A Notation

$\mathbb{S}_i$: Search options for $layer_i$.

$|\mathbb{S}_i|$: Num of search options on $layer_i$.

s_i: selected search option on $layer_i$ from the set $\mathbb{S}_i$

o_i: default search option applied on $layer_i$, $o_{1..L}$ is the architecture of the baseline model.

$(s_1, s_2, .., s_L)$: A model architecture that applies s_1 on $layer_1$, s_2 on $layer_2$, ... , s_L on $layer_L$

$(s_{1..i-1}, x_i, o_{i+1..L})$: A model architecture that applies s_1 on $layer_1$, s_2 on $layer_2$, ..., default search option o_{i+1} on $layer_{i+1}$, ... o_L on $layer_L$, and search x_i on $layer_i$.

$\mathcal{M}_i$: A model candidate that is searched on $layer_i$, it's in the form of $(s_{1..i-1}, x_i, o_{i+1..L})$.

$\mathbb{M}_i$: All model candidates searching on $layer_i$.

$\mathbb{M}_{i,h}$: Model candidates searching on $layer_i$, and are mapped to $h \in \mathbb{H}$.

$\varphi : \mathbb{M} \to \mathbb{H}$: transforms a model architecture $\mathcal{M} \in \mathbb{M}$ to a finite integer set $\mathbb{H}$

B NASBench-101 Search Details

NASBench-101 defines a search space on 5 ops, each op has 3 options (conv1x1, conv3x3, maxpool 3x3), and 21 potential edges to connect these ops and input, output ops. It contains 509M candidates with their number of parameters, accuracy on Cifar-10, and other information.

We construct the LayerNAS search space by adding a new edge for each layer. Search options in each layer are used to determine either to include a new op or connect two existing ops. By doing so, all constructed candidates can be legit, because all candidates are connected graphs. And this approach of search space construction can satisfy the assumption of LayerNAS: the best model candidate in $layer_i$ can be constructed from candidates in $layer_{i-1}$ by adding an new edge.

In the experiments, Regularized Evolution (RE) sets population_size=50, tournament_size=10; Proximal Policy Optimization (PPO) sets train_batch_size=16, update_batch_size=8, num_updates_per_feedback=10. Both RE and PPO are using MNAS as objective function: $Accuracy \times (Cost/Target)^{-0.07}$

In Figure 1, we observe that in earlier searching iterations LayerNAS performs slightly worse than other algorithms. This is because LayerNAS initially searches model candidates with fewer ops and edges, which intuitively perform poorly. However, after collecting enough information from early layers, LayerNAS consistently performs better. This is because LayerNAS does not rely on randomness, rather, it adds ops and edges from successful candidates in each layers, leading to continuous improvement.

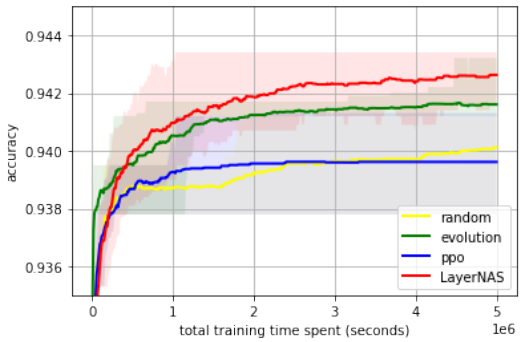

Figure 1: NASBench-101 test accuracy on Cifar-10, average on 100 runs

Table 1: Comparison on NASBench-101

Algorithm	Validation accuracy	Test accuracy
RS	0.9480	0.9401
RE	0.9497	0.9416
PPO	0.9476	0.9396
LayerNAS	**0.9505**	**0.9426**
Optimal	0.9432	0.9445

C NATS-Bench Search Details

In the experiments, Regularized Evolution (RE) sets population_size=50, tournament_size=10; Proximal Policy Optimization (PPO) sets train_batch_size=16, update_batch_size=8, num_updates_per_feedback=10. Both RE and PPO are using MNAS as objective function: $Accuracy \times (Cost/Target)^{-0.07}$

C.1 NATS-Bench topology search

NATS-Bench topology search defines a search space on 6 ops that connect 4 tensors, each op has 5 options (conv1x1, conv3x3, maxpool3x3, no-op, skip).

In our experiments, we construct the LayerNAS search space by adding a new tensor for each layer. Search options in each layer are encoded with all op types that connect this tensor to previous tensors. So it has only 3 layers, each layer has 5, 25, 125 options.

