# TOWARDS LEVERAGING AUTOML FOR SUSTAINABLE DEEP LEARNING: A MULTI-OBJECTIVE HPO APPROACH ON DEEP SHIFT NEURAL NETWORKS

**Leona Hennig, Tanja Tornede, Marius Lindauer**
Institute of Artificial Intelligence, Leibniz University Hannover, Germany
{l.hennig, t.tornede, m.lindauer}@ai.uni-hannover.de

## ABSTRACT

Deep Learning (DL) has advanced various fields by extracting complex patterns from large datasets. However, the computational demands of DL models pose environmental and resource challenges. Deep shift neural networks (DSNNs) offer a solution by leveraging shift operations to reduce computational complexity at inference. Following the insights from standard DNNs, we are interested in leveraging the full potential of DSNNs by means of AutoML techniques. We study the impact of hyperparameter optimization (HPO) to maximize DSNN performance while minimizing resource consumption. Since this combines multi-objective (MO) optimization with accuracy and energy consumption as potentially complementary objectives, we propose to combine state-of-the-art multi-fidelity (MF) HPO with multi-objective optimization. Experimental results demonstrate the effectiveness of our approach, resulting in models with over 80% in accuracy and low computational cost. Overall, our method accelerates efficient model development while enabling sustainable AI applications.

## 1 INTRODUCTION

Deep Learning (DL) is considered one of the most promising approaches to extracting information from large data collections with complex structures. This includes performing computations in IoT environments and on edge devices (Li et al., 2018; Zhou et al., 2019). With the ever-increasing size and performance of such models due to the progress in science and industry, a computational cost is associated with them (Sze et al., 2017). Minimizing this cost directly affects a model's environmental impact (Schwartz et al., 2020). Thus, resources are freed up and can be used for other tasks, e.g., those with resource limitations (Howard et al., 2017). With our approach, we contribute to DL in such low-resource settings.

Deep shift neural networks (DSNNs) offer great potential in reducing power consumption compared to traditional deep learning models (Elhoushi et al., 2021). Instead of floating point arithmetic, they leverage shift operations — specifically, bit shifting — as the computational unit, which boosts efficiency by replacing costly multiplication operations in convolutional networks. We suspect that the configuration of DSNNs has a huge impact on both performance and computational efficiency.

One of the key challenges with DSNNs is determining the appropriate level of precision for the shift operations to minimize quantization errors without excessively increasing the computational load. In this study, we leverage DSNNs in combination with automated machine learning (AutoML) to find the optimal configuration of DSNNs in the spirit of Green AutoML (Tornede et al., 2023). This is achieved by hyperparameter optimization (HPO) using the framework SMAC3 proposed by Lindauer et al. (2022), which automates the search for optimal model configurations. Integrating multi-fidelity (MF) and multi-objective optimization (MO) techniques (Belakaria et al., 2020) facilitates an optimal exploration of the hyperparameter space that jointly prioritizes performance and energy consumption (Deb, 2014). SMAC's MO implementation effectively balances the trade-off between achieving high predictive accuracy and minimizing energy consumption. The MF aspect allows for the efficient use of computational resources by evaluating configurations at varying levels of detail. Employing tools like CodeCarbon (Lacoste et al., 2019; Lottick et al., 2019) during the training and evaluation phases

provides real-time insights into the energy consumption and carbon emissions associated with each model configuration.

Overall, we make the following contributions:

1. A configuration space specific for DSNNs,
2. a Green AutoML approach for efficiency-driven model building,
3. insights into the design decisions for obtaining optimal tradeoffs between accuracy and energy efficiency, and
4. combining MO and MF approaches in SMAC to improve optimization performance and computational resource utilization.

## 2 BACKGROUND

The following chapter introduces foundational concepts for our approach.

### 2.1 DEEP SHIFT NEURAL NETWORKS

Deep shift neural networks (DSNNs) are a novel approach to reducing the computational and energy demands of deep learning (Elhoushi et al., 2021). They achieve a considerable reduction in latency time by simplifying the network architecture such that they replace the traditional multiplication operations in neural networks by bit-wise shift operations and sign flipping, making DSNNs suitable for computing devices with limited resources. There are two methods for training DSNNs (Elhoushi et al., 2021): DeepShift-Q (Quantization) and DeepShift-PS (Powers of two and Sign). DeepShift-Q involves training regular weights constrained to powers of 2 by quantizing weights to their nearest power of two during both forward and backward passes. DeepShift-PS directly includes the values of the shifts and sign flips as trainable parameters.

