Evolving Search Space for Neural Architecture Search

**Supplementary Material**

<table>
<thead>
<tr>
<th>Shape</th>
<th>Block</th>
<th>c</th>
<th>n</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>224^2 × 3</td>
<td>3x3 conv</td>
<td>16</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>112^2 × 16</td>
<td>MBL</td>
<td>16</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>112^2 × 16</td>
<td>MBL</td>
<td>24</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>56^2 × 24</td>
<td>MBL</td>
<td>40</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>28^2 × 40</td>
<td>MBL</td>
<td>80</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>14^2 × 80</td>
<td>MBL</td>
<td>96</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>14^2 × 96</td>
<td>MBL</td>
<td>192</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>7^2 × 192</td>
<td>MBL</td>
<td>320</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7^2 × 320</td>
<td>1x1 conv</td>
<td>1024</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7^2 × 1024</td>
<td>7x7 avgpool</td>
<td>-</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1024</td>
<td>fc</td>
<td>1024</td>
<td>1</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1. Macro-architecture for FLOPs constraint setting. “MBL” denotes the learnable Multi-Branch layer. c, n, s refer to the number of backbone filters, number of layers and the stride, respectively.

A. Search Space Details

A.1. FLOPs Constraint Search Space

The 27 OPs space for FLOPs constraint, as shown in Figure 1, is derived from multiple groups of operation designs. The first group of operations is depthwise (DW) convolution with kernel size \{3, 5, 7, 9, 11\} and expand ratio \{1, 3, 6\}. The second group is 3 × 3 dilated convolution with dilation \{2, 3\} and expand ratio \{1, 3, 6\}, this kind of operation, according to the study in MixNet [13], is not efficient under FLOPs constrained scenarios. However, we still include them in our search space to test the robustness of the proposed method and see if it can find competitive architectures in a noised large search space. We also include the 1 × k − k × 1 convolutions with k ∈ \{5, 7\} and expand ratio \{1, 2, 4\}, this operation is derived from the Inception-ResNet [11] and is a rarely included operation in NAS literature as well. Our major experiments are conducted in this setting.

The second space as shown in Figure 1 consists of DW convolutions with grouped 1 × 1 projections, a special variant of standard DW convolution that is included in FBNet [15] and MixNet [13]. The options of kernel size and expand ratio for this variant are \{3, 5, 7, 9, 11\} and \{1, 3, 6\} respectively, which is identical with standard DW convolutions in 27 OPs space. For both search space, we use identical macro-architecture as shown in Table 1.

A.2. Latency Constraint Search Space

Our search space for Latency constraint as shown in Figure 2 and Table 2 is identical with the extended search space used by Li et al. [4].

A.3. Identity Mapping Path

Inspired by the Inception-Resnet [11], our search space has a residual structure, which means that all normal layers in the network have an identity mapping path (identity op-
Table 2. Macro-architecture for Latency constraint setting. “MBL” denotes the learnable Multi-Branch layer. c, n, s refer to the number of backbone filters, number of layers and the stride, respectively.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Block</th>
<th>c</th>
<th>n</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>$224^2 \times 3$</td>
<td>3x3 conv</td>
<td>16</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>$112^2 \times 16$</td>
<td>MBL</td>
<td>16</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$112^2 \times 16$</td>
<td>MBL</td>
<td>32</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>$56^2 \times 32$</td>
<td>MBL</td>
<td>64</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>$28^2 \times 64$</td>
<td>MBL</td>
<td>128</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>$14^2 \times 128$</td>
<td>MBL</td>
<td>256</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

$7^2 \times 256$ | 1x1 conv | 1024 | 1 | 1 |

$7^2 \times 1024$ | 7x7 avgpool | - | 1 | 1 |

1024 | fc | 1024 | 1 | - |

Table 3. ImageNet NAS search space size compared. * when we use 27 OPs space and $K = 5$.

