Hardware-aware NAS by Genetic Optimisation with a Design Space Exploration Simulator

Lotte Hendrickx[1], Arne Symons[2], Wiebe Van Ranst[1], Marian Verhelst[2], Toon Goedemé[1]
[1]EAVISE-PSI-ESAT, KU Leuven
Jan Pieter De Nayerlaan 5, 2860 Sint-Katelijne-Waver, Belgium
[2]MICAS, Department of Electrical Engineering, KU Leuven
Kasteelpark Arenberg 10, 3310 Leuven, Belgium
lotte.hendrickx@kuleuven.be

Abstract

Neural Architecture Search (NAS) has shown its potential in aiding in the development of more efficient neural networks. In regard to hardware, efficiency often equates to power usage or latency. Over the years many researchers have incorporated hardware performance into their NAS experiments. However, accurately modelling hardware performance is a challenge in itself. We look to the field of design space exploration (DSE) for more precise performance metrics on neural network accelerators and incorporate the results into our NAS search. Our experiments show that doing so achieves a significant reduction in latency and energy consumption. The approach we propose also enables detailed insight in the breakdown of the energy consumption and latency of the optimised model.

1. Introduction

The introduction of Neural Architecture Search (NAS) has helped accelerate developments in deep learning in the past few years. NAS aids in finding the most suitable neural networks for a specific goal instead of being handcrafted by experts to suit one specific purpose. In many real-life applications of artificial intelligence compute and memory resources are limited. These limitations make it difficult to use off-the-shelf deep learning models that are not optimal for the target hardware. In these cases NAS' capabilities in discovering new neural networks can be leveraged to overcome these challenges. For example, NAS can take the specifics of certain types of hardware into account when searching for optimal solutions to the problem at hand by looking for smaller, faster and more energy efficient neural networks. A plethora of NAS strategies have been proposed in the past few years. Many of these strategies are multi-objective and therefore offer the possibility to find an optimal neural network that satisfies multiple requirements. Depending on the implementation, some of these existing NAS algorithms already take into account some form of hardware performance as well as the performance of the neural network. For example, hardware performance can be measured through latency or by calculating the size of the neural network. More in-depth metrics can provide better insight but are more complicated and require expert knowledge on different hardware architectures.

Alongside the algorithmic deep learning improvements, specialized application-specific integrated circuit (ASIC) hardware architectures to accelerate the inference of deep learning workloads have been proposed [3,4,13,31,40]. Acceleration performance has drastically improved in the last decade, but improvements are slowing down due to the end of traditional transistor scaling [39]. As a result, optimally mapping these algorithmic workloads onto the hardware resources becomes vital for efficient inference. To this extent, multiple design space exploration (DSE) frameworks have been developed that model the hardware architecture at a higher level of abstraction and optimise the mapping of the workload onto the architecture [14, 29, 34]. Because these frameworks rely on an analytical cost model to estimate the hardware performance, they are orders of magnitude faster than actually running the workload on the hardware.

Combining both NAS and these frameworks gives researchers detailed insights. The outputs of the framework can be used as an optimisation objective for NAS, but also give a detailed breakdown of their mapping cost in general. They also allow researchers to optimise their neural networks for different hardware architectures without needing physical access to them. In turn also providing feedback not only on the design of the neural network, but also on the performance of the different accelerators.

2. Related Work

In this section we give an overview of existing work in the fields of NAS and DSE.

2.1. Neural Architecture Search

NAS and all its related facets have been greatly studied during the past few years. Many different types of algorithms have been implemented to serve the purpose of the algorithm that drives the search process. Popular options include genetic algorithms [20, 22, 35, 45], reinforcement learning (RL) [38, 50], differentiable search [2, 19] and many other types of optimisation algorithms such as Bayesian optimisation and particle swarm optimisation [8, 32, 43].

Due to NAS having the possibility to take into account multiple optimisation objectives, multi-objective (meaning two or more objectives to be satisfied by the NAS algorithm) have almost become standard practice in NAS research. Up until now, hardware and performance related metrics includes model size, amount of compute operations needed to run the neural network (MACs or FLOPs), consumed energy or latency. A first option for estimating the hardware related costs are metrics that don't require any physical type of hardware. These are metrics such as multiply-accumulate operations (MACs) or floating point operations (FLOPs) that can be computed or model size, which is a characteristic of the neural network [2, 9, 22, 23, 48].

NAS approaches that are aware of metrics involving hardware performance require additional steps to provide these metrics to the algorithm. We can distinguish different strategies in the NAS experiments involving hardware [1]. One possibility is adding the target hardware in the loop and measuring the metrics the researchers are interested in. This is an approach that is often used for obtaining the total latency when running the neural network on the the device [18, 38] but can be used for energy as well [11]. An alternative to this strategy is using a pre-defined lookup table (LUT) to get the latency for a specific neural network architecture on a given type of hardware [15, 44, 49]. Another widely used technique is building and training prediction models. These prediction models are generally based on a wide variety of machine learning (ML) algorithms, ranging from multilayer perceptrons (MLPs) and other types of neural networks [26, 27, 42] to random forests [16]. A final category consists of making an (often rather rough) estimation based on analysing the operations in the neural network with the option of adding some form of mathematical term to account for the specifics of the hardware [24, 28].