Validation and test accuracy are shown in Figure 2 and Figure 3.

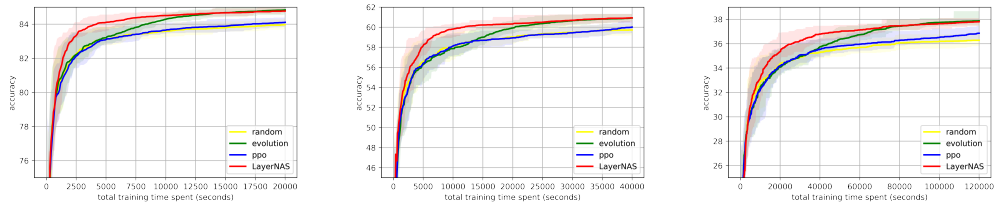

Figure 2: NATS-Bench topology search valid accuracy on (a) Cifar10 (b) Cifar100 (c) Imagenet16-120

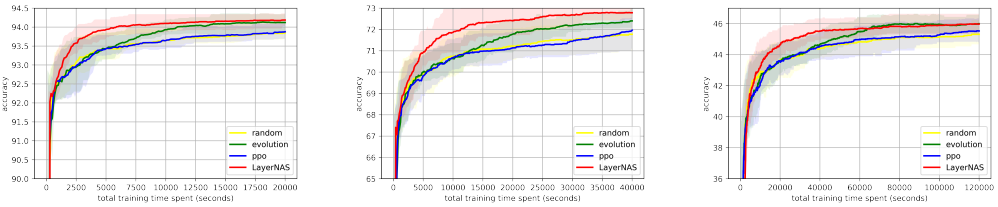

Figure 3: NATS-Bench topology search test accuracy on (a) Cifar10 (b) Cifar100 (c) Imagenet16-120

C.2 NATS-Bench size search

NATS-Bench size search provides a dataset with information on model architectures with 5 layers. Each layer is a convolutional layer with different num of channels selected from {8, 16, 24, 32, 40, 48, 56, 64}. The model with 64 channels for all layers has the most model

parameters, the largest latency and the best accuracy. The objective is to find the optimal model with 50% FLOPs.

LayerNAS constructs the search space by using the largest model as base model, and applies search options that reduce channels per layer. Althoughh LayerNAS steadily improves valid accuracy over time, test accuracy drops. This is due to in-correlation between test accuracy and valid accuracy.

Validation and test accuracy are shown in Figure 4 and Figure 5. We can observe that LayerNAS can outperform other algorithms on both validation and test accuracy. We can also attribute test accuracy drop in LayerNAS to the lack of correlation with validation accuracy.

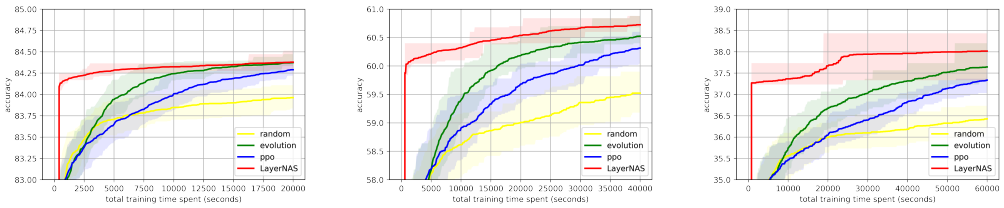

Figure 4: NATS-Bench size search valid accuracy on (a) Cifar10 (b) Cifar100 (c) Imagenet16-120

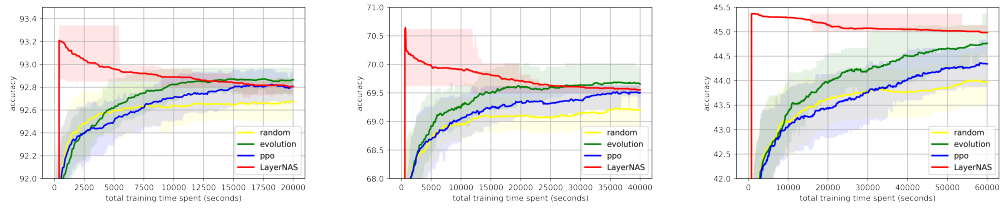

Figure 5: NATS-Bench size search test accuracy on (a) Cifar10 (b) Cifar100 (c) Imagenet16-120

D Dynamic Programming Implementation of LayerNAS for Multi-objective NAS

Algorithm 1 demonstrates how to implement LayerNAS with Dynamic Programming, which has clear explanation why search complexity is $O(H \cdot |\mathbb{S}| \cdot L)$.

