A. Additional Information about Experiment Setup

This section provides additional information about the experiment setup for image classification (Sec. 4.1) and depth estimation (Sec. 4.2).

A.1. Image Classification

For the latency-guided experiment, we design the initial network based on MobileNetV3 [25] with an input resolution of 224×224, which is widely used to construct the search space. Starting from MobileNetV3-Large, we round up all layer widths to the power of 2 and add two MobileNetV3 blocks, each with the input resolution of 28×28 and 14×14. The kernel sizes of depthwise layers are increased by 2. For sampling sub-networks, we allow 9 uniformly-spaced layer widths from 0 to the full layer width and different odd kernel sizes from 3 to the full kernel size for each layer. We use the initial network of Once-for-All [11] for knowledge distillation when training the discovered network for a fair comparison with Once-for-All [11] and BigNAS [12].

For the MAC-guided experiment, following the practice of Once-for-All [11] and NSGANetV2 [38], we increase layer widths of the initial network used in the latency-guided experiment by $1.25\times$ and add one MobileNetV3 block with an input resolution of 7×7 to support a large-MAC operating condition. For sampling sub-networks, we allow 11 uniformly-spaced layer widths from 0 to the full layer width and different odd kernel sizes from 3 to the full kernel size for each layer. When training the discovered DNN, we use the initial network of Once-for-All [11] for knowledge distillation for a fair comparison with NSGANetV2 [38].

A.2. Depth Estimation

The initial network is FastDepth [42]. FastDepth consists of an encoder and a decoder. The encoder uses MobileNetV1 [40] as a feature extractor, and the decoder uses depthwise separable convolution and nearest neighbor upsampling. Please refer to FastDepth [42] for more details. For sampling sub-networks, we allow 9 uniformly-spaced layer widths from 0 to the full layer width and different odd kernel sizes from 3 to the full kernel size for each layer.

Following a common practice of training DNNs on the NYU Depth V2 dataset, we pre-train the encoder of the initial network on ImageNet. However, as shown in Table 7, this step is relatively expensive and takes 96 GPU-hours. To avoid pre-training the encoder of the discovered DNN on ImageNet again, we transfer the knowledge learned by the encoder of the initial network to that of the discovered DNN, which is achieved by the following method. We log the architecture of the best sample in each iteration of the MCD optimizer, which forms an architecture trajectory. The starting point of this trajectory is the initial DNN architecture, and the end point is the discovered DNN architecture. For training the discovered network, we start from the starting point of the trajectory with the pre-trained weights of the initial DNN and follow this trajectory to gradually shrink the architecture. In each step of shrinking the architecture, we reuse the overlapped weights from the previous architecture and train the new architecture for two epochs. This process continues until we get to the end point of the trajectory, which is the discovered DNN architecture. Then, we train it until convergence. This knowledge transfer method enables high accuracy of the discovered DNN without pre-training its encoder on ImageNet.

B. Discovered DNN Architectures

For the latency-guided experiment on ImageNet (Table 1), Table 8 shows the discovered 51ms DNN architecture of NetAdaptV2. To make the numbers of MACs of all layers as similar as possible, modern DNN design usually doubles the number of filters and channels when the resolution of activations is reduced by $2\times$. Similarly, to fix the ratio of T (Sec. 2.2) to the number of input channels, we use one value of T for each resolution of input activations and set T inversely proportional to the resolution. Ts of all depthwise layers are set to infinity to allow bypassing all the input channels. We observe that channel-level bypass connections (CBCs) are widely used in the discovered DNN. Moreover, block 12 is removed, which demonstrates the ability of CBCs to remove a layer.

For the MAC-guided experiment on ImageNet (Table 2), Table 9 shows the discovered 314M-MAC DNN architecture. We apply the same rule for setting Ts as in the latency-guided experiment. We observe that CBCs are widely used in the discovered DNN. Moreover, MobileNetV3 block 7, 8, 12, 15 are removed.

For the depth estimation experiment on the NYU Depth V2 dataset (Table 7), Table 10 shows the discovered 87ms DNN architecture of NetAdaptV2. We apply the same rule for setting Ts as in the image classification experiments. We observe that NetAdaptV2 reduces the kernel sizes of the 37-th and 40-th depthwise convolutional layers from 5 to 3, which demonstrates that the ability to search kernel sizes may improve the performance of the discovered DNN.