The DeepShift-Q approach obtains the sign matrix $S$ from the trained weight matrix $W$ as $S = sign(W)$. The power matrix $P$ is the base-2 logarithm of $W$'s absolute values, i.e., $P = \log_2(|W|)$. After rounding $P$ to the nearest power of two, $P_r = round(P)$, the quantized weights $\tilde{W}_q$ are calculated by applying the sign from $S$, shown as $\tilde{W}_q = \mathit{flip}(2^{P_r}, S)$. The DeepShift-PS approach optimizes neural network weights by directly adapting the shift ($\tilde{P}$) and sign ($\tilde{S}$) values. The shift matrix $\tilde{P}$ is obtained by rounding the base-2 logarithm of the weight values, $\tilde{P} = round(P)$, and the sign flip $\tilde{S}$ is computed as $\tilde{S} = sign(round(S))$. Weights are calculated as $\tilde{W}_{ps} = \mathit{flip}(2^{\tilde{P}}, \tilde{S})$, where the sign flip operation $\tilde{S}$ assigns values of $-1$, $0$, or $+1$ based on $s$.

### 2.2 HYPERPARAMETER OPTIMIZATION

The increasing complexity of deep learning algorithms enhances the need for automated hyperparameter optimization (HPO) to increase model performance (Bischl et al., 2023). Consider a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N} \in \mathbb{D} \subset \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the instance space and $\mathcal{Y}$ is the target space, and a hyperparameter configuration space $\Lambda = \{\lambda_1, \ldots, \lambda_L\}$, $L \in \mathbb{N}$. In our study, $\mathcal{M}$ denotes the space of possible DSNN models. An algorithm $\mathcal{A} : \mathbb{D} \times \Lambda \to \mathcal{M}$ trains a model $M \in \mathcal{M}$, instantiated with a configuration of $L$ hyperparameters sampled from $\Lambda$, on a training subset of $\mathcal{D}$. The dataset $\mathcal{D}$ is split into training, validation, and test sets: $\mathcal{D}_{train}$, $\mathcal{D}_{val}$, and $\mathcal{D}_{test}$ respectively. The performance of the algorithm is assessed via an expensive-to-evaluate loss function $\mathcal{L} : \mathcal{M} \times \mathbb{D} \to \mathbb{R}$. The optimization objective of HPO is to find the configuration $\lambda^* \in \Lambda$ with minimal validation loss $\mathcal{L}$, such that:

$$\lambda^* \in \arg\min_{\lambda \in \Lambda} \mathcal{L}\big(\mathcal{A}(\mathcal{D}_{train}, \lambda), \mathcal{D}_{val}\big). \tag{2.1}$$

This process tunes hyperparameters based on validation set performance. Models are trained on $\mathcal{D}_{train}$ and optimized using $\mathcal{D}_{val}$. Finally, the model's performance is assessed on $\mathcal{D}_{test}$.

### 2.3 BAYESIAN OPTIMIZATION

Bayesian Optimization (BO) is a strategy for global optimization of black-box loss functions $\mathcal{L} : \mathcal{M} \times \mathbb{D} \to \mathbb{R}$ that are expensive to evaluate (Jones et al., 1998). BO uses a surrogate model $\mathcal{S}$,

a probabilistic model to approximate the loss function, commonly given by a Gaussian Process or a Random Forest (Rasmussen & Williams, 2006; Hutter et al., 2011). An acquisition function $\mathcal{C} : \Lambda \rightarrow \mathbb{R}$ guides the search for the next optimal evaluation points by balancing the exploration-exploitation trade-off, based on the set of previously queried points $\{(\lambda_1, \mathcal{L}_1), ..., (\lambda_{m-1}, \mathcal{L}_{m-1})\}$ at time $m$. $\mathcal{L}$ is only evaluated at certain points, with which the surrogate model is updated.

### 2.4 Multi-Fidelity Optimization

Since it is not feasible to fully train multiple configurations of DSNNs for comparison due to computational efficiency, we employ a multi-fidelity (MF) approach (Li et al., 2017), which is a common strategy in AutoML to navigate the trade-off between performance and approximation error (Hutter et al., 2019). MF approaches train cheap-to-evaluate proxies of black-box functions following different heuristics, e.g., allocating a small number of epochs to many configurations in the beginning and training the best-performing ones on an increasing number of epochs. Formally, we define a space of fidelities $\mathcal{F}$ and aim to minimize a high-fidelity function $F \in \mathcal{F}$ (Kandasamy et al., 2019):

$$\min_{\lambda \in \Lambda} F(\lambda) \,. \tag{2.2}$$

We aim to approximate $F \in \mathcal{F}$, using a series of lower-fidelity, less expensive approximations $\{f(\lambda)_1, \ldots, f(\lambda)_j\}$, where $j$ denotes the total number of fidelity levels. The allocated resources for evaluating a model's performance at various fidelities are referred to as a budget. MF assumes that the highest fidelities approximate the black-box function best.

