GREEN-AUTOML FOR PLASTIC LITTER DETECTION

Daphne Theodorakopoulos[1,2], **Christoph Manss**[1], **Frederic Stahl**[1] **& Marius Lindauer**[2]
[1]Marine Perception Research Department, German Research Center for Artificial Intelligence (DFKI)
[2]Institute of Artificial Intelligence, Leibniz University Hannover
daphne.theodorakopoulos@dfki.de

ABSTRACT

The world's oceans are polluted with plastic waste and the detection of it is an important step toward removing it. Wolf et al. (2020) created a plastic waste dataset to develop a plastic detection system. Our work aims to improve the machine learning model by using Green Automated Machine Learning (AutoML). One aspect of Green-AutoML is to search for a machine learning pipeline, while also minimizing the carbon footprint. In this work, we train five standard neural architectures for image classification on the aforementioned plastic waste dataset. Subsequently, their performance and carbon footprints are compared to an Efficient Neural Architecture Search as a well-known AutoML approach. We show the potential of Green-AutoML by outperforming the original plastic detection system by 1.1% in accuracy and using 33 times fewer floating point operations at inference, and only 29% of the carbon emissions of the best-known baseline. This shows the large potential of AutoML on climate-change relevant applications and at the same time contributes to more efficient modern Deep Learning systems, saving substantial resources and reducing the carbon footprint.

1 INTRODUCTION

Plastic pollution in oceans has become a threat to ocean health, the health of marine species, and food safety. Therefore, the UN set the conservation of the oceans as one of the Sustainable Development Goals (Desa et al., 2016). The detection and quantification of plastic waste is the first step to confronting this danger. Wolf et al. (2020) created a labeled dataset of polluted coastline images and a system to detect floating plastic litter, see Figure 1 for an example. Their system uses a two-step approach which first detects litter and then quantifies it. The work at hand aims to improve the first part, i.e. the plastic detector, in terms of accuracy and carbon emissions.[1]

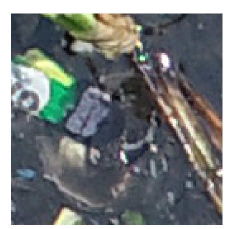

Figure 1: Example image of the data

As discussed by Tornede et al. (2021) and Tu et al. (2022), Green Automated Machine Learning (AutoML) has the potential to greatly contribute to efficiently finding better predictive models with smaller carbon footprints. We directly follow the call to action of Tu et al. (2022) by providing more evidence and insights. Concretely, we use AutoML to find an improved plastic detector. To this end, AutoML aims to find machine learning (ML) pipelines automatically, which minimizes human effort and the need for expert knowledge (Hutter et al., 2019; Bischl et al., 2023). Most AutoML methods, such as Neural Architecture Search (NAS) (Elsken et al., 2019; White et al., 2023), train many models which are later discarded or super-networks with parts being discarded, thus although the human effort is saved, the method in itself is fairly resource-intensive. The field of Green Artificial Intelligence reduces the energy consumption of machine learning models by taking the computational cost into account as an evaluation measure (Schwartz et al., 2020). Green-AutoML combines these two objectives and aims at the best of both worlds. Previous research was mainly conducted in resource-aware NAS, often motivated by hardware constraints, e.g., to deploy models on the edge (Zhao et al., 2021; Benmeziane et al., 2021). However, recent research switched to a viewpoint

[1]We acknowledge personal communication with Wolf et al. (2020) who provided the dataset, the original model, and insights into the problem domain as well as the development process of the system. This helped us to identify important aspects that Green-AutoML can contribute to in this important application.

of sustainability by aiming to reduce the carbon footprint of NAS, e.g., Bakhtiarifard et al. (2022) and Dou et al. (2023) created first energy-aware benchmarks for AutoML, unfortunately without any connection to climate-change relevant applications.

The aim of this work is to use Green-AutoML to create an improved plastic litter detection system in terms of performance and energy consumption. The chosen method is Efficient Neural Architecture Search (ENAS) (Pham et al., 2018). Although not considered state-of-the-art anymore, preliminary experiments with other NAS approaches showed that ENAS can still be considered one of the most efficient approaches and returns fairly small models. Moreover, the approach is compared to five well-established neural architectures a human ML expert might use to solve the problem without using AutoML. Finally, we show that while the chosen AutoML method is consuming more energy in the model construction phase, it is more sustainable at scale. Imagine the entire coastline of the world's oceans would be screened for plastic litter, the large number of images would require a model with low emissions at inference time.