NAS can also be used as part of a hardware-software co-design method. Since these methods are also modelling (part of) the accelerators they need accurate ways to predict their performance. Similar methods to the ones mentioned above are used, including machine learning [5, 10], FLOPs [18] and analysing the operators with a compensation factor for memory based on on device measurements [25].

2.2. NAS with Design Space Exploration Simulators

The MAESTRO framework [14], a tool for deep neural network (DNN) accelerator analysis, has been used in multiple works to provide energy and latency estimations to NAS searches [21, 46]. Timeloop [34] similarly predicts energy and latency performance for mapping of DNNs onto accelerator architectures. While these frameworks provide more insight than simply counting the number of MAC operations or the DNN model size, modelling algorithmic workloads is cumbersome, their underlying architectural templates have limited expressiveness and their latency modelling is simple.

The ZigZag exploration framework [29] provides mapping optimisations and accurate hardware performance estimations through an enhanced latency model [30] that takes into account on-chip memory stalls due to limited bandwidth.

In this work we will investigate the results of combining NAS with a state-of-the-art HW mapping and DSE framework. Our contributions are as follows:

- We propose a NAS approach that searches the best neural network architecture for a specific type of hardware accelerator on a constrained dataset.
- We implement the use of a DSE that allows the exploration of different types of neural network accelerators.
- We optimise for either energy or latency.
- We use this method to perform experiments that offer understanding of optimal neural network choices for typical hardware architectures.

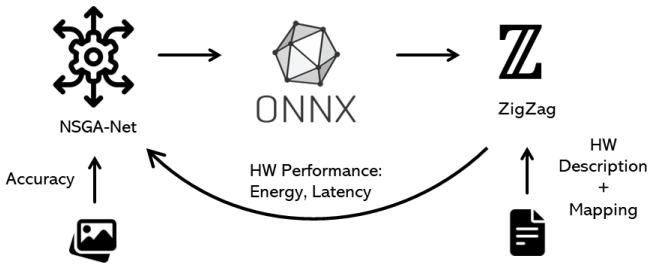
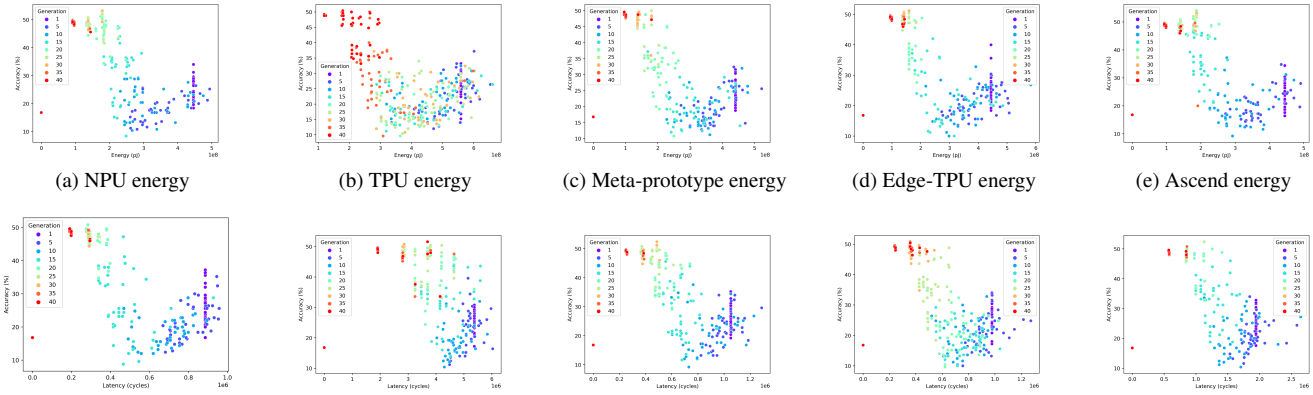


Figure 1. Schematic overview of our NAS approach that incorporates the custom hardware modelling tool ZigZag.

3. Approach

This section gives an overview of our choices for the development of our proposed NAS pipeline, shown in figure Figure 1.



(a) NPU energy (b) TPU energy (c) Meta-prototype energy (d) Edge-TPU energy (e) Ascend energy

(f) NPU latency (g) TPU latency (h) Meta-prototype latency (i) Edge-TPU latency (j) Ascend latency

Figure 2. Overview of results of NAS searches for different objectives for different hardware layouts for the *sunflower* dataset.