### 2.5 Multi-Objective Optimization

Multi-Objective Optimization (MO) addresses problems involving multiple, often competing objectives. This approach is used in scenarios where trade-offs between two or more conflicting objectives must be navigated, such as, in the context of DSNNs, enhancing accuracy alongside reducing energy consumption. MO aims to identify Pareto-optimal solutions (Deb, 2014). To ensure effective approximation of the Pareto front, new points are added based on the current observation dataset $\mathcal{D}_{obs} = \{(\lambda_1, \mathcal{L}(\lambda_1)), \ldots, (\lambda_n, \mathcal{L}(\lambda_n))\}$ at time $n$. These points augment the surface formed by the non-dominated solution set $D_n^\star$, which satisfies the condition for $d$ objective variables and a loss function $\mathcal{L} = (\mathcal{L}_1, \ldots, \mathcal{L}_d)$ (Wada & Hino, 2019): For all $\lambda, (\lambda, \mathcal{L}(\lambda)) \in \mathcal{D}_n^\star \subset \mathcal{D}_n$ and $\lambda', (\lambda', \mathcal{L}(\lambda')) \in \mathcal{D}_n$ exists $k \in \{1, \ldots, d\}$, such that $\mathcal{L}_k(\lambda) \leq \mathcal{L}_k(\lambda')$.

## 3 Approach

To computationally enhance Deep shift neural networks (DSNN) via AutoML, we employ multi-fidelity optimization (MF). We introduce different fidelities to the training process by increasing the number of shift layers in the model. The initially trained models will include fewer shift layers, which will increase during the optimization process. We assume this will guide the search toward the highest-performing shift networks since a low number of shift layers introduces the least numerical inaccuracy and mathematical uncertainty to the model. We aim to investigate whether picking the best-performing models under low model-intrinsic uncertainty continues to show improved performance when combined with more shift layers.

In Section 2, we explained the MF approach. One algorithm that formalizes its heuristics is successive halving (Jamieson & Talwalkar, 2016), where $n_c$ configurations are trained on an initial small budget $b_I$. After that, the $\nu/(\nu+1)$ best-performing configurations are trained on a budget of $\nu b_I/(\nu+1)$ until the best configuration is determined. We address the trade-off between $b_I$ and $n_c$, or between approximation error and exploration inherent in successive halving, using the HyperBand algorithm for MF. HyperBand (Li et al., 2017) runs successive halving in multiple brackets, where each bracket provides a combination of $n_c$ and a fraction of the total budget per configuration so that they sum up to the total budget.

We extend this to multi-fidelity multi-objective optimization (MFMO). We simultaneously address the accuracy of the model as well as its energy consumption. The goal is to optimize the performance of the DSNNs, ensuring they achieve high accuracy and robustness in their predictive capabilities; secondly, to minimize energy consumption, a critical factor given the environmental implications of

computational efficiency. To this end, we implement a two-dimensional objective function $f_{MO} : \Lambda \to \mathbb{R}^2, \quad f_{M0}(\lambda) = \big(f_{loss}(\lambda), f_{emission}(\lambda)\big)$, where, given a configuration $\lambda \in \Lambda$, $f_{loss}(\lambda)$ aims to minimize the loss, enhancing the model's accuracy, and $f_{emission}(\lambda)$ seeks to minimize the energy consumption during training and inference, promoting environmental sustainability. We aim to solve the following optimization problem:

$$\arg\min_{\lambda \in \Lambda} f_{M0}(\lambda)\,. \tag{3.1}$$

SMAC3 uses a mean aggregation strategy by calculating the objective's arithmetic mean to aggregate multiple objectives into a single scalar value for MF optimization. We use the ParEGO algorithm (Knowles, 2006) to compute Pareto optimal objectives. It transforms the multi-objective problem into a series of single-objective problems by introducing varying weight parameters for the objectives in each iteration of HyperBand. Thus optimizing a different scalarization per evaluation to approximate the Pareto front. The resulting single-objective optimization function can now be evaluated in an MF setting.

## 4 Experiments

In the following section, we detail the setup and methodology used to evaluate our approach discussed in Section 3, focusing on optimizing deep shift neural networks (DSNNs) through multi-fidelity, multi-objective optimization (MFMO) as well as multi-fidelity optimization (MF). We discuss how our approach successfully navigates the model performance and environmental impact trade-offs.