2 Methodology

In this section, we provide a brief overview of the approach for determining an efficient architecture for predicting plastic litter on coastlines.

2.1 Data

The dataset contains 26 147 high geospatial resolution images taken in Cambodia by aerial surveys with a drone labeled to eight classes: Litter - high, Litter - low, Organic debris, Other, Sand, Stones, Vegetation, and Water. We refer to Wolf et al. (2020) for more details.

2.2 Carbon Footprint

To measure the CO_2 emissions, estimations are conducted using the `CodeCarbon` emissions tracker (Lacoste et al., 2019; Lottick et al., 2019). The emissions are estimated in grams of CO_2 equivalent (gCO_2eq). It is calculated as follows:

$$g\mathrm{CO}_2eq = 1000 \times \text{Carbon Intensity} \times \text{Power Usage}$$

The Carbon Intensity of electricity used for computation refers to the amount of CO_2 emitted per kilowatt-hour of electricity in kilograms dependent on the energy mix at the time and location. The power consumed by the computational infrastructure is referred to as Power Usage, which is measured in kilowatt-hours. For an Nvidia GPU, this is calculated using the Nvidia Management Library (NVIDIA Corporation, 2023).[2]

2.3 Standard Architectures

Since the baseline detector by Wolf et al. (2020) is only one neural architecture to solve the problem, we also evaluate five more well-known architectures from PyTorch (Paszke et al., 2019) for image classification: AlexNet (Krizhevsky et al., 2012), EfficientNetV2-S (Tan & Le, 2021), MobileNetV3small (Howard et al., 2019), ResNet18 (He et al., 2016) and VGG11 (Simonyan & Zisserman, 2014). For each model, the learning rate, the number of epochs, batch size, weight decay, and dropout are tuned by random search for 60 trials, each with the validation data. Table 1 presents the best-performing hyperparameters per model and the respective search space. Finally, the models are trained and pruned using PyTorch's global pruning with L1 unstructured pruning on all convolutional and linear layers by 90% (Paszke et al., 2019) to reduce the CO_2 emissions for classification.

2.4 Efficient Neural Architecture Search

We use ENAS (Pham et al., 2018), as an efficient AutoML method, to build an improved plastic detector. According to Elsken et al. (2019), applying NAS usually consists of a search space, a strategy to optimize it, and a strategy estimating the performance of the sampled architectures. ENAS is

[2]For more details, see `https://mlco2.github.io/codecarbon/methodology.html`

Table 1: Hyperparameter search space and the tuned settings for each architecture

Hyperparameter	**Range**	**Scaling**	**AlexNet**	**EfficientNet**	**MobileNet**	**ResNet**	**VGG**
Learning Rate	0.0001 - 0.1	log	0.001	0.001	0.001	0.0001	0.001
Weight Decay	0.0001 - 0.1	log	0.0001	0.0001	0.0001	0.001	0.0001
Dropout	0.1 - 0.9	linear	0.5	-	0.5	-	0.9
Number of Epochs	10 - 100	linear	80	30	80	50	80
Batch size	8 - 64	log	8	32	8	8	64

a one-shot method; it trains a large super-network and extracts sub-networks as neural architectures from it. ENAS uses reinforcement learning as a strategy, i.e., a controller searches for a sub-network that maximizes the reward on the validation data, and the selected sub-network is trained to minimize the loss on the training data. Because of shared weights, ENAS is fairly memory efficient.

The Python package Neural Network Intelligence (NNI) (Microsoft, 2021) provides an implementation of the ENAS search space and strategy. We set the search space as follows: width is set to 32, the number of stacked cells to (3,3), and the 'dataset' to 'imagenet'.[3] All other hyperparameters are left at their default values. The reward metric is set to the training loss. For the evaluation strategy for each architecture and also the training of the final architecture, we use the classification defaults with weight decay of 0.0001, 100 epochs, and a batch size of 32 for training. Additional hyperparameter optimization and pruning might further improve results (Bansal et al., 2022), but we have not done so here.

3 EXPERIMENTS

The CO_2 emissions were calculated as described in section 2.2. Each model was trained on an RTXA6000 GPU. The location of the server is Kaiserslautern, Germany, and the experiments were run in January and February 2023. The average CO_2 emissions in Kaiserslautern at the time were 536 g/kWh [4]. After training, each model was evaluated on the test set with the following metrics: test accuracy, CO_2 emissions consumed to classify the test set, and the number of floating point operations (flops) consumed to classify one image. The implementation is available on GitHub [5].