3.1. NAS Approach

In our previous work [9] we focused on extending NSGA-Net [22] with a hot-start based on MobileNetV2. NSGA-Net is an implementation of the NSGA-II [7] algorithm, a widely adopted genetic algorithm, for NAS. The algorithm optimises for accuracy and the size of the neural network. Model size can be one of the limiting factors for embedded devices. It also has the advantage of being a parameter that can be quite easily computed and introduces very little additional overhead. Other options, such as latency or consumed energy, can be obtained by choosing a hardware in the loop approach. This, however, requires the physical hardware, and can take a long time because of compilation and deployment. Hence, DSE frameworks are an attractive alternative for hardware-related performance metrics, as they are fast and only require an architectural template at a high level of abstraction.

3.2. Hardware Performance Modelling

This work integrates the ZigZag DSE framework into our NAS approach. This has several key benefits:

- Easy interfacing through ZigZag's Open Neural Network Exchange (ONNX) front-end.
- Ability to estimate performance for a large number of popular accelerators through provided architectural templates.
- Fast, yet accurate hardware performance estimation through analytical modelling, including a detailed latency model.

3.3. Dataset

We use two datasets as we want to compare the differences between academic datasets and industrial ones. Previous research on pruning [33] has shown that constrained problems have a higher reduction potential than general academic datasets. NAS doesn't just prune channels, it allows to change the whole neural network architecture. We opted to use an application-specific dataset that was used to steer a drone towards sunflowers using on board processing [6] and the well-known CIFAR-10 dataset. Both are RGB classification datasets and contain six and ten classes respectively. The main difference is that the sunflower dataset contains only one type of object compared to the ten classes of CIFAR-10. Which means that the sunflower dataset is much less diverse compared to CIFAR-10.

4. Experiments

We conducted various experiments to gain insight in the impact of the different parameters that can be changed for our NAS pipeline. In this section we present our chosen configurations. We will also share our training set-up.

4.1. Training Set-up

For the training parameters we follow the original training schedule for NSGA-Net, but train each proposed neural network for 20 epochs, employing an early stopping training strategy. We use a batch size of 250 images with dimension 32 by 32 for both datasets. Our parameters for the genetic algorithm are a population size of 30 and an offspring of 40 neural networks. We let the genetic algorithm evolve for 40 generations.

DNN accelerators quantise the network parameters and activations to lower (typically fixed) precision, as opposed to the floating point representation used during training. As a consequence, the model size and required memory bandwidth is reduced. This leads to better latency and energy, while having limited to no impact on inference accuracy [12, 41]. For this work, we rely on post-training quantisation and encode parameters and activations to the

common 8 bit for parameters and activations, and 16 bit for partial sums of the channel reduction.

4.2. Hardware Architectures

The ZigZag framework allows users to model different accelerators designed for various purposes. Each architecture consists of an n-dimensional array of processing elements (PEs), is connected to a hierarchy of memories and employs a specific dataflow [3]. The PE array size, memory interconnections, memory capacities, memory read and write costs and memory bandwidths differ for each architecture, impacting hardware performance for diverse DNN topologies. Five different architectures are encoded to study the architecture impact: Tesla NPU [37], Ascend [17], TPU [13], Edge-TPU [47] and Meta-prototype [36].

4.3. ZigZag Output

We configure ZigZag to give us a detailed breakdown of the hardware related performance. When using the accelerators for real-life embedded applications there are often two important metrics: energy consumption and latency. The output of ZigZag contains a detailed breakdown of the consumed energy and the latency of the discovered neural networks. For our NAS approach we propose to focus the optimisation for our bi-objective search on the parameters *latency_with_onloading_with_offloading* and *energy_total*. *latency_with_onloading_with_offloading* is the complete latency cycle of the neural network, with all time spent on onloading and offloading data to the accelerator included. The parameter *energy_total* represents all energy used by executing the neural network on the accelerator, including operational energy and energy spent on operations related to memory. The analytical breakdown produced by ZigZag will allow us to study the detailed parts that make up the global parameters.

5. Results

In this section we present and discuss our results for energy and latency on the sunflower dataset and the CIFAR-10 dataset respectively.

5.1. Optimising for Energy Consumption

Our first group of experiments aims to maximise accuracy while minimising energy consumed for all DNN accelerators introduced in section 4 for the sunflower dataset. Figure 2 shows the evolution of the NAS process over a selected number of generations.

In order to show a stable and outlier-free trend for each accelerator, we plot the median of each generation for every accelerator type in Figure 3. We conclude that while energy consumption decreases during the optimisation, accuracy actually increases significantly on such a task-specific dataset.

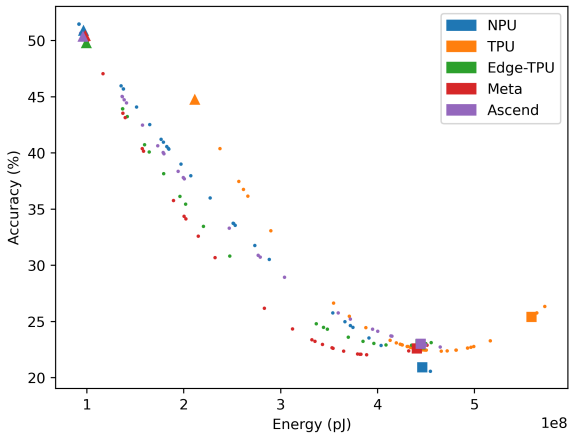


Figure 3. Evolution of median neural network performance over 40 generations, optimised for energy. □ denotes generation 1 and △ generation 40.

