Supplementary Material for "Rethinking Bottleneck Structure for Efficient Mobile Network Design"

Anonymous ECCV submission

Paper ID 1101

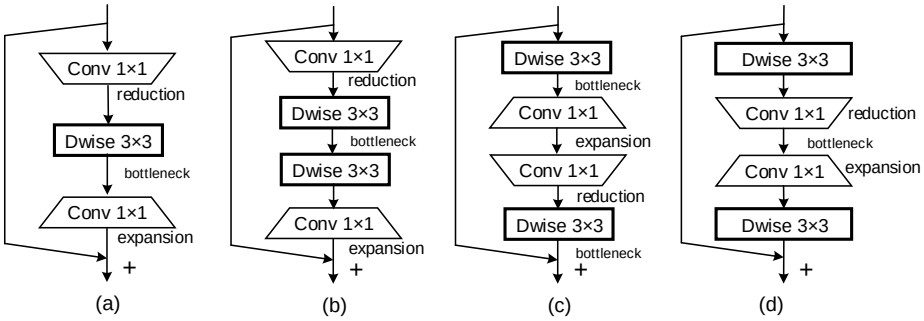


Fig. S1. Conceptual diagram for variants of our proposed sandglass block. (a) Bottleneck structure with standard convolutions replaced by depthwise convolutions. (b) Structure in (a) with one more depthwise convolution in the bottleneck. (c) Switching the positions of the two pointwise convolutions in our sandglass block. (d) Our proposed sandglass block.

1 Variants of the Proposed Sandglass Block

In this section, we introduce and compare the variants of our proposed sandglass block, which are shown in Figures S1 (a-c). The corresponding results are listed in Table S1.

1. The first variant (Figure S1(a)) is built from direct modification of the classic bottleneck structure [2] by replacing the standard 3×3 convolution with a 3×3 depthwise convolution. From the result, we can observe performance drop of about 5% compared to our sandglass block. We argue this is mostly because the depthwise convolution is conducted in the bottleneck with a low-dimensional feature space and hence cannot capture enough spatial information, leading to much worse performance compared to our proposed sandglass block (Figure S1(d)).
2. The second variant (Figure S1(b)) is derived from the first variant, but differently we add another 3×3 depthwise convolution in the bottleneck. As can be seen, the top-1 accuracy improves by more than 1% compared to the structure shown in Figure S1(a). This indicates encoding more spatial information indeed helps. This phenomenon can also be observed by comparing Figure S1(b) and Figure S1(d) (70.11 *v.s.* 74.02).

Table S1. Performance of different variants of our proposed sandglass block shown in Figure S1.

Block structure	#Dwise convs	Param. (M)	M-Adds (M)	Top-1 Acc. (%)
MobileNetV2	1	3.5	300	72.3
Figure S1(a)	1	3.4	240	68.90
Figure S1(b)	2	3.4	250	70.11
Figure S1(c)	2	3.5	300	69.26
Figure S1(d)	2	3.5	300	74.02

3. The third variant (Figure S1(c)) is based on the original inverted residual block [4]. We move the depthwise convolution from the high-dimensional feature space to the bottleneck positions with less feature channels. Compared with Figure S1(b), this variant in Figure S1(c) has a comparable number of learnable parameters and more computational cost but worse performance (69.26 *v.s.* 70.11). This also means building shortcuts between high-dimensional representations is more beneficial to the network performance.

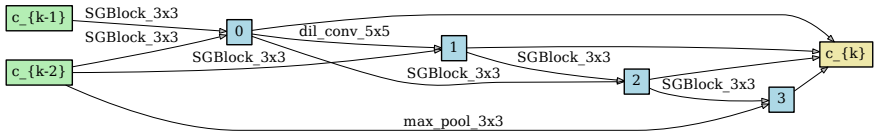
As shown in Table S1, our proposed sandglass block achieves much better results than all the three variants. The performance improvements can be explained by the two rules that we have presented in the main paper: (1) adding shortcut connections between high-dimensional representations, and (2) performing the depthwise convolution in high-dimensional feature space. Our experiments also indicate that bottleneck structure is suitable for mobile networks and it can work better than the inverted residual block.

