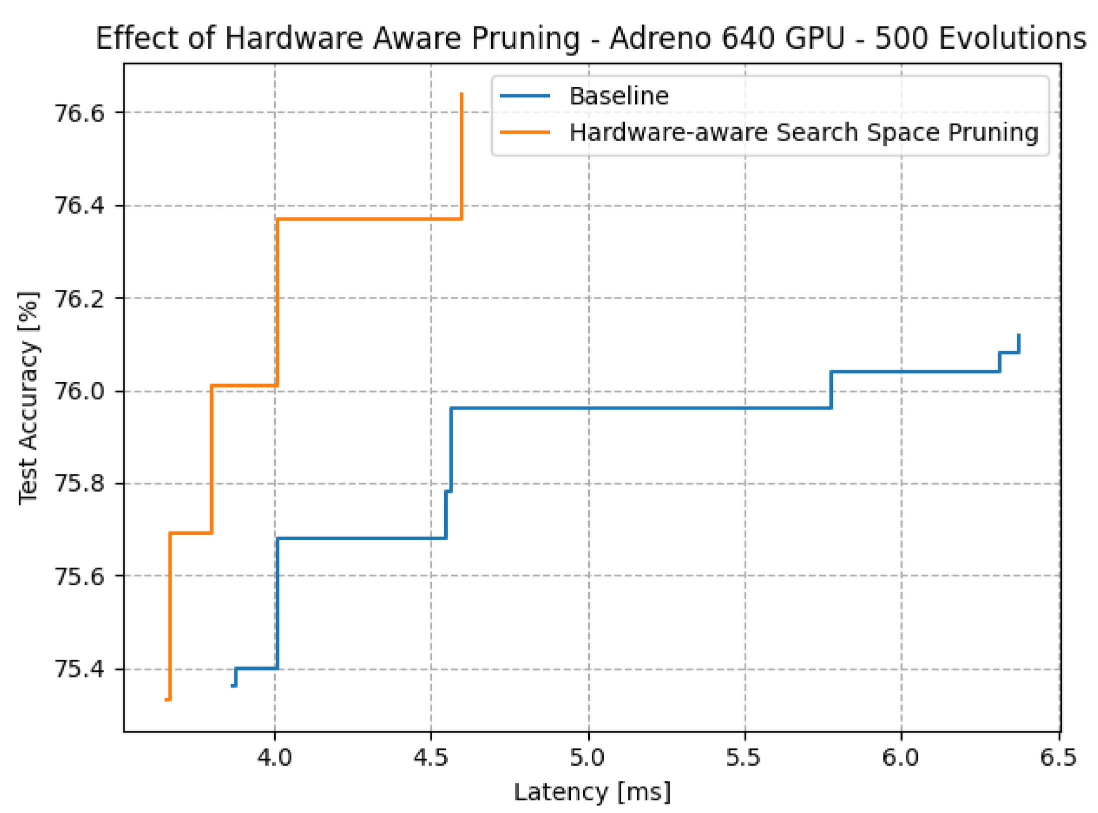
Search Space Optimization in Hardware-Aware Neural Architecture Search

Dennis Rieber, Joschka Theissen*, Thomas Elsken
Bosch Research, Germany *RWTH Aachen University, Germany