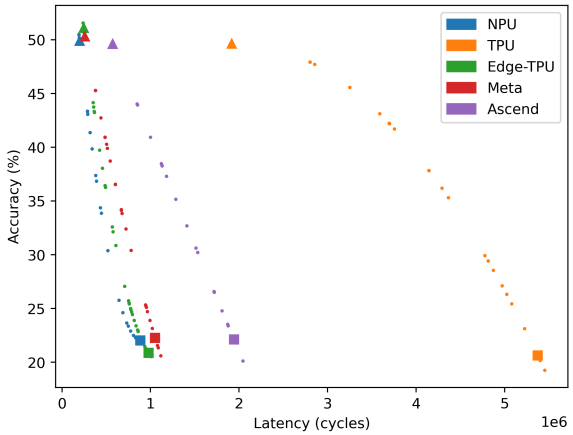


Figure 4. Evolution of median neural network performance over 40 generations, optimised for latency. □ denotes generation 1 and △ generation 40.

We calculate the reduction for each accelerator compared to the MobileNetV2-inspired hot-started neural networks of the first generation. Our NAS approach is able to reduce the energy consumption up to 4.85 times (see Table 1).

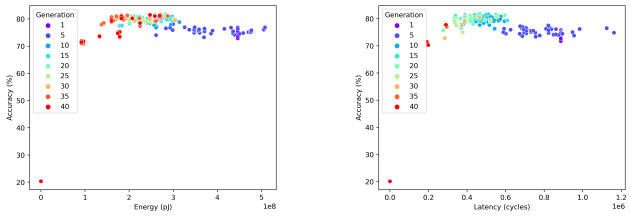
5.2. Optimising for Latency

The second type of experiments has the goal of maximising accuracy while minimising latency for all DNN accelerators introduced in section 4 for the sunflower dataset. The bottom row of Figure 2 illustrates this NAS process.

We compute the median for every generation and show

Hardware Type	Energy Gen 1 (10^8 pJ)	Energy Gen 40 (10^8 pJ)	Reduction Factor	Latency Gen 1 (10^6 cycles)	Latency Gen 40 (10^6 cycles)	Reduction Factor
NPU	4.46	0.92	4.85	0.89,	0.19	4.68
TPU	5.59	1.80	3.11	5.37	3.70	1.45
Edge-TPU	4.45	0.95	4.67	0.98	0.36	2.72
Meta	4.41	0.96	4.59	1.05	0.25	4.20
Ascend	4.458	0.94	4.76	1.94	0.57	3.40

Table 1. Comparison of best results, focusing on energy vs accuracy trade-off and latency vs accuracy trade-off for the *sunflower* dataset.



(a) CIFAR-10 NPU energy (b) CIFAR-10 NPU latency

Figure 5. Results of NAS searches for different objectives for NPU for the *CIFAR-10* dataset.

the results in Figure 4. These results show a significant decrease in latency, whilst still achieving an increase in accuracy on the sunflower dataset.

Table 1 illustrates the reduction in latency relative to the MobileNetV2-inspired neural network architectures in generation 1. We are able to achieve a speed up of up to 4.68 times.

5.3. Breakdown of Energy and Latency

ZigZag enables us to give a detailed breakdown of the energy consumed by the accelerator, as well as the breakdown of the latency. Figure 6a shows the breakdown of energy consumed by memory and compute operations for generation 1 and 40. We conclude that most energy is used for accessing the memory of the accelerator. Figure 6b breaks down the different aspects of latency by level of dataloading for each accelerator. We remark that for some accelerators, like the TPU, the latency introduced by the dataloading only accounts for a small percentage op the total latency. For other types of accelerators the difference is less pronounced.

5.4. CIFAR-10 Experiments

Figure 5 depicts the progression of our NAS approach for the more diverse dataset CIFAR-10 for both energy and latency on the NPU accelerator. This shows that while our hardware-related optimisation metrics decrease, accuracy remains relatively the same, even decreasing a little towards the last generations. Analysing the breakdown of the latency and the energy in Figure 6d and Figure 6c still show a reduction in energy and latency. However, compared to the NPU results described in Table 1, the reduction factor is smaller, around 2 times.