The implementation is not used in practice because it spends most of time searching in $layer_{1..L-1}$, we cannot get a model in expected cost range until last layer is searched.

E Discussion on search space assumptions

Assumption 4.1 sets some characteristics of search spaces that can be leveraged to improve the search efficiency. Instead of expecting all search spaces can satisfy this assumption, in experiments, we construct search spaces based on MobileNet to intentionally make them satisfy Assumption 4.1. While we cannot guarantee that all search spaces can be transformed to satisfy Assumption 4.1, most search spaces used in existing models or studies either implicitly use this assumption or can be transformed to satisfy it. We also demonstrate the effectiveness of this assumption from the experiments on MobileNet.

Algorithm 1 Dynamic Programming for Combinatorial Optimization

```
for l = 1 to L − 1 do
  for M_l ∈ 𝕄_l do
    for s ∈ 𝕊_{l+1} do
      M_{l+1} = apply_search_option(M_l, s)
      h = cost(M_{l+1})
      accuracy = train_and_eval(M_{l+1})
      if accuracy > Accuracy(𝕄_{l+1,h}) then
        𝕄_{l+1,h} = M_{l+1}
      end if
    end for
  end for
end for
```

E.1 Search space is complete

Assume we are searching for the optimal model $s_1..s_n$, and we store all possible model candidates on each layer. During the search process on $layer_n$, we generate model architectures by modifying o_n to other options in $\mathbb{S}$. Since we store all model architectures for $layer_{n-1}$, the search process can create all $|\mathbb{S}|^n$ candidates on $layer_n$ by adding each $s_n \in \mathbb{S}$ to the models in $\mathbb{M}_{n-1}$. Therefore, $\mathbb{M}_n$ contains all possibilities in the search space. This process can then be applied backward to the first layer.

E.2 Sequential search order

Assume, after LayerNAS sequential search, we get optimal model defined as $a_1..a_i..a_n$. For sake of contradiction, there exists a model $a_1..b_i..a_n$, with superior performance, by applying a change in previous layers. Since the search space is complete, model $a_1..b_io_{i+1}..o_n$ must exist, and has been processed in $\mathbb{M}_i$. In the sequential search, model $a_1..b_ia_{i+1}..o_n$ can be created by using a_{i+1} on $layer_{i+1}$. Repeating this process for all subsequent layers will eventually lead to $a_1..b_i..a_n$, contradicting our assumption that optimal model from sequential search is $a_1..a_i..a_n$. Therefore, we can search sequentially.

E.3 Limit of the assumption

MobileNet architecture does not satisfy Assumption 4.1 by default. Residual requires $layer_i$ and $layer_j$ have the same num of filters. Suppose $\mathbb{S}_i = \{32, 64, 96\}$, $\mathbb{S}_j = \{64, 96, 128\}$, the residual shortcut cannot be created if $s_i = 32, s_j = 96$. This is the case when preceding layers are coupled with succeeding layers. To overcome this issue, we introduce a virtual layer, with options $\{64, 96\}$. We first search this shared filter to create residual shortcuts, and then search specs for each layer. This transformation ensures that the new search space satisfy Assumption 4.1. In the case of MobileNet search space, we first search for the common filters for the block and then for the expanded filters for each layer. This approach allows us to perform LayerNAS on a search space that satisfies Assumption 4.1.

F Discussion on num of replicas to store

From experiments on MobileNet, we observed that multiple runs on the same model architecture can yield standard deviations of accuracy ranging from 0.08% to 0.15%. Often times, the difference can be as high as 0.3%. To address this, we propose storing multiple candidates for the same cost to increase the likelihood of keeping the better model architecture for every layer search.

Suppose we have two models with the same cost, x and y, where x is inferior and y is superior, and the training accuracy follows a Gaussian distribution $N(\mu, \sigma^2)$. The probability of x obtaining a higher accuracy than y is $P(x - y > 0)$, where $x - y \sim N(\mu_x - \mu_y, {\sigma_x}^2 + {\sigma_y}^2)$. In emprical examples, $\mu_x - \mu_y = -0.002$ and $\sigma_x = 0.001$, then x has the probability of 9.2%

of obtaining a higher accuracy. When we have $L = 20$ layers, the probability of keeping the better model architecture for every layer search is $(1 - p)^2 = 18\%$.