BOSCH

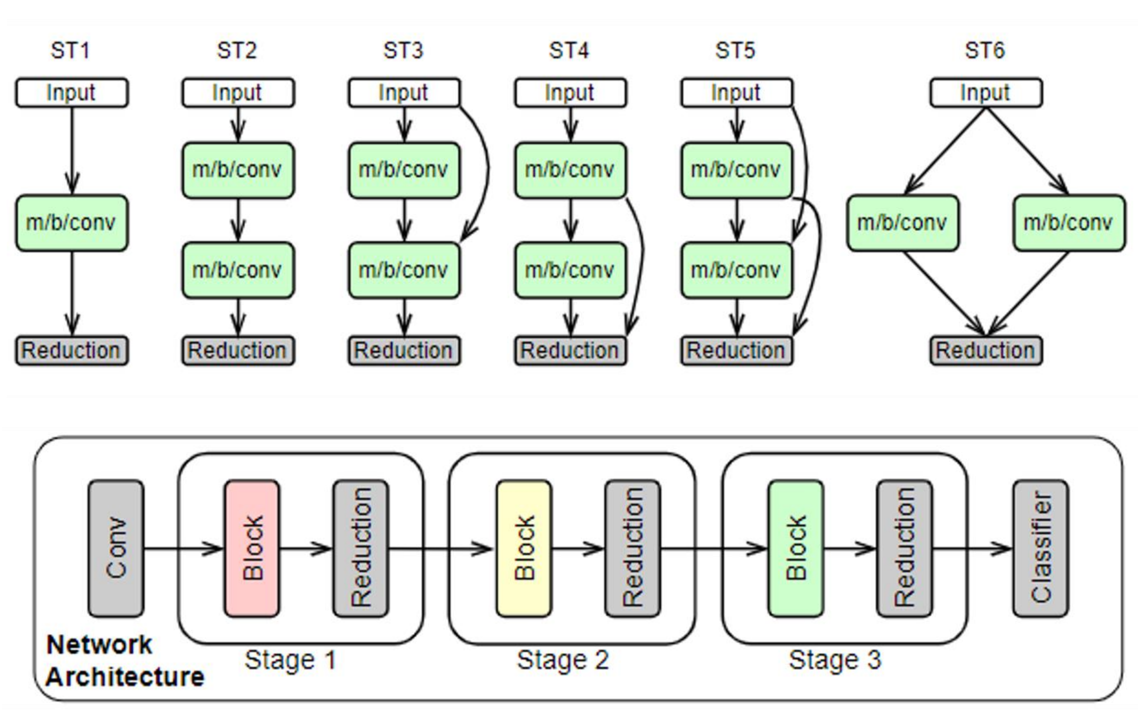
Motivation



- A search space design that is not aware of the target hardware (HW) architecture can lead prolonged search time
- From experience, **specific HW-architectures can execute some operators more efficiently than others**
- Only searching in the part of the search space with