All reviewers: *Improvement over SOTA:* Our work 1) improves results by the union of narrow-space search, 2) accounts for constraints, and 3) improves accuracy for the given constraints. The reviewers agree with the novelty of point 1. Even though reference work solves points 2 and 3 and we reuse a variant of a genetic algorithm inside the inner optimization loop of our automatic workflow (L163), we claim the uniqueness of our overall approach and results. More precisely, we provide a low effort solution to a NAS problem by decomposing it into three steps that can be solved 1) analytically, 2) by calibration measurement, and 3) by training models. We stress run times inequalities; 1 analytical evaluation (100ths per sec) $<<$ 1 calibration measurement (1sec – 100sec) $<<$ 1 model training (>1h even on P100 GPUs). Solving the NAS for one specific constraint, the left parts depicted in Fig. 3 can be evaluated efficiently (especially, the genetic algorithm does not involve model training). We assess 10'000 of models before training, where only a small subset of those is required to be trained.

Novelty: First, we extend the search range to small models (we cover five orders of magnitudes in the number of parameters) whereas reference work reports around a reference point or on a limited dynamic range. Second, we evaluate quantized models on the front of optimal models. Even though quantization and reduced precision models are established, key reference results are obtained on selected operation points. In contrast, our evaluation reveals the effects of trade-offs between quantization and NAS results for smaller networks.

Significance: In our opinion, the significance and innovation should not be solely judged on the subparts of contributions on their own but rather by the overall impact of the approach. Our method is simple and clean. It has been designed for practical industrial applications, targetting industrial experts with limited AI background and/or time to invest in AI expertise. We provide a flexible and economical design strategy that helps to reduce development time to design AI models for edge devices and impact industrial solutions.

Reviewer 1: The manual and automatic workflow are independent approaches to solve the same NAS problem and they do not interact. Fig. 4 (right), is the analog plot of Fig. 1 (right), where the "random configuration" shows the statistics of networks that are sampled in the full space, the "automatic search" are the results of running and concatenating the genetic algorithm as described in L181+. We will better clarify this in the revised version.

The closest related work is MnasNet [46] that follows a traditional NAS approach, where the feedback loop includes the expensive training step reporting a run time of 4.5 days on 64 TPUv2 devices. That is equivalent to about 640 GPUh/TPUv2 pod*24h/d*4.5d = 69'120 GPU hours. In comparison, our approach (100 selected models) leads to about 200 GPU hours (similar model size as ref.). Accounting for the explainable difference of the dataset 24x (1.2M/50k samples) but also for the training 1/20x (they: 5 epochs, we: 100 epochs) our NAS approach remains about two orders of magnitude more resource-efficient! 2) Results: all MnasNet models exceed 3.9M parameters and their latency domain spans one order of magnitude between 20ms and 140ms (MnasNet, Fig. 5). In contrast, we include the 1k up to 1M parameters range that is essential to reduce the per IoT device cost and we cover two orders of magnitude on a much weaker device in terms of latency time.

Reviewer 2: We agree, manual and automatic comparisons are challenging to compare due to the expert knowledge. In our results, we assume a best-effort approach with an experienced data scientist. All decisions taken by the expert are listed in the appendix. We stress, that for a fair comparison, the narrow-search space is common for both approaches. Knowing Fig. 6 implies you have already spent the experimenting runtime, so if your intention is to automate and shorten the overall development time you aim to not depend on the "human in the loop" at that stage. If you still do, most probably you want to extend the search space to account for new design patterns, if you do so, for a fair comparison, you should rerun both workflows on the new space. If your initial assumption was right, and you indeed have a stronger search space, we expect to improve results in both cases.

Reviewer 3: The number of sampled narrow spaces that are needed to deliver a good solution depends directly on the quality of the search space. If it would be a priori clear, or there is strong evidence that one space is considerably better than others, it would be enough to perform all considerations on that space only. However, since we do not have such evidence prior to run the experiments, we propose the aggregation of multiple subspaces. Since each definition involves paper and pen definitions and related implementations the number of considered narrow spaces must be reasonably small. We conducted all experiments with five narrow-spaces.

We do not provide theoretical guarantees. In contrast, we adopt the in-the-field practice of empirically reporting results. Our opinion is that the main contribution of our work is formulating the NAS problem in a decomposed way that allows constraint evaluations and HW measurements to rule out candidates before they are required to be trained.

We conducted the key results on CIFAR-10 since we know that this is a well-established dataset with many reference results, especially experiments conducted in IoT settings on non-standard hardware. We hope that this choice allows a common ground between HW developers, the deep learning community, and the IoT industry to better judge our achievements. Fig. 8 of our paper includes results where we applied our search strategy to additional thirteen image classification tasks, as detailed in the appendix.