By storing k candidates with the same cost, we can increase the probability of keeping the better model architecture. When $k = 3$, the probability of storing all inferior models is $p^k = 0.08\%$. The probability of keeping the better model architecture for all $L = 20$ layer searches is 98.4%, which is practically good enough.

Theoretically, if we store infinite candidates per layer, we are performing a complete grid search, which guarantees a optimal model architecture.

G Transferability

LayerNAS's explored model architectures exhibit improved performance across various tasks as well.

Table 2: Comparison of models on ImageNet

Model	ImageNet top-1 acc	CoCo mAP	Params	MAdds
MobileNetV2	72.0	22.1	3.5M	300M
LayerNAS w/o SE	77.1	23.85	7.6M	598M
LayerNAS	78.6	24.84	9.7M	527M
MobileNetV3-Small	67.4	16	2.5M	56M
LayerNAS	69.0	17.94	3.7M	61M
MobileNetV3-Large	75.2	22.0	5.4M	219M
LayerNAS	75.6	23.75	5.1M	229M

H MobileNetV2 and MobileNetV3 Search Details

We aim to search models under different MAdds constrants: 60M (similar to MobileNetV3-Small), 220M (similar to MobileNetV3-Large), 300M (similar to MobileNetV2), 600M (similar to MobileNetV2 1.4x).

For each block, we will search the number of output filters of the block first. All layers in the block have the same number of output filters to create residual block correctly. Following the search for the block output filters, we search expanded filter and kernel size of each layers in this block. Strides are fixed for all layers. We use $|\mathbb{S}|$ to denote the number of search options of this layer, which facilitates the computation on the number of unique model architectures, and max number of required search trials in LayerNAS.

H.1 60M MAdds Model

The search spaces has $L = 16$ encoded length. Number of unique model architecture is $\prod |\mathbb{S}| = 5.0e + 20$. We store up to 300 model candidates per layer, so max number of trials is $300 \times \sum |\mathbb{S}| = 1.2e + 5$.

H.2 220M MAdds Model

The search spaces has $L = 21$ encoded length, the number of unique model architecture is $\prod |\mathbb{S}| = 4.8e + 26$ For LayerNAS, we store up to 300 model candidates per layer, so max number of trials is $300 \times \sum |\mathbb{S}| = 1.5e + 5$

H.3 300M MAdds Model

The search spaces has $L = 26$ encoded length, the number of unique model architectures is $\prod |\mathbb{S}| = 5.3e + 30$. We store up to 300 model candidates per layer, so max number of trials is $300 \times \sum |\mathbb{S}| = 1.4e + 5$.

Table 3: 60M MAdds Search Space

Operator	# Output filter	# Expanded Filter	strides	$\lvert\mathbb{S}\rvert$
conv2d{3x3}	16		2	
bneck {3x3}	{24, 20, 18, 16, 14, 12}		2	6
Block filter	{36, 32, 28, 24, 20, 18, 16}			7
bneck {3x3, 5x5}		{144, 136, 128, 120, 112, 104, 96, 88, 80, 72, 68, 64, 60, 56}	2	28
bneck {3x3, 5x5}		{144, 136, 128, 120, 112, 104, 96, 88, 80, 72 68, 64, 60, 56}	1	28
Block filter	{60, 56, 52, 48, 44, 40, 36, 32, 28}			9
bneck {3x3, 5x5, 7x7}		{192, 176, 160, 144, 128, 112, 104, 96, 88, 80, 72, 64}	2	36
bneck {3x3, 5x5, 7x7}		{480, 440, 400, 360, 320, 300, 280, 260, 240, 220, 200, 180, 160}	1	39
bneck {3x3, 5x5, 7x7}		{480, 440, 400, 360, 320, 300, 280, 260, 240, 220, 200, 180, 160}	1	39
Block filter	{96, 88, 80, 72, 64, 60, 56, 52, 48, 44, 40, 36, 32}			13
bneck {3x3, 5x5, 7x7}		{240, 200, 180, 160, 140, 120, 100, 90, 80}	1	27
bneck {3x3, 5x5, 7x7}		{288, 256, 224, 208, 192, 176, 160, 152, 144, 136, 128, 120}	1	36
Block filter	{192, 176, 160, 144, 128, 120, 112, 104, 96, 88, 80, 72, 64}			13
bneck {3x3, 5x5, 7x7}		{576, 544, 512, 480, 448, 416, 384, 352, 320, 288, 256, 224}	2	36
bneck {3x3, 5x5, 7x7}		{1152, 1088, 1024, 960, 896, 832, 768, 704, 640, 576, 516, 448}	1	36
bneck {3x3, 5x5, 7x7}		{1152, 1088, 1024, 960, 896, 832, 768, 704, 640, 576, 516, 448}	1	36
conv2d 1x1	{864, 576},			2
pool, 7x7				
conv2d 1x1	{1536, 1024}			2
conv2d 1x1	{1001}			