6. Discussion

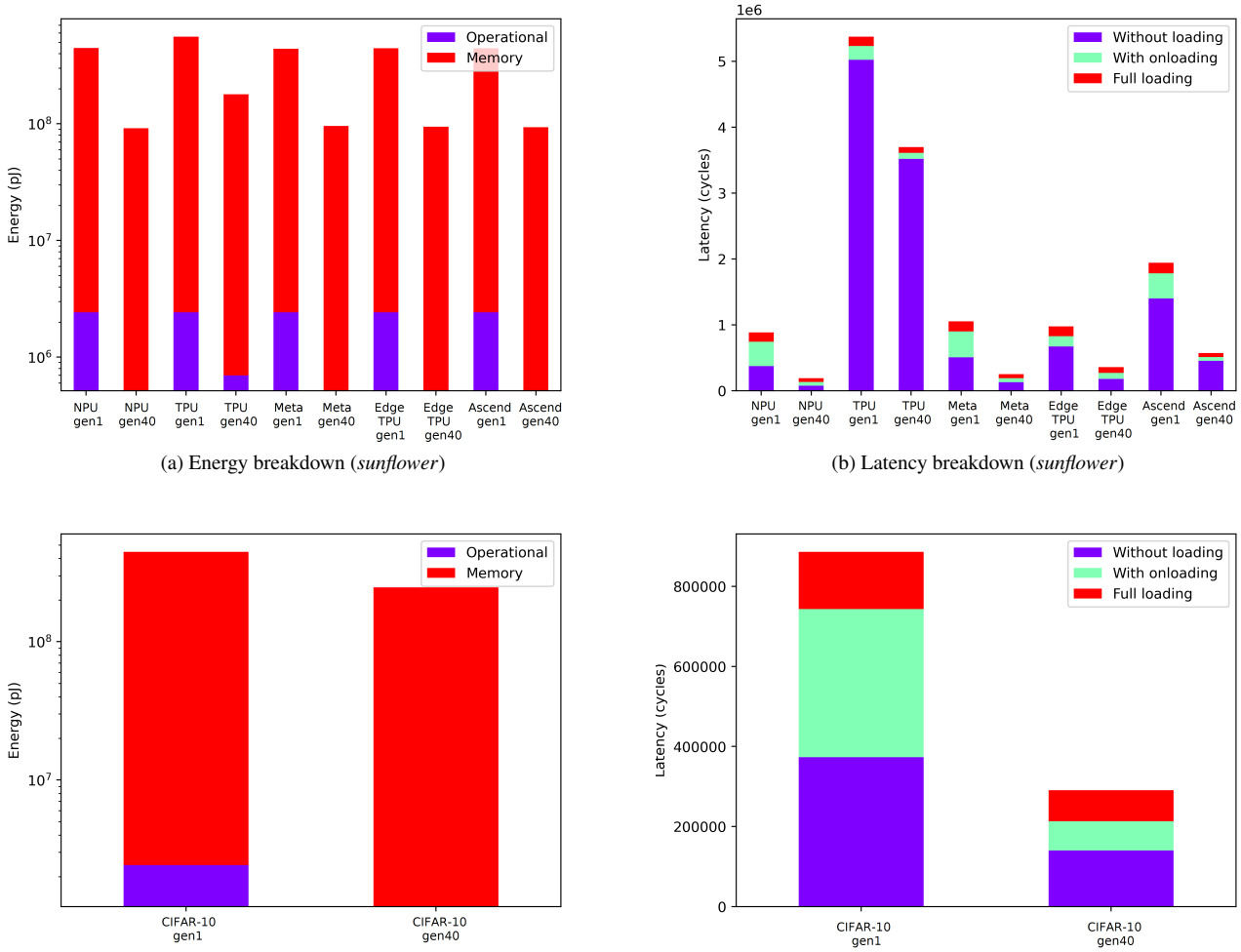
From the results of our experiments described above, a few remarkable observations can be made.

We firstly observe that the hardware architecture choice does not influence the accuracy of the final optimised model, for each of the hardware types the final sunflower classification accuracy reaches around 50% on the validation dataset. However, because of to the early stopping technique used for training the neural networks, the final accuracy after further training might still be higher.

Despite reaching similar accuracy, the reduction factors we are able to achieve vary depending on the type of accelerator. This can be explained by the differences in hardware layout. For example, different types of accelerators have registers with different dimensions. Registers that are bigger introduce a higher energy cost per operation, so when using registers that are not optimal this can introduce a higher energy cost compared to using an accelerator with registers that can be optimally used. Our experiments show the differences in performance for the accelerators and can aid in the choice of the most suitable one.

We also remark that for a constrained dataset accuracy increases drastically during the search process. This happens despite the hot-start from the proven MobileNetV2 architecture. As mentioned before, we only trained the neural networks for 20 epochs. The final accuracy after a full training cycle is expected to be higher. Towards the end of the evolutionary search process the neural networks have become smaller with less parameters to optimise. Smaller neural networks are able to be trained more quickly than larger counterparts and generalise better.