$Comb_{arch}$ is computed as follows:

We denote the number of $k$-combinations given $n$ elements as $C^n_k = \binom{n}{k}$. The number of possible combinations is $Comb_{norm} = \sum_{k=0}^{5} C^5_k$ for the normal layer and $Comb_{redu} = \sum_{k=1}^{5} C^5_k$ for the reduction layer. There are in total 16 normal layers and 6 reduction layers in FLOPs constrained macro architecture. Each layer has its own selected candidate operations. Thus the total number of possible architectures is $Comb_{arch} = (Comb_{norm})^{16} \times (Comb_{redu})^6 \approx 1.4 \times 10^{110}$.
space to be inherited, we evaluate the quality of aggregated search space achieved by different pruning thresholds in Figure 3. As the threshold -1 is too close to 0 (the initialized value of fitness indicators Θ), its result is significantly worse when compared to lower thresholds. However, as the threshold is set lower than -2, the result seems saturated, and a lower threshold could even harm the quality of optimized search space.

D. Detection Result for NSENet

We have also evaluated our NSENet on object detection task. We take the pretrained NSENet as a drop-in replacement for the backbone feature extractor in EfficientDet-D0 [14]. Table 4 shows the performance of our NSENet, comparing with MobileNetV2 and the original backbone network EfficientNet-B0. We trained the network with identical configs as used by EfficientDet-D0. As shown in Table 4, our model significantly improves mAP score over MobileNetV2 and EfficientNet-B0 with fewer FLOPs.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>FLOPs</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>EfficientNet-B0</td>
<td>2.50B</td>
<td>33.8</td>
</tr>
<tr>
<td>MobileNetV2 1.0</td>
<td>2.24B</td>
<td>32.7</td>
</tr>
<tr>
<td>NSENet</td>
<td>2.18B</td>
<td>34.5</td>
</tr>
</tbody>
</table>

Table 4. NSENet object detection performance on COCO [5] dataset. All experiments adopt identical configs as used by EfficientDet-D0 except backbone network.

E. Omitted Figures

Below we show the omitted figures. Figure 4 shows architecture details for final results. Figure 5 shows intermediate results of aggregated search space subset on 27 OPs space. Figure 6 illustrates the edging effect on Pareto frontier.
Figure 4. The detailed operations (a)(d) and structure (b)(c)(e) of our final results for FLOPs constraint and latency constraint, notice that (b) is the final result derived from 27 OPs space while (c) inherits the final search space subset derived from the 27 OPs space, then search on the second space as shown in Figure 1. The two numbers within the operation blocks shown in (c) represents the group number (G_{in}, G_{out}) of 1x1 projections. The width of the blocks correspond to the T in (a)(d) for candidate operation, which denotes the expand ratio of the corresponding operation, with details in Figure 1 and Figure 2. A straight line is put after every reduction layer in (b)(c) and (e). A "Scale Factor" [3] is used to adjust the amount of resource (e.g. FLOPs) consumed by the architecture by changing the number of channels uniformly. We can see that architectures searched under FLOPs constraint tend to go deeper while both constraints prefer efficient operations such as DW convolutions over less commonly used operations such as SSC convolutions.
Figure 5. Intermediate results of the search space subset derived from Pareto front architecture aggregation. The results are based on the 27 OPs space and are from the same experiment where we get the NSENet-27 architecture. We can see that less commonly used operations such as SSC convolutions and dilated DW convolutions are seldom in the search space subset. On the other hand, most of the operations being included in the search space subset would last for multiple rounds or even till the final round, demonstrating the effectiveness of the proposed pipeline in terms of knowledge extraction and preservation.
Figure 6. Edging effect in constrained Pareto frontier retrieval. When trying to get Pareto-optimal architectures only with the samples within the constraint interval, some of the samples (orange points in this figure) located close to the limit boundary (300M FLOPs) could be mistakenly considered as Pareto-optimal architectures. By considering auxiliary samples outside the limit interval, we can alleviate this issue. The data used in this figure is derived from the final round of search over 27 OPs space.
References