the most promising candidates can reduce the overall search effort

BLOX NAS Benchmark

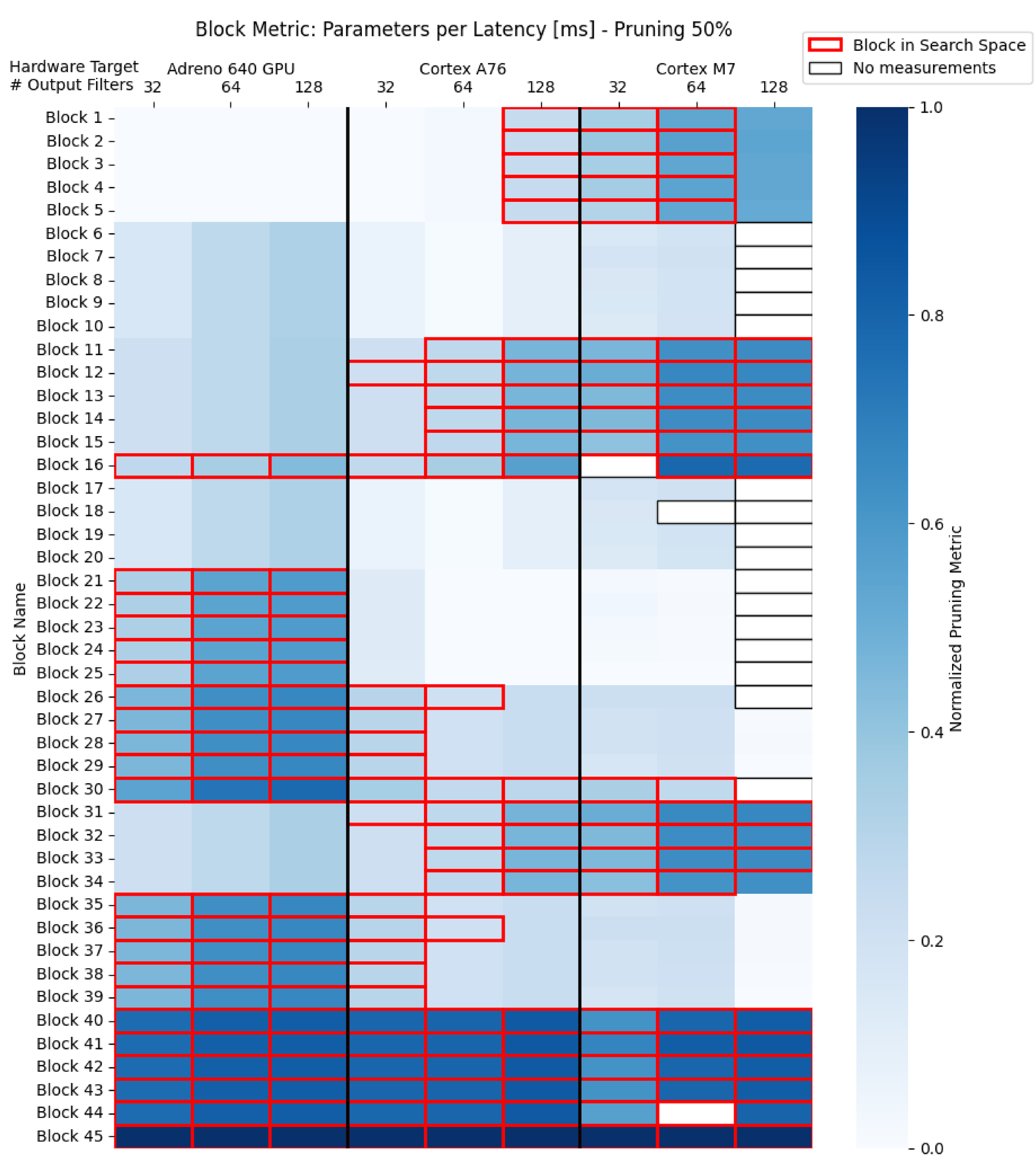
- We use the **BLOX NAS Benchmark [5]** as a baseline search space. It contains ~90k possible architectures. A DNN is constructed from three subsequent blocks, where each block is one of 45 sub-networks.