Our experiments on the general purpose CIFAR-10 show a different evolution in accuracy. The accuracy remains more or less constant across the different generations. The first generation already reaches a relatively high accuracy achieved by training for 20 epochs. This illustrates the suitability of MobileNetV2 for this dataset. With a slight drop in accuracy , as shown in Figure 5, we are still able to re-



(a) Energy breakdown (*sunflower*)

(b) Latency breakdown (*sunflower*)

(c) Energy breakdown (*CIFAR-10*)

(d) Latency breakdown (*CIFAR-10*)

Figure 6. Breakdown results of NAS searches for energy and latency.

duce latency and energy consumption. This is in line with results from similar experiments for pruning [33].

Lastly we also note that using either latency or energy as an optimisation metric can lead to significant reductions for these optimisation metrics.

7. Conclusion

In this paper we explored the benefits of expanding a NAS algorithm with a state of the art DSE framework. The DSE framework provides accurate hardware performance metrics, such as latency and energy, for DNN accelerators. Our experiments have shown that using these metrics leads to a significant reduction in consumed energy and latency, because the NAS optimisation can exploit the hardware-specific properties. Additionally, we note that these reductions are on average more pronounced for a task-specific dataset. As future work, we plan to investigate the possibility to also involve parameters of the actual hardware architecture into the genetic optimisation, hence transforming this method to hardware/software co-optimisation finding simultaneously the optimal neural network architecture and the best hardware architecture for a specific application.

Acknowledgements

This work has been supported by the Flemish Government under the AI Research Program. The authors thank the company MAGICS for supplying the sunflower dataset.

References

[1] Hadjer Benmeziane, Kaoutar El Maghraoui, Hamza Ouarnoughi, Smail Niar, Martin Wistuba, and Naigang Wang. A Comprehensive Survey on Hardware-Aware Neu-

ral Architecture Search, Jan. 2021. arXiv:2101.09336 [cs]. 2

[2] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. *arXiv:1812.00332 [cs, stat]*, Feb. 2019. arXiv: 1812.00332. 2

[3] Yiran Chen, Yuan Xie, Linghao Song, Fan Chen, and Tianqi Tang. A survey of accelerator architectures for deep neural networks. *Engineering*, 6(3):264–274, 2020. 1, 4

[4] Yu-Hsin Chen, Tushar Krishna, Joel S Emer, and Vivienne Sze. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE journal of solid-state circuits*, 52(1):127–138, 2016. 1

[5] Kanghyun Choi, Deokki Hong, Hojae Yoon, Joonsang Yu, Youngsok Kim, and Jinho Lee. DANCE: Differentiable Accelerator/Network Co-Exploration. *arXiv:2009.06237 [cs, stat]*, Sept. 2020. arXiv: 2009.06237. 2

[6] Hulens D, Van Ranst W., Cao Y., and Goedemé T. The autonomous pollination drone. *Proceedings of the 2nd Winter IFSA Conference on Automation, Robotics & Communications for Industry 4*, 2:38–41, Feb. 2022. 3

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. 3

[8] David Eriksson, Pierce I.-Jen Chuang, Samuel Daulton, Peng Xia, Akshat Shrivastava, Arun Babu, Shicong Zhao, Ahmed Aly, Ganesh Venkatesh, and Maximilian Balandat. Latency-Aware Neural Architecture Search with Multi-Objective Bayesian Optimization, June 2021. arXiv:2106.11890 [cs]. 2

[9] Lotte Hendrickx, Wiebe Van Ranst, and Toon Goedemé. Hot-started nas for task-specific embedded applications. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1970–1977, 2022. 2, 3

[10] Deokki Hong, Kanghyun Choi, Hye Yoon Lee, Joonsang Yu, Noseong Park, Youngsok Kim, and Jinho Lee. Enabling hard constraints in differentiable neural network and accelerator co-exploration. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, DAC '22, pages 589–594, New York, NY, USA, Aug. 2022. Association for Computing Machinery. 2

[11] Chi-Hung Hsu, Shu-Huan Chang, Jhao-Hong Liang, Hsin-Ping Chou, Chun-Hao Liu, Shih-Chieh Chang, Jia-Yu Pan, Yu-Ting Chen, Wei Wei, and Da-Cheng Juan. Monas: Multi-objective neural architecture search using reinforcement learning, 2018. 2

[12] Sitao Huang, Aayush Ankit, Plinio Silveira, Rodrigo Antunes, Sai Rahul Chalamalasetti, Izzat El Hajj, Dong Eun Kim, Glaucimar Aguiar, Pedro Bruel, Sergey Serebryakov, et al. Mixed precision quantization for reram-based dnn inference accelerators. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, pages 372–377, 2021. 3

[13] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12, 2017. 1, 4

[14] Hyoukjun Kwon, Prasanth Chatarasi, Vivek Sarkar, Tushar Krishna, Michael Pellauer, and Angshuman Parashar. Maestro: A data-centric approach to understand reuse, performance, and hardware cost of dnn mappings. *IEEE Micro*, 40(3):20–29, 2020. 1, 2

[15] Jooyeon Lee, Junsang Park, Seunghyun Lee, and Jaeha Kung. Implication of Optimizing NPU Dataflows on Neural Architecture Search for Mobile Devices. *ACM Transactions on Design Automation of Electronic Systems*, 27(5):48:1–48:24, June 2022. 2

