13] present this technique, which consists in activating only some parts of the network at run-time. To do so, a policy is learned for each layer of the network, using reinforcement learning. This policy is input-dependent and determines the probability of activation for each node within the layer. The authors' motivation is mainly to accelerate computations and they show speed-ups of 5x to 10x on CPUs and 2x to 4x on GPUs. Similarly, for example, Rashid et al. [14] propose an adaptive CNN for human activity recognition (AHAR). AHAR uses an output block predictor to determine which part of a baseline CNN architecture must be used for a given input. Results show that AHAR leads to a 1.12x decrease in energy consumption (as measured by the EFM32 Giant Gecko microcontroller) compared to the baseline CNN.

3.5 Hardware optimizations

A last technique that can help reducing the EC of CNNs is hardware optimization, orthogonal to the above software optimizations. Hardware optimizations can therefore be combined with other techniques, such as reduced precision networks [9] or pruning [8]. Liu et al. [15] present a low-power hardware architecture where computations are performed in a column-wise manner. This aims to reduce hardware resources and power consumption. Indeed, power is reduced for LeNet, AlexNet and VGG16 by up to 8.45%, 49.41% and 50.64% respectively.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

4 Conclusion

A variety of existing techniques can reduce the EC of CNNs, including but not limited to pruning and network sparsification techniques, reduced precision networks, conditional computation, spatially adaptive networks and hardware optimizations. However, so far, papers have not been consistent about the way they measure this decrease in EC. Furthermore, some metrics might vary according to the hardware on which the model runs (OPS or runtime for instance). Results may also vary according to the allowed loss in accuracy, the task at hand and the baseline architecture. One of the biggest axes that can be suggested for future research is to reimplement all these methods and to benchmark them using an identical hardware equipment, baseline architecture, task, maximum accuracy loss and metric for energy. In the current situation, it is difficult to evaluate which methods are the most promising. Ideally, hardware measurements should be preferred to measure the actual EC (instead of proxies). Secondly, these techniques are primarily focused on reducing the evaluation time or the size of CNNs, but not on decreasing their EC as such. Therefore, it would be judicious to leverage these existing techniques and to create new architectures with a focus on EC, considering the growing environmental concerns. Finally, it is to note that the presented techniques usually focus on the testing phase, but that new techniques could aim to reduce the EC of the training phase as well.

References

[1] Radosvet Desislavov, Fernando Martínez-Plumed, and José Hernández-Orallo. Trends in ai inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustainable Computing: Informatics and Systems*, 38:100857, 2023.

[2] Forrest N Iandola, Song Han, Matthew W Moskewicz, et al. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size. *arXiv:1602.07360*, 2016.

[3] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, et al. Adaptive neural networks for efficient inference. In *ICML*, volume 70, pages 527–536, 2017.

[4] Tien-Ju Yang, Yu-Hsin Chen, and Vivienne Sze. Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning. In *CVPR*, pages 6071–6079, 2017.

[5] Swagath Venkataramani, Anand Raghunathan, Jie Liu, et al. Scalable-effort classifiers for energy-efficient machine learning. In *DAC*, pages 1–6, 2015.

[6] Ermao Cai, Da-Cheng Juan, Dimitrios Stamoulis, et al. *NeuralPower*: Predict and deploy energy-efficient convolutional neural networks. In *ACML*, volume 77, pages 622–637, 2017.

[7] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In *NIPS*, volume 2, pages 598–605, 1989.

[8] Wei Wen, Chunpeng Wu, Yandan Wang, et al. Learning structured sparsity in deep neural networks. In *NIPS*, volume 29, pages 2082–2090, 2016.

[9] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, et al. Quantized neural networks: Training neural networks with low precision weights and activations. *JMLR*, 18(187):1–30, 2018.

[10] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *NIPS*, pages 3123–3131, 2015.

[11] Priyadarshini Panda, Abhronil Sengupta, and Kaushik Roy. Conditional deep learning for energy-efficient and enhanced pattern recognition. In *DATE*, pages 475–480, 2016.

[12] Dimitrios Stamoulis, Ting-Wu Rudy Chin, Anand Krishnan Prakash, et al. Designing adaptive neural networks for energy-constrained image classification. In *ICCAD*, pages 1–8, 2018.

[13] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, et al. Conditional computation in neural networks for faster models. In *Workshop track - ICLR*, 2016.

[14] Nafiul Rashid, Berken Utku Demirel, and Mohammad Abdullah Al Faruque. Ahar: Adaptive cnn for energy-efficient human activity recognition in low-power edge devices. *IEEE Internet of Things Journal*, 9(15):13041–13051, 2022.

[15] Xinyu Liu, Chenhong Cao, and Shengyu Duan. A low-power hardware architecture for real-time cnn computing. *Sensors*, 23(4):2045, 2023.