- To implement the search, we extend a simple evolutionary NAS method [2] to multi-objective optimization by using non-dominated sorting [3] for ranking candidates.
- All networks are **evaluated on CIFAR-100** and the shown accuracies are top-1 accuracies on the test set.
- Latency values are obtained by actual measurements on **ARM Cortex-M7** and estimates using Microsoft NN-Meter [6] on **ARM Cortex-A76 and Qualcomm Adreno 640 GPU**.

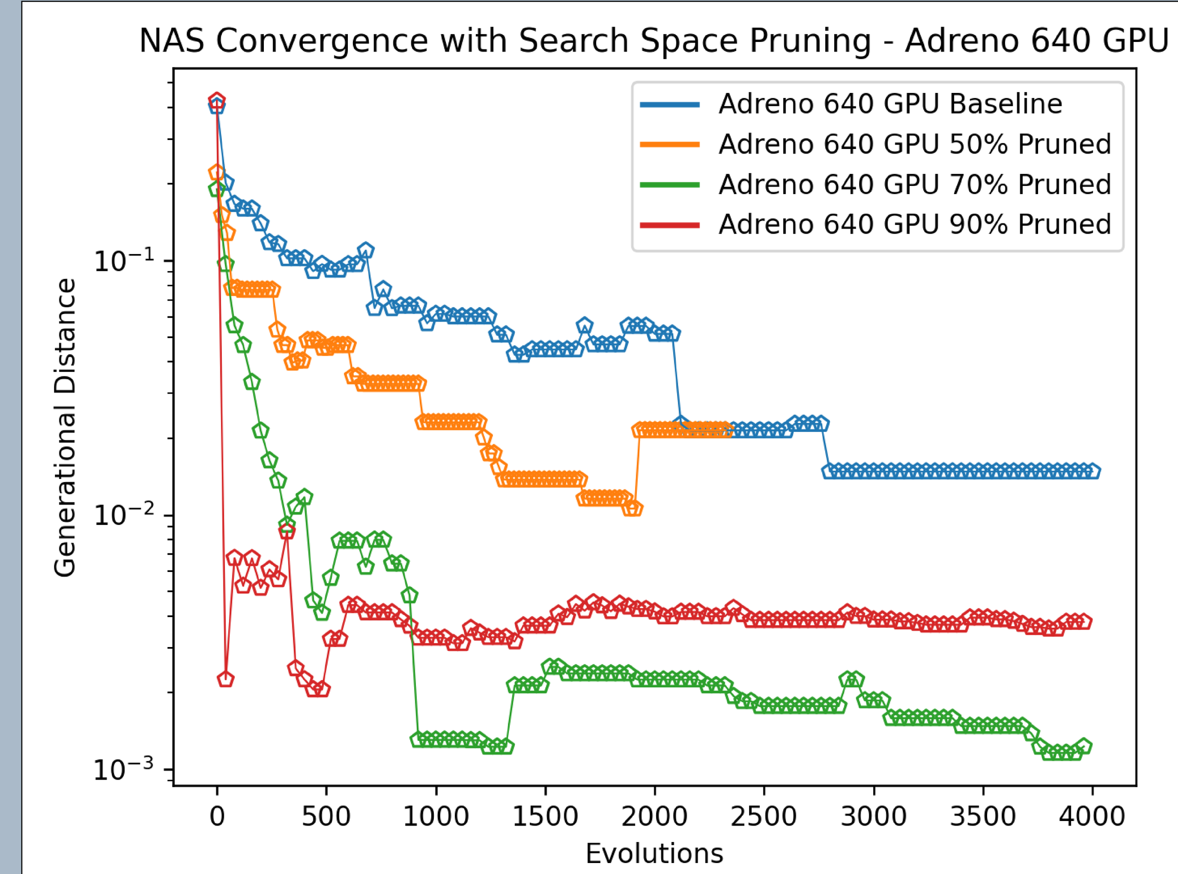
Search Space Pruning

- We investigate search space pruning **prior to the search**.
- An **"efficiency" metric** is used to evaluate all building blocks
- The metric aims to quantify the computational effort necessary for a forward-pass of a single block and its potential to improve the accuracy of the network.
- This metric is then used to prune the search space, before the search is started.
- Evaluated metric: **parameters/latency**: this metric is fully hardware-aware while requiring **no training**



Convergence Analysis

- We employ **the generational distance**, which measures the distance w.r.t. the theoretically optimal pareto front to evaluate convergence speed in different search spaces
- **Pruning significantly accelerates the speed of convergence**
- **Very aggressive pruning** (i.e., 90%) leads to very fast early convergence, but eventually to **sub-optimal** result
- Comparing HW-aware metric-based pruning with random pruning: significantly worse performance than HW-aware pruning as well as un-pruned, baseline search space