[16] Zhuojin Li, Marco Paolieri, and Leana Golubchik. Inference Latency Prediction at the Edge, Oct. 2022. arXiv:2210.02620 [cs]. 2

[17] Heng Liao, Jiajin Tu, Jing Xia, Hu Liu, Xiping Zhou, Honghui Yuan, and Yuxing Hu. Ascend: a scalable and unified architecture for ubiquitous deep neural network computing: Industry track paper. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 789–801. IEEE, 2021. 4

[18] Ji Lin, Wei-Ming Chen, Yujun Lin, John Cohn, Chuang Gan, and Song Han. MCUNet: Tiny Deep Learning on IoT Devices. *arXiv:2007.10319 [cs]*, July 2020. arXiv: 2007.10319. 2

[19] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable Architecture Search. *arXiv:1806.09055 [cs, stat]*, Apr. 2019. arXiv: 1806.09055. 2

[20] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, and Gary Yen. A Survey on Evolutionary Neural Architecture Search. *arXiv:2008.10937 [cs]*, Aug. 2020. arXiv: 2008.10937. 2

[21] Bingqian Lu, Zheyu Yan, Yiyu Shi, and Shaolei Ren. A Semi-Decoupled Approach to Fast and Optimal Hardware-Software Co-Design of Neural Accelerators, Mar. 2022. arXiv:2203.13921 [cs]. 2

[22] Zhichao Lu, Ian Whalen, Vishnu Boddeti, Yashesh Dhebar, Kalyanmoy Deb, Erik Goodman, and Wolfgang Banzhaf. NSGA-Net: Neural Architecture Search using Multi-Objective Genetic Algorithm. *arXiv:1810.03522 [cs]*, Apr. 2019. arXiv: 1810.03522. 2, 3

[23] Zhichao Lu, Ian Whalen, Yashesh Dhebar, Kalyanmoy Deb, Erik Goodman, Wolfgang Banzhaf, and Vishnu Naresh Boddeti. Multi-Objective Evolutionary Design of Deep Convolutional Neural Networks for Image Classification. *arXiv:1912.01369 [cs]*, Sept. 2020. arXiv: 1912.01369. 2

[24] Xiangzhong Luo, Di Liu, Shuo Huai, Hao Kong, Hui Chen, and Weichen Liu. Designing Efficient DNNs via Hardware-Aware Neural Architecture Search and Beyond. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(6):1799–1812, June 2022. Conference Name: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2

[25] Xiangzhong Luo, Di Liu, Shuo Huai, and Weichen Liu. HSCoNAS: Hardware-Software Co-Design of Efficient DNNs via Neural Architecture Search, Mar. 2021. arXiv:2103.08325 [cs]. 2

[26] Xiangzhong Luo, Di Liu, Hao Kong, Shuo Huai, Hui Chen, and Weichen Liu. LightNAS: On Lightweight and Scalable Neural Architecture Search for Embedded Platforms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1, 2022. Conference Name: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2

[27] Xiangzhong Luo, Di Liu, Hao Kong, and Weichen Liu. EdgeNAS: Discovering Efficient Neural Architectures for Edge Systems. In *2020 IEEE 38th International Conference on Computer Design (ICCD)*, pages 288–295, Oct. 2020. ISSN: 2576-6996. 2

[28] Alberto Marchisio, Andrea Massa, Vojtech Mrazek, Beatrice Bussolino, Maurizio Martina, and Muhammad Shafique. NASCaps: A Framework for Neural Architecture Search to Optimize the Accuracy and Hardware Efficiency of Convolutional Capsule Networks. *arXiv:2008.08476 [cs, stat]*, Aug. 2020. arXiv: 2008.08476. 2

[29] Linyan Mei, Pouya Houshmand, Vikram Jain, Sebastian Giraldo, and Marian Verhelst. Zigzag: Enlarging joint architecture-mapping design space exploration for dnn accelerators. *IEEE Transactions on Computers*, 70(8):1160–1174, 2021. 1, 2

[30] Linyan Mei, Huichu Liu, Tony Wu, H Ekin Sumbul, Marian Verhelst, and Edith Beigne. A uniform latency model for dnn accelerators with diverse architectures and dataflows. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 220–225. IEEE, 2022. 2

[31] Bert Moons, Roel Uytterhoeven, Wim Dehaene, and Marian Verhelst. 14.5 envision: A 0.26-to-10tops/w subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm fdsoi. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 246–247. IEEE, 2017. 1

[32] Sergiu Cosmin Nistor and Gabriela Czibula. IntelliSwAS: Optimizing deep neural network architectures using a particle swarm-based approach. *Expert Systems with Applications*, 187:115945, Jan. 2022. 2

[33] Tanguy Ophoff, Cedric Gullentops, Kristof Van Beeck, and Toon Goedemé. Investigating the Potential of Network Optimization for a Constrained Object Detection Problem. *Journal of Imaging*, 7(4), 2021. Publisher: MDPI. 3, 6