3.1 COMPARING STANDARD ARCHITECTURES VS AUTOML FOR INFERENCE

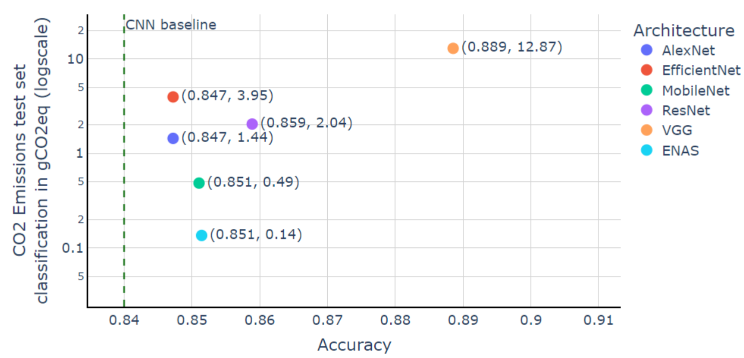

Figure 2: Test accuracy vs. CO_2 emissions on the test set

Figure 2 shows the accuracy and emissions on the test set for each model to provide a qualitative impression and Table 2 provides the quantitative results. For the CNN by Wolf et al. (2020), the

[3] 'Dataset' can be set to 'cifar' or 'imagenet'. The documentation of NNI states "The essential differences are in "stem" cells[. . .]. Choosing "imagenet" means more downsampling at the beginning of the network."

[4] According to the "SWK Stadtwerke Kaiserslautern" https://www.swk-kl.de/produkte-services/service/energietraegermix

[5] The code is publicly available on GitHub: https://github.com/DFKI-NI/green_automl_for_plastic_litter_detection

Table 2: Showing different metrics comparing predictive performance, flops and emissions.

Architecture	**Test Accuracy**	**Flops**	**Emissions on Test set** (in gCO_2eq)
CNN of Wolf et al. (2020)	0.840	803 371 952	N/A
AlexNet	0.847	250 774 464	1.44
EfficientNet	0.847	941 679 616	3.95
MobileNet	0.851	**19 857 408**	0.49
ResNet	0.859	593 817 600	2.04
VGG	**0.889**	2 563 809 280	12.87
ENAS	0.851	24 103 936	**0.14**

emission data is not available. Interestingly, the accuracy varies only a little, whereas the flops and CO_2 emission cover a large range. All models outperform the human baseline in accuracy. VGG is the best in accuracy but also by far the largest considering flops and CO_2 emissions. AlexNet and EfficientNet are slightly worse in accuracy than ENAS but are a lot larger and thus less efficient. While MobileNet and ENAS have the same accuracy, MobileNet is slightly lower in flops than ENAS, but it produces considerably more CO_2 emissions. Although not being optimized for it, ENAS is lowest in CO_2 emissions, while achieving considerable accuracy. Compared to the CNN baseline, the flops used by ENAS are reduced by a factor of around 33. To this end, when considering accuracy, flops and CO_2 emission combined, ENAS built one of the most promising models.

3.2 Discussing the Carbon Footprint of Training

The disadvantage of AutoML is its large resource consumption in the model creation phase. However, when regarding the model's CO_2 emissions at scale, the lower emissions in the inference phase compensate for the high resource usage during training. The length of the world's coastline is approximately 1.6 million km (World Resources Institute, 2004-2023). Based on a single image size of 20 cm^2 and a water coverage ratio of 10 meters per meter of coastline (Wolf et al., 2020), an estimated 408.5 billion images require classification.

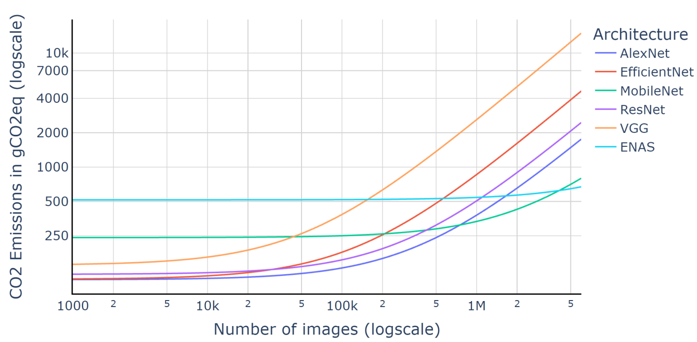

Figure 3: CO_2 emissions on the test set per number of images including training.

