We would like to thank all the reviewers for careful reading our paper, and we sincerely appreciate the constructive comments. We will individually address the questions raised by each reviewer.

Reviewer 1 **Transferability to diverse datasets.** We conducted additional experiments on the `SVHN` dataset shown in the Table to further demonstrate our algorithm. We used the same Meta network prior previously trained on the multi-scale `Imagenet` datasets and quickly adapted the meta network to `SVHN` for two epochs in less than an hour. The networks were trained for 160 epochs with cutout augmentation. We also trained the `CIFAR10` specialized network from DARTS. The adapted network achieves the best performance in our experiments and has comparable performance to other work for the model size. It is difficult to compare to [1] since they don't explicitly state their final performance on the `SVHN` dataset outside of small graphs and search smaller networks. Roughly, based on the figures, it appears to achieve about 97% accuracy in a 2 GPU hours and around 97.3% accuracy in 20 GPU hours.

Architecture	Top-1 Err	Params(M)
WideResnet [2]	1.30 ± 0.03	11.7
MetaQNN [3]	2.24	9.81
DARTS (`CIFAR10` Searched)	2.09	3.3
BASE (Multitask Prior)	2.13	3.22
BASE (`Imagenet32` Tuned)	2.07	3.29
BASE (`SVHN` Tuned)	**2.01**	3.22

Reviewer 2 **Novelty of the Bayesian View.** Our Bayesian view of architecture search is fundamentally different from SNAS [4] since our approach has the ability to naturally extend to meta architecture search while SNAS cannot. Specifically, we can transfer prior information among tasks which is critical for meta architecture search. SNAS's formulation which is motivated from the policy gradient in reinforcement learning, has no prior structure and can only be applied to a single task. Compared with SNAS, the networks found by BASE achieved comparable performance on *CIFAR10* and higher accuracy on *Imagenet* with meta architecture search.

Optimization Embedding. Optimization embedding is used for parameterization of the posterior distribution of the convolutional weights and architecture parameters in Eq. (11), $\phi^t_{\mathcal{D}}$ and $\psi^t_{\mathcal{D}}$. In the few-shot learning experiments, we fully unroll the optimization embedding while in the other experiments, we use the finite difference approximation corresponding to line 10 of algorithm 1. Compared to the vanilla independent posterior parametrization with an arbitrary neural network, optimization embedding allows us to model the dependencies between D and W *explicitly* which allows to better adaptation to new datasets.

Training Time. As with most work in the field of architecture search, our method focuses on architecture search over model training. However, with our low search times, we do recognize the increasing importance of training time, and we will add a discussion to our final paper. Our Bayesian formulation does naturally allow transfer of weights from the meta-network if we don't transfer to larger networks which could potentially eliminate the additional training time.

Transferability of Meta-Weights. We did not transfer the posterior of weights to the final networks in our experiments. Figure 4, however, does demonstrate the fast adaptation of both weights and architecture where our meta-network achieves 75% accuracy on the unseen `CIFAR10` in one epoch based on the posterior of both weights and architecture.

Clarification about few-shot learning results. Indeed, MAML and REPTILE use the same model architectures. However, they are fundamentally different algorithms for Meta-learning: MAML uses the full gradient and REPTILE uses the finite difference approximation, leading to different performances.

To be clear, DARTS, BASE (Softmax), and BASE (Gumbel) are the algorithms used to generate network architectures. They are *not* few-shot classification algorithms and in Table. 3, the actual few-shot learning with which the models are trained is conducted using MAML. Therefore, it only makes sense to directly compare against the MAML baseline. The REPTILE baseline is included only for completeness. It is true that the models used in MAML were much smaller, and we will update the paper. However, we chose to use our larger search space so we could directly compare to the the `CIFAR10` optimized network architecture from DARTS. This network, though much larger, significantly beat the original MAML paper results with the same algorithm. BASE was able to find networks in the same search space which achieved even better results. We will make this point clearer in our final paper.

Error Rates. It is definitely possible that the cell space and scaling may complicate the results, but this is still a common methodology and shouldn't invalidate our work. I believe the particular part you quoted from our paper was in reference to our found Imagenet-optimized architecture which achieved fairly high performance, but we will revise those sections to clarify our claims.

Clarity of Figures. We will be expanding the related work, clarifying the notation, and fixing those typos. To address the issue in Figure 1, the M in that figure should be L following the notation in the rest of the paper which corresponds to the number of choices of operations for each layer type.

Reviewer 3 Thank you for your appreciation and suggestions for paper refinement. We will polish the the final version of our paper following your comments.

[1] Wistuba et al. Inductive Transfer for Neural Architecture Optimization. *arXiv:1903.03536*, 2019.
[2] DeVries et al. Improved regularization of convolutional neural networks with cutout. *arXiv:1708.04552*, 2017.
[3] Baker et al. Designing neural network architectures using reinforcement learning. *ICLR* 2017.
[4] Xie et al. SNAS: stochastic neural architecture search. *ICLR* 2019.