Table 4: LayerNAS Model under 60M MAdds

Input	Operator	# Output filter	# Expanded Filter	strides
$224 \times 224 \times 3$	conv2d 3x3	16		2
$112 \times 112 \times 16$	bneck 3x3	16		2
$56 \times 56 \times 16$	bneck 3x3	28	144	2
$28 \times 28 \times 28$	bneck 3x3	28	128	1
$28 \times 28 \times 28$	bneck 5x5	44	96	2
$14 \times 14 \times 44$	bneck 3x3	44	220	1
$14 \times 14 \times 44$	bneck 3x3	44	200	1
$14 \times 14 \times 44$	bneck 7x7	40	160	1
$14 \times 14 \times 40$	bneck 3x3	40	152	1
$14 \times 14 \times 96$	bneck 5x5	96	224	2
$7 \times 7 \times 96$	bneck 3x3	96	448	1
$7 \times 7 \times 96$	bneck 3x3	96	512	1
$7 \times 7 \times 96$	conv2d 1x1	864		1
$7 \times 7 \times 864$	pool, 7x7			1
$7 \times 7 \times 864$	conv2d 1x1	1536		1
$7 \times 7 \times 1536$	conv2d 1x1	1001		1

H.4 600M MAdds Model

The search spaces has $L = 31$ encoded length, the number of unique model architecture is $\prod |\mathbb{S}| = 1.6e + 39$ For LayerNAS, we store up to 300 model candidates per layer, so max number of trials is $300 \times \sum |\mathbb{S}| = 2.0e + 6$

Table 5: 220M MAdds Search Space

Operator	# Output filter	# Expanded Filter	strides	$\lvert\mathbb{S}\rvert$
Conv2d{3x3}	16		2	
bneck {3x3}	{24, 20, 18, 16, 14, 12}		1	6
Block filter	{36, 32, 28, 24, 20, 16}			
bneck {3x3, 5x5}		{96, 88, 80, 72, 68, 64, 60, 56, 48}	2	18
bneck {3x3, 5x5, 7x7}		{124, 116, 108, 100, 92, 84, 72, 68, 64, 56, 48}	1	33
Block filter	{64, 56, 52, 48, 44, 40, 36, 32, 24}			9
bneck {3x3, 5x5, 7x7}		{128, 120, 112, 104, 96, 88, 80, 76, 72, 64, 56}	2	33
bneck {3x3, 5x5, 7x7}		{240, 200, 180, 160, 140, 120, 110, 100, 80}	1	27
bneck {3x3, 5x5, 7x7}		{240, 200, 180, 160, 140, 120, 110, 100, 80}	1	27
Block filter	{160, 140, 130, 120, 110, 100, 80, 70, 60}			9
bneck {3x3, 5x5, 7x7}		{360, 320, 300, 280, 260, 240, 220, 200, 180, 160}	2	30
bneck {3x3, 5x5, 7x7}		{400, 360, 340, 320, 300, 280, 260, 240, 220, 200, 180, 160, 120}	1	36
bneck {3x3, 5x5, 7x7}		{368, 336, 304, 288, 272, 256, 240, 224, 208, 184, 168, 152}	1	36
bneck {3x3, 5x5, 7x7}		{368, 336, 304, 288, 272, 256, 240, 224, 208, 184, 168, 152}	1	36
Block filter	{224, 208, 192, 176, 160, 144, 128, 112, 96, 80}			10
bneck {3x3, 5x5, 7x7}		{960, 880, 800, 720, 640, 560, 520, 480, 440, 400, 360}	1	33
bneck {3x3, 5x5, 7x7}		{1344, 1200, 1056, 960, 888, 816, 768, 720, 624, 576, 480}	1	33
Block filter	{320, 280, 240, 220, 200, 180, 160, 120, 100 }			9
bneck {3x3, 5x5, 7x7}		{1344, 1200, 1056, 960, 888, 816, 768, 720, 624, 576, 480}	2	33
bneck {3x3, 5x5, 7x7}		{1920, 1760, 1600, 1440, 1280, 1120, 960, 880, 800, 720, 640}	1	33
bneck {3x3, 5x5, 7x7}		{1920, 1760, 1600, 1440, 1280, 1120, 960, 880, 800, 720, 640}	1	33
bneck {3x3, 5x5, 7x7}	{480, 440, 400, 360, 320, 300, 280}	{1728, 1664, 1600, 1536, 1440, 1280, 1216}	1	7
conv2d 1x1	{960}			
pool, 7x7				
conv2d 1x1	{1440, 1280}			2
conv2d 1x1	{1001}			