[34] Angshuman Parashar, Priyanka Raina, Yakun Sophia Shao, Yu-Hsin Chen, Victor A Ying, Anurag Mukkara, Rangharajan Venkatesan, Brucek Khailany, Stephen W Keckler, and Joel Emer. Timeloop: A systematic approach to dnn accelerator evaluation. In *2019 IEEE international symposium on performance analysis of systems and software (ISPASS)*, pages 304–315. IEEE, 2019. 1, 2

[35] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-Scale Evolution of Image Classifiers. *arXiv:1703.01041 [cs]*, June 2017. arXiv: 1703.01041. 2

[36] H Ekin Sumbul, Tony F Wu, Yuecheng Li, Syed Shakib Sarwar, William Koven, Eli Murphy-Trotzky, Xingxing Cai, Elnaz Ansari, Daniel H Morris, Huichu Liu, et al. System-level design and integration of a prototype ar/vr hardware featuring a custom low-power dnn accelerator chip in 7nm technology for codec avatars. In *2022 IEEE Custom Integrated Circuits Conference (CICC)*, pages 01–08. IEEE, 2022. 4

[37] Emil Talpes, Debjit Das Sarma, Ganesh Venkataramanan, Peter Bannon, Bill McGee, Benjamin Floering, Ankit Jalote, Christopher Hsiong, Sahil Arora, Atchyuth Gorti, et al. Compute solution for tesla's full self-driving computer. *IEEE Micro*, 40(2):25–35, 2020. 4

[38] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. MnasNet: Platform-Aware Neural Architecture Search for Mobile. *arXiv:1807.11626 [cs]*, May 2019. arXiv: 1807.11626. 2

[39] Thomas N Theis and H-S Philip Wong. The end of moore's law: A new beginning for information technology. *Computing in Science & Engineering*, 19(2):41–50, 2017. 1

[40] Kodai Ueyoshi, Ioannis A Papistas, Pouya Houshmand, Giuseppe M Sarda, Vikram Jain, Man Shi, Qilin Zheng, Sebastian Giraldo, Peter Vrancx, Jonas Doevenspeck, et al. Diana: An end-to-end energy-efficient digital and analog hybrid neural network soc. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, volume 65, pages 1–3. IEEE, 2022. 1

[41] Marian Verhelst and Bert Moons. Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to iot and edge devices. *IEEE Solid-State Circuits Magazine*, 9(4):55–65, 2017. 3

[42] Xueying Wang, Guangli Li, Xiu Ma, and Xiaobing Feng. Facilitating hardware-aware neural architecture search with learning-based predictive models. *Journal of Systems Architecture*, 137:102838, Apr. 2023. 2

[43] Colin White, Willie Neiswanger, and Yash Savani. BANANAS: Bayesian Optimization with Neural Architectures for Neural Architecture Search. *arXiv:1910.11858 [cs, stat]*, Feb. 2020. arXiv: 1910.11858. 2

[44] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search. *arXiv:1812.03443 [cs]*, May 2019. arXiv: 1812.03443. 2

[45] Lingxi Xie and Alan Yuille. Genetic CNN. *arXiv:1703.01513 [cs]*, Mar. 2017. arXiv: 1703.01513. 2

[46] Lei Yang, Zheyu Yan, Meng Li, Hyoukjun Kwon, Liangzhen Lai, Tushar Krishna, Vikas Chandra, Weiwen Jiang, and Yiyu Shi. Co-Exploration of Neural Architectures and Heterogeneous ASIC Accelerator Designs Targeting Multiple Tasks. *arXiv:2002.04116 [cs, eess, stat]*, Feb. 2020. arXiv: 2002.04116. 2

[47] Amir Yazdanbakhsh, Kiran Seshadri, Berkin Akin, James Laudon, and Ravi Narayanaswami. An evaluation of edge tpu accelerators for convolutional neural networks. *arXiv e-prints*, pages arXiv–2102, 2021. 4

[48] Haichao Zhang, Jiashi Li, Xin Xia, Kuangrong Hao, and Xuefeng Xiao. Multi-Objective Evolutionary for Object Detection Mobile Architectures Search, Nov. 2022. arXiv:2211.02791 [cs]. 2

[49] Xindong Zhang, Hui Zeng, and Lei Zhang. Efficient Hardware-Aware Neural Architecture Search for Image

Super-Resolution on Mobile Devices. In Lei Wang, Juergen Gall, Tat-Jun Chin, Imari Sato, and Rama Chellappa, editors, *Computer Vision – ACCV 2022*, volume 13843, pages 409–426. Springer Nature Switzerland, Cham, 2023. Series Title: Lecture Notes in Computer Science. 2

[50] Barret Zoph and Quoc V. Le. Neural Architecture Search with Reinforcement Learning. *arXiv:1611.01578 [cs]*, Feb. 2017. arXiv: 1611.01578. 2