### 4.1 Evaluation Setup

We train and evaluate our models on the Cifar10 dataset (Krizhevsky et al., 2009), using NVIDIA A100 GPUs. For hyperparameter optimization (HPO), we rely on using SMAC3 (Lindauer et al., 2022). To incorporate the environmental impact into our HPO workflow, we use the CodeCarbon emissions tracker (Lacoste et al. (2019), Lottick et al. (2019)) to track carbon emissions from computational processes by monitoring energy use and regional energy mix in $gCO_2eq$, grams of $CO_2$ equivalent (Tornede et al., 2023). We chose a pre-trained ResNet20 (He et al., 2016).

### 4.2 Results

Figure 1 displays the Top-1 test accuracy of a DSNN trained with the default configuration from Table 1 as defined by Elhoushi et al. (2021), as well as the test accuracy from the configured DSNN according to SMAC3 with MF using HyperBand, with the number of shift layers inducing the fidelities. The Pareto-optimal Solution No. 1 configuration from Table 1 is used. The MF algorithm, trained on the same seed as MFMO, identifies it as the optimal model configuration out of fifty evaluated configurations. While volatile to a certain degree, the plot clearly shows that the model's performance with the optimal configuration surpasses that of the model instantiated with the default configuration by at least three percent. This verifies our approach that MF can successfully be executed with DSNN-specific budgets such as the shift depth.

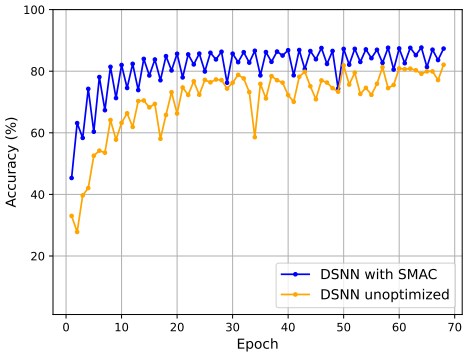


Figure 1: Test accuracy of the default DSNN and a configured DSNN

Furthermore, it confirms our assumption from Section 3 that the models can be evaluated under low model-intrinsic uncertainty and continue to perform well when instantiated with more shift layers.

| Parameter | Search Space | Default | POS 1 | POS 2 |
|---|---|---|---|---|
| Batch Size | [32, 128] | 128 | 97 | 102 |
| Optimizer | SGD, Adam, Adagrad, Adadelta, RMSProp, RAdam, Ranger | SGD | Ranger | SGD |
| Learning Rate | [0.001, 0.1] | 0.1 | 0.0847 | 0.0798 |
| Momentum | [0.0, 0.9] | 0.9 | 0.5016 | 0.6738 |
| Epochs | [5, 100] | 200 | 67 | 91 |
| Weight Bits | [2, 8] | 5 | **4** | **7** |
| Activation Integer Bits | [2, 32] | 16 | **4** | **29** |
| Activation Fraction Bits | [2, 32] | 16 | **4** | **10** |
| Shift Depth | [0, 20] | 20 | **5** | **16** |
| Shift Type | Q, PS | PS | Q | Q |
| Rounding | deterministic, stochastic | deterministic | stochastic | deterministic |
| Weight Decay | [1e-6, 1e-2] | 0.0001 | 0.0026 | 0.00012 |
| Accuracy (in %) | | | 83.50 | 84.67 |
| Emissions (in $gCO_2eq$) | | | 0.1661 | 0.1673 |

Table 1: MFMO findings, incl. two Pareto optimal solutions (POS 1 & 2)

Table 1 shows two Pareto optimal solutions out of 33 evaluated configurations with our MFMO approach. Evaluating solution No. 1 results in a Top-1 accuracy of 83.50% and 0.1661 $gCO_2eq$. Solution No. 2 yields a Top-1 accuracy of 84,67%, producing 0.1673 $gCO_2eq$. Note that they were trained for fewer epochs due to computational limitations. Both configurations yield good performance results while keeping emissions low.

The DSNN-specific model hyperparameters include activation integer bits, activation fraction bits, weight bits, and shift depth. Solution No.1 includes fewer shift layers with fewer bits representing the weights activation function values. Activation bits and weight bits seem to affect emissions and model performance similarly. Fewer shift layers increase the computational demand of a model. However, the reduced number of bits used to replace floating-point operations (FLOPs) seems to compensate for this since solution No. 1 produces approximately the same amount of emissions as solution No. 2. This one contains more than three times the number of shift layers, using only four convolutional layers with FLOPs, but uses substantially more representation bits. There seems to be a trade-off between the number of shift layers and the number of representation bits that, when navigated efficiently, yields model configurations with satisfying performance and energy consumption. This supports our claim that our MFMO approach is promising for building high-performance, energy-efficient DSNNs.