Table S2. Results produced by different network architectures searched by DARTS [3]. For Lines 2 and 3, we separately add the inverted residual (IR) block and our sandglass block into the original search space of DARTS. We report results on CIFAR-10 dataset as in [3].

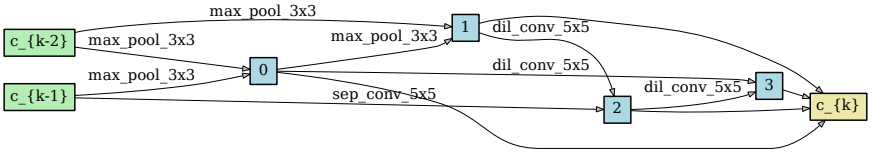
No.	Search Space	Test Error (%)	Param. (M)	Search Method	#Operators
1	DARTS original	3.11	3.25	gradient based	7
2	DARTS + IR Block	3.26	3.29	gradient based	7
3	DARTS + sandglass block	2.98	2.45	gradient based	7

2 Searched Architectures

Following [1,3,5], we also search for a computation cell and use it as the basic building block for the final architecture. The searching space and algorithm are described in details as below.



(a)



(b)

Fig. S2. Cell structures searched on CIFAR-10 with DARTS [3]. (a) Searched normal cell structure. (b) Searched reduction cell structure. 'SGBlock' denotes our proposed sandglass block. We use the same search space as used in [3] with only one more operator included, i.e. our proposed sandglass block.

Search space In our experiments, we use the search space from [3] as our baseline (denoted as *original*), which includes the following operators:

- Convolutional operations (ConvOp): regular convolution, dilated convolution, depthwise convolution;
- Convolution kernel size[1]: 3×3, 5×5, 7×1 followed by 1×7;
- Non-parametric operations: average pooling, max pooling, skip connection, None.

The results are reported in Line 1 of Table S2. To compare with the inverted residual block, we conduct architecture search within the following two new search spaces.

- **Original + IR block**: the original search space plus the inverted residual block as the depthwise separable convolution operation candidate.
- **Original + sandglass block**: the original search space plus the sandglass block.

The corresponding results are reported in Lines 2-3 of Table S2, respectively. The zero (None) operation is also included to indicate the miss of the connections as used in [3].

Searching algorithm As mentioned in the main paper, we adopt the DARTS searching algorithm [3] to search for the cell structure and set the number of nodes to 7 in each directed acyclic graph (DAG) of the cell. During the searching process, we strictly follow the training policy and use the same hyper-parameters as in [3] for a fair comparison.

Architecture and results The searched cell structures (including both the normal cell and the reduction cell) can be found in Figure S2. As can be seen in Table S2, adding the

[1] For the inverted residual block and our sandglass block, we only use a kernel size of 3×3 for depthwise convolutions.

proposed sandglass block into the original search space can largely reduce the number of learnable parameters in the searched architecture with improved classification performance on the CIFAR-10 dataset. This again shows when combined with the searching algorithm, our proposed sandglass block can be used to replace the original block to improve the performance.

Conclusion and discussion From the above results, we can observe that introducing appropriate super operators (*e.g.,* our sandglass block) into the search space can bring better performance compared to using the original basic operators. We hope this experiment could benefit the development of architecture searching algorithms in the future.

References

1. Cai, H., Zhu, L., Han, S.: Proxylessnas: Direct neural architecture search on target task and hardware. arXiv preprint arXiv:1812.00332 (2018)
2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
3. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055 (2018)
4. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR. pp. 4510–4520 (2018)
5. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: CVPR. pp. 8697–8710 (2018)