Table 6: LayerNAS Model under 220M MAdds

Input	Operator	# Output filter	# Expanded Filter	strides
$224 \times 224 \times 3$	conv2d 3x3	16		2
$112 \times 112 \times 16$	bneck 3x3	18		1
$112 \times 112 \times 16$	bneck 3x3	24	64	2
$56 \times 56 \times 28$	bneck 3x3	24	48	1
$56 \times 56 \times 28$	bneck 5x5	56	80	2
$28 \times 28 \times 44$	bneck 5x5	56	200	1
$28 \times 28 \times 44$	bneck 5x5	56	100	1
$28 \times 28 \times 44$	bneck 5x5	80	400	2
$14 \times 14 \times 40$	bneck 3x3	80	200	1
$14 \times 14 \times 96$	bneck 3x3	80	272	1
$7 \times 7 \times 96$	bneck 3x3	80	168	1
$14 \times 14 \times 44$	bneck 5x5	112	440	1
$14 \times 14 \times 40$	bneck 5x5	112	576	1
$14 \times 14 \times 96$	bneck 7x7	160	624	2
$7 \times 7 \times 96$	bneck 5x5	160	640	1
$7 \times 7 \times 96$	bneck 3x3	160	640	1
$7 \times 7 \times 96$	conv2d 1x1	960		1
$7 \times 7 \times 864$	pool, 7x7			1
$7 \times 7 \times 864$	conv2d 1x1	1280		1
$7 \times 7 \times 1536$	conv2d 1x1	1001		1

Table 7: 300M MAdds Search Space

Operator	# Output filter	# Expanded Filter	strides	$\lvert\mathbb{S}\rvert$
Conv2d{3x3}	32		2	
bneck {3x3}	{24, 20, 16, 14}		1	4
Block filter	{48, 44, 40, 36, 32, 28, 24}			7
bneck {3x3, 5x5}		{72, 64, 56, 52, 48, 44, 40}	2	14
bneck {3x3, 5x5}		{144, 128, 120, 112, 104, 96, 92, 88, 80, 76}	1	20
bneck {3x3, 5x5}		{144, 128, 120, 112, 104, 96, 92, 88, 80, 76}	1	20
Block filter	{60, 56, 52, 48, 44, 40, 36, 32}			8
bneck {3x3, 5x5}		{144, 128, 120, 112, 104, 96, 92, 88, 80, 76}	2	20
bneck {3x3, 5x5, 7x7}		{180, 160, 140, 130, 120, 110, 100, 80}	1	24
bneck {3x3, 5x5, 7x7}		{180, 160, 140, 130, 120, 110, 100, 80}	1	24
bneck {3x3, 5x5, 7x7}		{180, 160, 140, 130, 120, 110, 100, 80}	1	24
Block filter	{120, 110, 100, 90, 80, 70, 60}			7
bneck {3x3, 5x5, 7x7}		{360, 320, 280, 260, 240, 220, 200, 180}	2	24
bneck {3x3, 5x5, 7x7}		{360, 320, 280, 260, 240, 220, 200, 180}	1	24
bneck {3x3, 5x5, 7x7}		{360, 320, 280, 260, 240, 220, 200, 180}	1	24
Block filter	{144, 128, 120, 104, 96, 88, 80, 72}			8
bneck {3x3, 5x5, 7x7}		{360, 320, 280, 260, 240, 220, 200, 180}	1	24
bneck {3x3, 5x5, 7x7}		{432, 400, 368, 336, 304, 288, 272, 256, 240}	1	27
bneck {3x3, 5x5, 7x7}		{432, 400, 368, 336, 304, 288, 272, 256, 240}	1	27
bneck {3x3, 5x5, 7x7}		{432, 400, 368, 336, 304, 288, 272, 256, 240}	1	27
Block filter	{288, 256, 224, 192, 160, 144}			6
bneck {3x3, 5x5, 7x7}		{864, 800, 736, 672, 608, 576, 512, 448}	2	24
bneck {3x3, 5x5, 7x7}		{864, 800, 736, 672, 608, 576, 512, 448}	1	24
bneck {3x3, 5x5, 7x7}		{864, 800, 736, 672, 608, 576, 512, 448}	1	24
bneck {3x3, 5x5, 7x7}		{864, 800, 736, 672, 608, 576, 512, 448}	1	24
bneck {3x3, 5x5, 7x7}	{480, 440, 400, 360, 320, 300, 280}	{1728, 1664, 1600, 1536, 1440, 1280, 1216}	1	7
pool, 7x7 conv2d 1x1 conv2d 1x1	{1920, 1600, 1280} {1001}			3

