Table 1: Results on validation accuracy (%) during search and network training time per epoch (seconds).

Metric	Experiment	CIFAR10	CIFAR100	MIT67	FLOWER102	SPORT
Search Validation Accuracy	θ_G	95.6	77.2	71.8	93.3	95.3
	$(\theta_G, \theta_S, \theta_C)$	95.7	77.5	72.0	93.3	95.9
Mean(Max) Train Time	θ_G	54.9(246)	43.0(216)	85.4(291)	45.4(105)	22.8(37.0)
	$(\theta_G, \theta_S, \theta_C)$	99.3(998)	82.2(711)	130(1056)	58.4(397)	20.6(93.6)

We thank the reviewers for their positive comments and suggestions. As R1 and R3 note, our main contribution is a new paradigm for NAS which, by focusing on network generators instead of single architectures, enables us to explore a much larger search space but at much lower search dimensions. R1,R2,R3 agree this work presents a meaningful extension to [14], has very solid and comprehensive evaluations, has strong BO-based search results, and is well written. *Please refer to the main paper for [1], [7], [14].*

R1. Search for stage ratio θ_S and channel ratio θ_C Results for experiments with learnable θ_S and θ_C are in Table 1. While there is a marginal increase in performance, the worst case train time substantially increases due to extreme stage and channel ratios. So, while the number of architectures sampled stays the same, the computational cost increases due to more lengthy training. We had observed this effect during preliminary experiments on CIFAR10 and decided to fix θ_S and θ_C to standard values in order to obtain competitive results at a reasonable cost. Nonetheless, even with such constrains on the search space, our HNAG is still much more expressive than most NAS search spaces.

R2,R3. Why WS and ER? Why 3 levels? We choose WS as it was shown to offer the best performances [14]. The middle level graph was modelled with ER as WS doesn't allow single node graphs: we wanted the flexibility to reduce our search space to 2 levels and represent a DARTS-like architecture. Indeed HNAG is designed to be able to emulate existing search spaces while also exploring potentially better new ones. As most existing NAS methods require 2 levels to represent, we decided to have the option of falling back to 2 (when the mid-layer becomes single node) but allow for 3 levels, enabling the creation of local clusters of operation units, which can result in more memory efficient models.

R2,R3. Fig.5 Fig. 5 compares random sampling (x-axis) vs NAS (y-axis) method performance. It is intended to showcase which methods find a better architecture than the average one [7]. Every method is run with its original code and search space. For NAGO the baseline is obtained by randomly sampling non-optimised generators. NAGO clearly achieves the largest improvement over naive random sampling than all other methods on all tasks. Looking at the y-axis, NAGO convincingly outperforms other NAS approaches on not only MIT67, but also Flowers102 and Sport8.

R2. DropPath and Auxiliary Tower(AT). We naively apply DropPath and AT, following set-ups in [1], to re-train architectures from our hierarchical search space. These techniques lead to 0.54% increase in the average test accuracy over 8 architecture samples on CIFAR10, leading to results competitive with the state-of-the-art. However, as highlighted in line 265 and acknowledged by R3, more efforts are needed to effectively adapt these techniques onto our new search space, which is fundamentally different from existing NAS spaces. Note that training DARTS' best architecture without these techniques leads to similar results as our NAGO (both 96.6 - see [7] Fig.3).

R2. Search phase set-up. Using 1 architecture sample during BO search is a reasonable compromise for the following reasons: 1) while the variance of generator hyperparameters are not always small, the top performing generators (those we are interested in) mostly have very tight performance variances (Fig.2 in paper and Fig. 4 in App.); 2) in BOHB search, we resample architectures from the same generator hyperparameter when evaluated at different budgets. The rank correlation between architecture accuracies at subsequent budgets is quite high (>0.82 - see Fig.5b in App.), indicating that a good generator remains good even when a new architecture is reevaluated at a higher budget.

R1. Optimising for memory vs FLOPS. Our choice was to focus on model optimization for low-memory devices, but we agree with R1 that FLOPs and inference times are also important metrics. Our multi-objective framework can be easily adapted (by swapping the 2nd objective) to consider these objectives and we will investigate this in future work.

R1. Hierarchical works We thank the reviewer for suggesting these works; we have added a discussion regarding them. Our main search space contribution is not with it being hierarchical per-se, but rather lies in a) its expressiveness (orders of magnitude higher than competing methods), b) its representational compactness and c) its stochasticity.

R3. Combine BOHB and MOBO. We agree that it's not trivial to combine multi-fidelity BO with multi-objective BO, especially in the BOHB framework. A potential direction is to select the Pareto set points, instead of points with highest EI value, at each budget to be evaluated for longer epochs during Successive Halving. Added discussion to the paper.

R3. Merging different channel sizes Yes, the number of channels is increased to match the highest number. We only tried pooling, but strided convolutions are a promising alternative.

R3. References We thank R3 for pointing out relevant related work, which we now reference in the paper.