C. Formulation of Channel-Level Bypass Connections

The formulation of channel-level bypass connections (CBCs), $Z = max(min(C, T), M)$, can be derived by considering the case 1 to 3 in Sec. 2.2 and Fig. 3. For the case 1 ($C = T$) and 2 ($C < T$), CBCs start bypassing input channels when M becomes smaller than C ($M < C$) to maintain the number of output channels $Z = max(C, M) = C$. For the case 3 ($C > T$), CBCs start bypassing input channels when M becomes smaller than T ($M < T$) instead of C,

Index	Type	T	Kernel Size	Stride	BN	Act	Exp	DW	PW	SE
1	conv	8	3	2	✓	HS	16	-	-	-
2	mnv3 block	8	3	1	✓	RE	8	8	16	-
3	mnv3 block	16	5	2	✓	RE	48	48	20	16
4	mnv3 block	16	3	1	✓	RE	48	48	32	-
5	mnv3 block	32	7	2	✓	RE	80	80	32	32
6	mnv3 block	32	3	1	✓	RE	112	80	40	32
7	mnv3 block	32	3	1	✓	RE	64	32	16	32
8	mnv3 block	32	3	1	✓	RE	96	96	8	32
9	mnv3 block	64	5	2	✓	HS	192	192	128	64
10	mnv3 block	64	5	1	✓	HS	224	192	128	-
11	mnv3 block	64	3	1	✓	HS	128	32	48	64
12	mnv3 block	64	0	1	✓	HS	0	0	0	0
13	mnv3 block	64	3	1	✓	HS	512	256	80	256
14	mnv3 block	64	3	1	✓	HS	256	256	112	256
15	mnv3 block	64	5	1	✓	HS	512	512	64	256
16	mnv3 block	128	7	2	✓	HS	640	640	224	256
17	mnv3 block	128	7	1	✓	HS	640	384	224	256
18	mnv3 block	128	5	1	✓	HS	896	512	224	256
19	conv	128	1	1	✓	HS	1024	-	-	-
20	global avg pool	-	-	-	-	-	-	-	-	-
21	conv	1024	1	1		HS	1792	-	-	-
22	fc	-	1	1	-	-	1000	-	-	-

Table 8: The discovered 51ms DNN architecture of NetAdaptV2 on ImageNet presented in Table 1. Type: type of the layer or block. BN: using batch normalization. Act: activation type (HS: Hard-Swish, RE: ReLU). Exp: number of filters in the expansion layer or number of filters in the conv layer. DW: number of filters in the depthwise layer. PW: number of filters in the pointwise layer. SE: number of filters in the squeeze-and-excitation operation. All MobileNetV3 blocks (mnv3 block) with a stride of 1 have residual connections.

which requires replacing the C in $Z = max(C, M)$ with $min(C, T)$ and gives the formulation of CBCs.

D. Ablation Study on MobileNetV1

This section provides the ablation study on MobileNetV1 [40]. This ablation study employs the experiment setup outlined in Sec. 4.1.1 unless otherwise stated. The initial network is the largest MobileNetV1 (1.0 MobileNet-224 [40]).

D.1. Impact of Channel-Level Bypass Connections

The proposed channel-level bypass connections (CBCs) enable NetAdaptV2 to search for different network depths with marginal overhead. Table 11 shows that supporting CBCs only increases the training time of the super-network by $1.2\times$. Moreover, the ability to search network depth allows discovering DNNs with better performance. As shown in Table 11, CBCs improve the accuracy of the discovered DNN by 6.5% with the same latency.

D.2. Impact of Multi-Layer Coordinate Descent Optimizer

The proposed multi-layer coordinate descent (MCD) optimizer improves the performance of the discovered DNN while reducing the number of samples and hence the search time. In this experiment, the MCD optimizer generates 27 samples ($J = 27$) in each iteration, where J is equal to the number of layers, and each sample is obtained by randomly shrinking 4 layers ($L = 4$). Table 11 shows that the MCD optimizer reduces the time for evaluating samples by $1.9\times$ and improves the accuracy by 2.8%.

E. Estimation of CO_2 Emission

We estimate CO_2 emission based on Strubell et al. [34]. According to Table 3 in this paper, when BERT$_{base}$ is trained on 64 V100 GPUs for 79 hours, the CO_2 emission is 1438 lbs. Therefore, the ratio of CO_2 emission to GPU-hours is $\frac{1438}{64\times79} = 0.2844$. For each NAS method, we multiply its search time by this ratio to estimate its corresponding CO_2 emission.