Table 8: LayerNAS Model under 300M MAdds

Input	Operator	# Output filter	# Expanded Filter	strides
$224 \times 224 \times 3$	conv2d 3x3	32		2
$112 \times 112 \times 32$	bneck 3x3	24		1
$112 \times 112 \times 24$	bneck 3x3	28	40	2
$56 \times 56 \times 28$	bneck 3x3	28	144	1
$56 \times 56 \times 28$	bneck 3x3	28	88	1
$56 \times 56 \times 28$	bneck 3x3	40	104	2
$28 \times 28 \times 40$	bneck 5x5	40	110	1
$28 \times 28 \times 40$	bneck 3x3	40	180	1
$28 \times 28 \times 40$	bneck 5x5	40	130	1
$28 \times 28 \times 40$	bneck 7x7	90	260	2
$14 \times 14 \times 90$	bneck 3x3	90	220	1
$14 \times 14 \times 90$	bneck 3x3	90	200	1
$14 \times 14 \times 90$	bneck 7x7	120	320	1
$14 \times 14 \times 120$	bneck 5x5	120	288	1
$14 \times 14 \times 120$	bneck 7x7	120	256	1
$14 \times 14 \times 120$	bneck 3x3	120	368	1
$14 \times 14 \times 120$	bneck 7x7	160	608	2
$7 \times 7 \times 160$	bneck 7x7	160	576	1
$7 \times 7 \times 160$	bneck 5x5	160	608	1
$7 \times 7 \times 160$	bneck 3x3	160	448	1
$7 \times 7 \times 160$	bneck 3x3	280	1216	1
$7 \times 7 \times 280$	pool, 7x7			1
$7 \times 7 \times 280$	conv2d 1x1	1920		1
$7 \times 7 \times 1920$	conv2d 1x1	1001		1

