We thank the reviewers for their insightful feedback and constructive advice. In the following, we address their concerns and questions.

Table 1: Performance of various AdaptNAS-C settings.

α	β	Source Err. (%) (CIFAR-10)		Target Err. (%) (ImageNet)	
		Valid	**Test**	**Valid**	**Test**
0.00	0.50	49.26	3.00	42.52	24.5
0.25	0.50	30.00	2.97	40.13	**24.2**
0.50	0.50	25.16	**2.50**	40.13	24.5
0.75	0.50	22.78	2.62	42.41	25.1
1.00	0.50	23.15	2.53	55.37	25.4
0.00	0.83	52.19	3.21	53.65	25.5
0.25	0.83	38.06	3.17	51.56	25.0
0.50	0.83	33.82	**2.95**	49.86	**24.7**
0.75	0.83	28.68	3.00	54.17	25.5
1.00	0.83	23.89	2.98	56.39	25.8
0.00	0.98	74.80	3.91	69.65	29.5
0.25	0.98	67.31	3.66	70.90	26.5
0.50	0.98	51.93	3.56	64.25	25.8
0.75	0.98	40.68	3.02	62.75	**25.1**
1.00	0.98	30.15	**2.93**	61.85	25.7

[R1 Self-supervision] Self-supervision is a trick used to bridge the difference of labels between CIFAR-10 and ImageNet. We also concern the impact of it. Although the calculation of supervised loss on the blended CIFAR-10-ImageNet dataset is intractable, we perform ablation study on digits domains with supervised loss to show the effectiveness of solely using AdaptNAS.
[R1 Valid Error] In previous one-shot NAS works, the validation performances are not reported, because one-shot models are not trained completely during search. We follow them in other tables. However, in Table 1, we want to show the impact of hyper-parameters of the hybrid loss during search, so we report validation errors only in this specific table.
[R1 Subset construction] FBNet and HM-NAS use 100 out of 1000 classes, which is 10% of the ImageNet, and we use fewer samples than them (3.90%). In this case, if we only use several classes, we concern the data might lose divergence and be biased. Thus, we include 50 random samples from each of all the categories.
[R2 Digits results] Thanks for checking our experiment results. We double-checked our implementation and logs and found the results are correct. The test accuracy on MNIST with SVHN as the source domain by searching on source, searching with AdaptNAS and searching on target is 99.33%, 99.21% and 99.20%, respectively. The test accuracy by searching on source is the best. We deem this abnormal result is because the source SVHN is much more complicated than the target MNIST, which is opposite to the case of NAS.
[R3 Notation] Thanks for reminding. We will replace $\boldsymbol{\alpha}$ for architecture parameter with $\boldsymbol{A}$ to distinguish with α.
[R3 Gap in practice] A popular quantitative metrics for domain gap is the $\mathcal{A}$-distance, which we approximate with a domain discriminator during search, but it is inappropriate to use it during evaluation, because networks on CIFAR-10 and ImageNet are trained with different scales and targets. However, we provide a visualization of feature alignment on digits dataset in Figure 2 of the supplementary material, which gives a side view of the generalization gap in practice.
[R3 Self-supervision] We select the Rotation task because it is a promising yet simple task and can be easily format as a typical classification task where the network takes a single augmented image as the input and predicts its rotation degree as the label. This is consistent with our theorems. However, this does not always hold in other self-supervised tasks. For example, in Jigsaw, the network takes 9 images patches as input and outputs a conditional probability density function of the spatial arrangement. In another task, Exemplar, the triplet loss is used, and explicit class labels are avoided. Considering those reasons, we selected the Rotation task.
[R3 Larger ImageNet subset] In theory, a larger subset or the complete ImageNet can be used for searching, but the computation cost will raise significantly. As we aim to leverage a small number of target samples to decrease the cross-domain generalization gap under limited computation resources, we set $\beta = 0.5$, where both domains have identical number of samples, as the upper bound and gradually decrease the ratio of target samples.
[R3 Implementation Details] We follow DARTS and PC-DARTS and train networks for 600 and 250 epochs on CIFAR-10 and ImageNet, respectively. Samples in the Imagenet subset are randomly chosen from each category. We are glad to add more details in the supplementary material.
[R4 Computation overhead] Computation overhead of the domain discriminator (an MLP) is very minor (1.31M FLOPs) comparing to those convolutional layers (40M - 80M FLOPs for sampled architectures) during search.
[R2 R4 Improvement over SOTA] Regarding MdeNAS, all of our settings consistently outperform them in terms of ImageNet top-1 error (24.7%, 25.3%, 24.3% and 24.2% comparing to 25.5% in Table 4). We used GDAS as our baseline method, because it is easy to be implemented. Being orthogonal with these related works, we can boost NAS performance from a new perspective. For example, if we adapt the architecture searched by P-DARTS with our method, we can further boost the performance to a 23.8% top-1 error on ImageNet.
[R4 Without L_D] If the number of target samples is large (e.g. $\beta = 0.5$), searching without DA loss reaches an ImageNet test error of 24.7% comparing to 24.2% reached by using DA loss. If the number of target samples reduces (e.g. $\beta = 0.83$ or 0.98), the ImageNet test error raises to 25.5% and 27.3%, and searching with DA loss still keeps a competitive error rate of 24.7% and 25.1%. With the proposed algorithm, a few target sample can already benefit the generalization a lot, which is consistent with general DA findings.
[R4 The $\alpha = 0$ case] If there are sufficient target samples (e.g. $\beta = 0.5$), a decent performance might be achieved by solely using target loss (i.e. $\alpha = 0$). But if there are increasingly few target samples (e.g. $\beta = 0.83$ and 0.98), the effect of L_d can be even more remarkable, e.g. 25.5% ($\alpha = 0$) v.s. 24.7% ($\alpha = 0.5$) under $\beta = 0.83$, as shown in Tab. 1.
[R4 Rot-4 and Rot-1] Rot-4 rotates each sample 4 times to different directions and therefore enlarges the dataset. Rot-1 randomly selects one direction for each sample per epoch and keeps the size of dataset small. According to Table 4, AdaptNAS-S tends to choose small architectures, and such small architectures (e.g. 552M FLOPs) are difficult to fit the dataset enlarged by Rot-4. Thus, AdaptNAS-S performs well with Rot-1. In the contrary, AdaptNAS-C tends to choose large architectures (e.g. 583M FLOPs), which fits these enlarged dataset easily and yields good performance with Rot-4.