Index	Type	T	Kernel Size	Stride	BN	Act	Exp	DW	PW	SE
1	conv	8	3	2	✓	HS	24	-	-	-
2	mnv3 block	8	3	1	✓	RE	-	24	24	-
3	mnv3 block	16	5	2	✓	RE	64	48	32	24
4	mnv3 block	16	3	1	✓	RE	128	64	32	-
5	mnv3 block	32	5	2	✓	RE	96	96	48	40
6	mnv3 block	32	3	1	✓	RE	128	80	56	40
7	mnv3 block	32	0	1	✓	RE	0	0	0	0
8	mnv3 block	32	0	1	✓	RE	0	0	0	0
9	mnv3 block	64	5	2	✓	HS	224	224	96	80
10	mnv3 block	64	3	1	✓	HS	224	96	96	-
11	mnv3 block	64	3	1	✓	HS	256	256	96	80
12	mnv3 block	64	0	1	✓	HS	0	0	0	0
13	mnv3 block	64	5	1	✓	HS	640	640	144	320
14	mnv3 block	64	3	1	✓	HS	640	512	144	320
15	mnv3 block	64	0	1	✓	HS	0	0	0	0
16	mnv3 block	128	5	2	✓	HS	768	640	192	320
17	mnv3 block	128	5	1	✓	HS	768	256	192	320
18	mnv3 block	128	7	1	✓	HS	896	768	192	320
19	mnv3 block	128	7	1	✓	HS	1152	1152	192	320
20	conv	128	1	1	✓	HS	1152	-	-	-
21	global avg pool	-	-	-	-	-	-	-	-	-
22	conv	1024	1	1		HS	2048	-	-	-
23	fc	-	1	1	-	-	1000	-	-	-

Table 9: The discovered 314M-MAC DNN architecture of NetAdaptV2 on ImageNet presented in Table 2. Type: type of the layer or block. BN: using batch normalization. Act: activation type (HS: Hard-Swish, RE: ReLU). Exp: number of filters in the expansion layer or number of filters in the conv layer. DW: number of filters in the depthwise layer. PW: number of filters in the pointwise layer. SE: number of filters in the squeeze-and-excitation operation. All MobileNetV3 blocks (mnv3 block) with a stride of 1 have residual connections.

Index	Type	T	Kernel Size	Stride	Filter
1	conv	16	3	2	24
2	dw	∞	3	1	20
3	pw	16	1	1	48
4	dw	∞	3	2	48
5	pw	32	1	1	96
6	dw	∞	3	1	96
7	pw	32	1	1	112
8	dw	∞	3	2	112
9	pw	64	1	1	256
10	dw	∞	3	1	256
11	pw	64	1	1	192
12	dw	∞	3	2	192
13	pw	128	1	1	448
14	dw	∞	3	1	448
15	pw	128	1	1	448
16	dw	∞	3	1	448
17	pw	128	1	1	384
18	dw	∞	3	1	384
19	pw	128	1	1	512
20	dw	∞	3	1	512
21	pw	128	1	1	384
22	dw	∞	3	1	384
23	pw	128	1	1	448
24	dw	∞	3	2	448
25	pw	256	1	1	384
26	dw	∞	3	1	384
27	pw	256	1	1	768
28	dw	∞	5	1	768
29	pw	256	1	1	384
30	upsample	-	-	-	-
31	dw	∞	5	1	320
32	pw	128	1	1	192
33	upsample	-	-	-	-
34	dw	∞	5	1	160
35	pw	64	1	1	112
36	upsample	-	-	-	-
37	dw	∞	3	1	112
38	pw	32	1	1	48
39	upsample	-	-	-	-
40	dw	∞	3	1	28
41	pw	16	1	1	24
42	upsample	-	-	-	-
43	pw	0	1	1	1

Table 10: The discovered 87ms DNN architecture of NetAdaptV2 on NYU Depth V2 presented in Table 7. Type: type of the layer, which can be standard convolution (conv), depthwise convolution (dw), pointwise convolution (pw), or nearest neighbor upsampling (upsample). Filter: number of filters. All layers except for upsampling layers are followed by a batch normalization layer and a ReLU activation layer.

Methods		Top-1 Accuracy (%)	# of Layers	Super-Network Training Speed (min/epoch)	# of Samples
CBC	MCD				
		40.0 (+0)	28 (-0)	3.2 (100%)	1064 (100%)
✓		46.5 (+6.5)	19 (-9)	3.8 (119%)	1092 (103%)
✓	✓	49.3 (+9.3)	17 (-11)	3.8 (119%)	567 (53%)

Table 11: The ablation study of the channel-level bypass connections (CBCs) and the multi-layer coordinate descent (MCD) optimizer on ImageNet and MobileNetV1. The latency of the discovered networks is around 6.5ms.