Figure 3 shows the total CO_2 emissions of the models as a function of the number of images that are classified by the model. The function can be described as follows:

$$\text{total } CO_2 \text{ emissions(\#images)} = CO_2 \text{ emissions training} + \text{\#images} \times \frac{CO_2 \text{ emissions test set}}{\text{size test set}}$$

As expected, ENAS starts as the highest since the NAS process takes up more energy than training a single, known architecture. However, ENAS ends as the lowest in CO_2. It passes VGG at around 156 000 images. EfficientNet, ResNet, and AlexNet follow at 556 000, 1.14 million, and 1.66 million images, respectively. MobileNet is also quite efficient in classifying and is only surpassed by ENAS at around 4.12 million images. When considering the estimate of 408.5 billion images, the difference between ENAS and MobileNet lies in around 27.3 million gCO_2eq emissions, i.e., the ENAS model would consume only around 29% of the emissions that MobileNet would use to classify the world's coastline. This shows that the AutoML approach is more sustainable at scale and pays off even if it were only to be used on 0.001% of the coastlines compared to hand-designed solutions.

3.3 Limitations and Future Work

Our study focused on the CO_2 emissions of the model alone, but we recognize that the process of data collection, preprocessing, and algorithm implementation are also significant sources of emissions. According to personal communication with Wolf et al. (2020), data collection was a crucial and significant part of the project. The collection process took place in various locations and involved transportation emissions, as well as the emissions of the usage and manufacturing of the hardware used to capture the images, such as drones. Attention should be paid to the implementation of the algorithm. Before the data can be used, labeling and preprocessing need to be performed. In the development phase of the original model, many different settings were experimented with, which again used up energy. Future research should consider estimating the emissions associated with the entire research process.

Moreover, the CO_2 emissions depend on the energy mix and the hardware used for model training and execution. To reduce emissions, periods of low energy demand or high renewable energy availability can be identified for model training. Additionally, the choice of processor, programming language, and libraries can impact CO_2 emissions. Comparative studies of different processors and libraries could be conducted in future research.

In addition, the method used so far, i.e. ENAS, is no longer state-of-the-art in AutoML. ENAS has not explicitly searched for an energy-efficient model, but the ENAS search space appears to be well suited for that. Multi-objective optimization on CO_2 emissions (Bakhtiarifard et al., 2022) or other specific methods (Wang et al., 2019) are also promising directions for future work.

Another interesting aspect is the utility of a developed model in relation to its CO_2 emissions. How much CO_2 is the application worth? In this work, we assumed that one would want to classify the shorelines of the world's oceans. In most applications, however, such an upper bound cannot be determined easily. Estimating the net worth of an application could be done through a CO_2 budget calculator that takes hardware components into account as well as environmental harm, prospected usage, and other factors. Possibly, the CO_2 compensated by the application can also be considered.

Lastly, there are other aspects besides CO_2 emissions to consider, such as other greenhouse gases, nuclear waste, and e-waste produced by technology. Additionally, there might be better ways to calculate CO_2 emissions. We used CodeCarbon but that library has some limitations in the way that it calculates the CO_2 equivalent. That is, the life-cycle emissions of the computing infrastructure are not taken into account, and the energy mix is estimated based on the country or city, rather than being directly measured.

4 Conclusion

In this work, we demonstrated the potential of Green-AutoML as a key technology for the efficient development of efficient models for climate-change-related applications. This opens up further paths towards real Green-AutoML systems that so far are only independently studied in terms of efficient development, efficient models, or important applications, but not jointly. In particular, we studied ways to obtain an improved plastic detector compared with Wolf et al. (2020). To the best of our knowledge, we are the first to show that additional CO_2 emissions spent by AutoML methods are compensated by the lower emissions consumed at inference time if the model is applied frequently. Future work includes the development of an AutoML algorithm optimizing for energy efficiency and considering the overall resource consumption of the entire process.

Acknowledgements

We thank Wolf et al. (2020) for the insights, and access to the data and the system. Marius Lindauer was supported by the German Federal Ministry of the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (GreenAutoML4FAS project no. 67KI32007A). The DFKI Niedersachsen (DFKI NI) is funded in the ”zukunft.niedersachsen” by the Lower Saxony Ministry of Science and Culture and the Volkswagen Foundation (funding no. ZM3480).