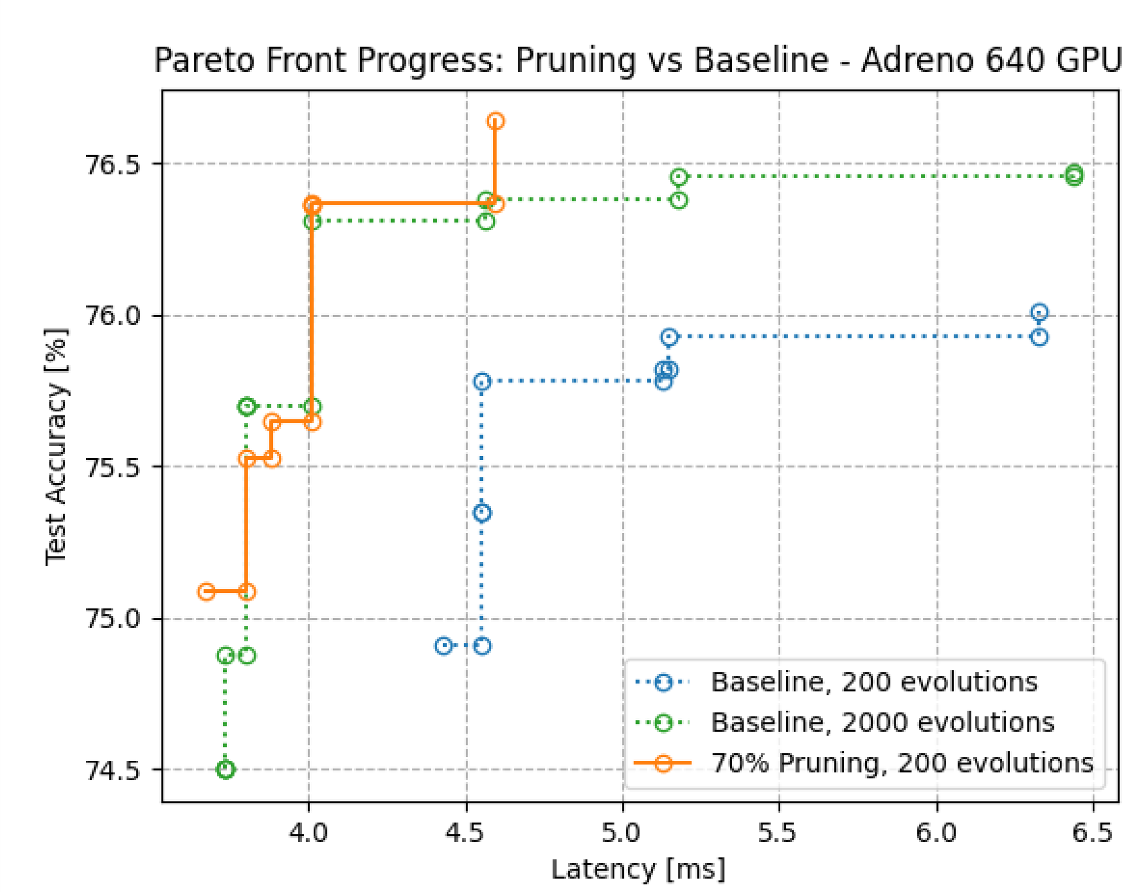


- The following table shows **the generational distance** at different stages during the evolution for **three different devices** and **different pruning strategies**.

Hardware	Pruning	Evolutions 10	100	250	500	1000	2000	3000
Adreno 640	0%	0.351	0.175	0.139	0.110	0.078	0.058	**0.015**
	70%	0.200	0.041	0.017	0.004	**0.001**	0.002	0.002
	90%	0.028	0.012	0.010	**0.006**	0.006	0.007	0.006
	Random 70%	0.167	0.099	**0.025**	0.077	0.083	0.088	0.089
Cortex A76	0%	0.363	0.183	0.128	0.079	0.061	**0.019**	0.033
	50%	0.342	0.112	0.030	0.031	0.027	0.011	**0.003**
	70%	0.242	0.047	0.035	0.005	0.002	0.001	**0.000**
	90%	0.056	0.013	0.015	0.002	**0.001**	0.001	0.001
Cortex M7	0%	0.304	0.237	0.166	0.111	0.097	**0.000**	0.000
	50%	0.054	0.039	0.046	0.026	0.003	**0.000**	0.000
	70%	0.334	0.064	0.029	0.003	**0.000**	0.000	0.000
	90%	0.012	0.035	**0.002**	0.002	0.002	0.002	0.002

Post-Pruning Solution Space

- NAS **yields similar results after just 200 evolutions** on a pruned search space compared to 2000 evolutions on the baseline search space, while significantly outperforming the baseline search space at 200 evolutions:



- Looking at all discovered architectures reveals, that **search space pruning shifts the discovered space towards more promising regions** with architectures of lower inference latency and higher test accuracy:

Conclusion

- Search Space Pruning demonstrates promising results across different hardware targets
- **Drastically reduced search time**
- NAS on unpruned search spaces also discovers as good or better solutions in theory, but doesn't reach these levels within reasonable searching time
- Trade-off between quality and time-to-solution

However:

- BLOX NAS is a "closed" problem, with a finite set of reasonably good solutions on a well-known task.
- In real-world applications, even the coarse structure of a good solutions in not always known.
- Thus "open problems" with potentially infinite search spaces are a reality that needs to be faced.
- How well offline-methods work on open problems remains to be investigated.

References

[1] Hadjer Benmeziane, et al. "A comprehensive survey on hardware-aware neural architecture search", IJCAI 2021
[2] Real et al., "Regularized Evolution for Image Classifier Architecture Search", AAAI 2019
[3] Dennis Rieber, et al. "Joint program and layout transformations to enable convolutional operators on specialized hardware based on constraint programming.", Transactions on Architectures and Code-Optimizations (TACO), volume 19, 2021.
[4] Colin White, et al. "Neural architecture search: Insights from 1000 papers." arXiv preprint, 2023
[5] Thomas Chau et al. "BLOX: Macro Neural Architecture Search Benchmark and Algorithms", 2022
[6] Li Lyna Zhang et al. "Nn-METER: Towards Accurate Latency Prediction of DNN Inference on Diverse Edge Devices.", GetMobile: Mobile Computing and Communications, 2022