Table 9: 600M MAdds Search Space

Operator	# Output filter	# Expanded Filter	strides	$\lvert\mathbb{S}\rvert$
Conv2d{3x3}	32		2	
bneck {3x3}	{36, 32, 28, 24, 20, 16}		1	6
Block filter	{56, 52, 48, 44, 40, 36, 32, 28}			8
bneck {3x3, 5x5}		{88, 80, 72, 64, 56, 52, 48}	2	14
bneck {3x3, 5x5}		{88, 80, 72, 64, 56, 52, 48}	1	14
bneck {3x3, 5x5}		{88, 80, 72, 64, 56, 52, 48}	1	14
bneck {3x3, 5x5}		{88, 80, 72, 64, 56, 52, 48}	1	14
Block filter	{72, 64, 60, 56, 52, 48, 44, 40}			8
bneck {3x3, 5x5}		{180, 160, 144, 128, 120, 112, 104, 96, 92, 88, 80}	2	22
bneck {3x3, 5x5, 7x7}		{240, 220, 200, 180, 160, 140, 130, 120, 100}	1	27
bneck {3x3, 5x5, 7x7}		{240, 220, 200, 180, 160, 140, 130, 120, 100}	1	27
bneck {3x3, 5x5, 7x7}		{240, 220, 200, 180, 160, 140, 130, 120, 100}	1	27
bneck {3x3, 5x5, 7x7}		{240, 220, 200, 180, 160, 140, 130, 120, 100}	1	27
Block filter	{200, 180, 160, 140, 120, 100, 90, 80}			8
bneck {3x3, 5x5, 7x7}		{440, 400, 360, 320, 280, 260, 240, 200}	2	24
bneck {3x3, 5x5, 7x7}		{560, 520, 480, 440, 400, 360, 320, 280, 240}	1	27
bneck {3x3, 5x5, 7x7}		{560, 520, 480, 440, 400, 360, 320, 280, 240}	1	27
bneck {3x3, 5x5, 7x7}		{560, 520, 480, 440, 400, 360, 320, 280, 240}	1	27
Block filter	{180, 160, 144, 128, 120, 104, 96, 88, 80}			9
bneck {3x3, 5x5, 7x7}		{560, 520, 480, 440, 400, 360, 320, 280, 240}	1	27
bneck {3x3, 5x5, 7x7}		{560, 528, 496, 464, 432, 400, 368, 336, 304, 288, 272, 256}	1	36
bneck {3x3, 5x5, 7x7}		{560, 528, 496, 464, 432, 400, 368, 336, 304, 288, 272, 256}	1	36
bneck {3x3, 5x5, 7x7}		{560, 528, 496, 464, 432, 400, 368, 336, 304, 288, 272, 256}	1	36
bneck {3x3, 5x5, 7x7}		{560, 528, 496, 464, 432, 400, 368, 336, 304, 288, 272, 256}	1	36
Block filter	{320, 288, 256, 224, 192, 160}			6
bneck {3x3, 5x5, 7x7}		{992, 928, 864, 800, 736, 672, 608, 576, 512}	2	27
bneck {3x3, 5x5, 7x7}		{992, 928, 864, 800, 736, 672, 608, 576, 512}	1	27
bneck {3x3, 5x5, 7x7}		{992, 928, 864, 800, 736, 672, 608, 576, 512}	1	27
bneck {3x3, 5x5, 7x7}		{992, 928, 864, 800, 736, 672, 608, 576, 512}	1	27
bneck {3x3, 5x5, 7x7}		{992, 928, 864, 800, 736, 672, 608, 576, 512}	1	27
bneck {3x3, 5x5, 7x7}	{600, 560, 520, 480, 440, 400, 360, 320}	{1920, 1856, 1792, 1728, 1664, 1600, 1536, 1440}	1	24
pool, 7x7 conv2d 1x1 conv2d 1x1	{2560, 2240, 1920} {1001}			3

Table 10: LayerNAS Model under 600M MAdds

Input	Operator	# Output filter	# Expanded Filter	strides
$224 \times 224 \times 3$	conv2d 3x3	32		2
$112 \times 112 \times 32$	bneck 3x3	36		1
$112 \times 112 \times 36$	bneck 5x5	36	80	2
$56 \times 56 \times 36$	bneck 5x5	36	72	1
$56 \times 56 \times 36$	bneck 3x3	36	80	1
$56 \times 56 \times 36$	bneck 5x5	36	72	1
$56 \times 56 \times 36$	bneck 3x3	48	144	2
$28 \times 28 \times 48$	bneck 3x3	48	140	1
$28 \times 28 \times 48$	bneck 3x3	48	160	1
$28 \times 28 \times 48$	bneck 3x3	48	130	1
$28 \times 28 \times 48$	bneck 5x5	48	140	1
$28 \times 28 \times 48$	bneck 7x7	140	360	2
$14 \times 14 \times 140$	bneck 5x5	140	360	1
$14 \times 14 \times 140$	bneck 3x3	140	560	1
$14 \times 14 \times 140$	bneck 5x5	140	440	1
$14 \times 14 \times 140$	bneck 7x7	144	360	1
$14 \times 14 \times 144$	bneck 5x5	144	560	1
$14 \times 14 \times 144$	bneck 3x3	144	288	1
$14 \times 14 \times 144$	bneck 5x5	144	400	1
$14 \times 14 \times 144$	bneck 5x5	144	256	1
$14 \times 14 \times 144$	bneck 3x3	192	864	2
$7 \times 7 \times 192$	bneck 5x5	192	928	1
$7 \times 7 \times 192$	bneck 7x7	192	736	1
$7 \times 7 \times 192$	bneck 7x7	192	800	1
$7 \times 7 \times 192$	bneck 3x3	192	928	1
$7 \times 7 \times 192$	bneck 3x3	320	1440	1
$7 \times 7 \times 320$	pool, 7x7			1
$7 \times 7 \times 320$	conv2d 1x1	2560		1
$7 \times 7 \times 2560$	conv2d 1x1	1001		1