REFERENCES

P. Bakhtiarifard, C. Igel, and R. Selvan. Energy consumption-aware tabular benchmarks for neural architecture search. (arXiv:2210.06015), Oct 2022. URL `http://arxiv.org/abs/2210.06015`. arXiv:2210.06015 [cs, stat].

A. Bansal, D. Stoll, M. Janowski, A. Zela, and F. Hutter. Jahs-bench-201: A foundation for research on joint architecture and hyperparameter search. In *Proceedings of the Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

H. Benmeziane, K. El Maghraoui, H. Ouarnoughi, S. Niar, M. Wistuba, and N. Wang. A comprehensive survey on hardware-aware neural architecture search. (arXiv:2101.09336), Jan 2021. URL `http://arxiv.org/abs/2101.09336`. arXiv:2101.09336 [cs].

B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A. Boulesteix, D. Deng, and M. Lindauer. Hyperparameter optimization: Foundations, algorithms, best practices and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2023.

UN Desa et al. Transforming our world: The 2030 agenda for sustainable development. 2016.

S. Dou, X. Jiang, C. Zhao, and D. Li. Ea-has-bench: Energy-aware hyperparameter and architecture search benchmark. In *Proceedings of the international conference on representation learning (ICLR)*, 2023.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *J. Mach. Learn. Res.*, 20:55:1–55:21, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le. Searching for MobileNetV3. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1314–1324. IEEE Computer Society, 2019.

F. Hutter, L. Kotthoff, and J. Vanschoren (eds.). *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019. Available for free at http://automl.org/book.

A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres. Quantifying the carbon emissions of machine learning. *Workshop on Tackling Climate Change with Machine Learning at NeurIPS 2019*, 2019.

K. Lottick, S. Susai, S. A. Friedler, and J. P. Wilson. Energy usage reports: Environmental awareness as part of algorithmic accountability. *Workshop on Tackling Climate Change with Machine Learning at NeurIPS 2019*, 2019.

Microsoft. Neural Network Intelligence, 1 2021. URL `https://github.com/microsoft/nni`.

NVIDIA Corporation. Nvidia management library (nvml), 2023. URL `https://developer.nvidia.com/nvidia-management-library-nvml`.

A. Paszke, S. Gross, F. Massa, A. Lerer, et al. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Proceedings of the 32nd International Conference on Advances in Neural Information Processing Systems (NeurIPS'19)*, pp. 8024–8035, 2019.

H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean. Efficient Neural Architecture Search via parameter sharing. In J. Dy and A. Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, volume 80. Proceedings of Machine Learning Research, 2018.

R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni. Green AI. *Commun. ACM*, 63(12):54–63, 2020.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

M. Tan and Q. Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pp. 10096–10106. PMLR, 2021.

T. Tornede, A. Tornede, J. Hanselle, M. Wever, F. Mohr, and E. Hüllermeier. Towards green automated machine learning: Status quo and future directions. *arXiv:2111.05850 [cs.LG]*, 2021.

R. Tu, N. Roberts, V. Prasad, S. Nayak, P. Jain, F. Sala, G. Ramakrishnan, A. Talwalkar, W. Neiswanger, and C. White. Automl for climate change: A call to action. In *Workshop Proceedings of Tackling Climate Change with Machine Learning*, 2022. URL `https://doi.org/10.48550/arXiv.2210.03324`.

D. Wang, M. Li, L. Wu, V. Chandra, and Q. Liu. Energy-aware neural architecture optimization with fast splitting steepest descent. *CoRR*, abs/1910.03103, 2019. URL `http://arxiv.org/abs/1910.03103`.

C. White, M. Safari, R. Sukthanker, B. Ru, T. Elsken, A. Zela, D. Dey, and F. Hutter. Neural architecture search: Insights from 1000 papers. *CoRR*, abs/2301.08727, 2023.

M. Wolf, K. van den Berg, S. P. Garaba, N. Gnann, K. Sattler, F. Stahl, and O. Zielinski. Machine learning for aquatic plastic litter detection, classification and quantification (aplastic-q). *Environmental Research Letters*, 15(11):114042, 2020.

World Resources Institute. Coastal and marine ecosystems — marine jurisdictions: Coastline length, 2004-2023. URL `https://web.archive.org/web/20120419075053/http://earthtrends.wri.org/text/coastal-marine/variable-61.html`.

Z. Zhao, K. Wang, N. Ling, and G. Xing. Edgeml: An automl framework for real-time deep learning on the edge. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, IoTDI '21, pp. 133–144. Association for Computing Machinery, 2021.