## 5 Conclusion

In this work, we present our Green AutoML approach towards the sustainable optimization of DSNNs through a multi-fidelity, multi-objective (MFMO) HPO framework. We address the critical intersection between advancing the capabilities of Deep Learning and environmental sustainability. By leveraging AutoML tools and integrating the environmental impact as an objective, we navigate the trade-off between model performance and efficient resource utilization.

Our experimental results highlight the potential of our approach. We successfully optimized a DSNN to achieve high accuracy while minimizing energy consumption. We have delivered a thorough configuration space for DSNNs, introduced a Green AutoML approach for efficiency-driven model development, and provided valuable insights into the design decisions. Future work includes extending our approach to multiple benchmarks and neural network architectures to sufficiently verify our approach on a broad spectrum of model designs and applications. This way, we hope to gain more insight into critical design decisions for DSNNs and other models built for energy-efficient computations. Revisiting our MFMO implementation to find a more efficient way for ParEGO and Hyperband to intertwine, e.g. by finding a more effective way to assign budgets and weights to configurations, is subject to further work to lighten the computational load when computing the MFMO Pareto fronts. Furthermore, we will investigate more DSNN-specific fidelity types and MO algorithms to achieve further reductions in model emissions.

ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry of the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (GreenAutoML4FAS project no. 67KI32007A).

## REFERENCES

S. Belakaria, A. Deshwal, and J. Doppa. Multi-Fidelity Multi-Objective bayesian optimization: An output space entropy search approach. *AAAI*, 34(06):10035–10043, 2020.

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. *WIREs Data. Mining. Knowl. Discov.*, 13(2), 2023.

K. Deb. *Multi-objective Optimization*, pp. 403–449. Springer US, 2014.

M. Elhoushi, Z. Chen, F. Shafiq, Y. Henry Tian, and J. Yiwei Li. DeepShift: Towards multiplication-less neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2021, virtual, June 19-25, 2021*, pp. 2359–2368, 2021.

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 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778, 2016.

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

F. Hutter, H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In C. Coello Coello (ed.), *Learning and Intelligent Optimization - 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers*, volume 6683, pp. 507–523, 2011.

F. Hutter, L. Kotthoff, and J. Vanschoren (eds.). *Automated Machine Learning - Methods, Systems, Challenges*. Springer Publishing Company, Incorporated, 2019.

K. Jamieson and A. Talwalkar. Non-stochastic best arm identification and Hyperparameter Optimization. In A. Gretton and C. Robert (eds.), *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS'16)*, volume 51. Proceedings of Machine Learning Research, 2016.

D. Jones, M. Schonlau, and W. Welch. Efficient global optimization of expensive black-box functions. *J. Glob. Optim.*, 13(4):455–492, 1998.

K. Kandasamy, G. Dasarathy, J. Oliva, J. Schneider, and B. Póczos. Multi-fidelity gaussian process bandit optimisation. *J. Artif. Intell. Res.*, 66:151–196, 2019.

J. Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Trans. Evol. Comput.*, 10(1):50–66, 2006.

A. Krizhevsky, G. Hinton, and o. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

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

H. Li, K. Ota, and M. Dong. Learning IoT in edge: Deep learning for the internet of things with edge computing. *IEEE Netw.*, 32(1):96–101, 2018.

L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *J. Mach. Learn. Res.*, 18:185:1–185:52, 2017.

M. Lindauer, K. Eggensperger, M. Feurer, A. Biedenkapp, D. Deng, C. Benjamins, T. Ruhkopf, R. Sass, and F. Hutter. SMAC3: A versatile bayesian optimization package for hyperparameter optimization. *J. Mach. Learn. Res.*, 23:54:1–54:9, 2022.

K. Lottick, S. Susai, S. Friedler, and J. Wilson. Energy usage reports: Environmental awareness as part of algorithmic accountability. *arXiv:1911.08354*, 2019.

C. Edward Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

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

V. Sze, Y. Chen, T. Yang, and J. Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proc. IEEE*, 105(12):2295–2329, 2017.

T. Tornede, A. Tornede, J. Hanselle, F. Mohr, M. Wever, and E. Hüllermeier. Towards green automated machine learning: Status quo and future directions. *J. Artif. Intell. Res.*, 77:427–457, 2023.

T. Wada and H. Hino. Bayesian optimization for multi-objective optimization and multi-point search. *arXiv:1905.02370*, 2019.

Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proc. IEEE*, 107(8):1738–1762, 2019.