Table 3: ILSVRC2012 transfer results.

Method	**# params** (mil.)	**# multi-add** (mil.)	**top-1 Test Error** (%)
Inception-v1 (Szegedy et al., 2015)	6.6	1448	30.2
MobileNetV2 (Sandler et al., 2018)	6.9	585	28.0
NASNet-A (Zoph et al., 2017)	5.3	564	26.0
NASNet-B (Zoph et al., 2017)	5.3	488	27.2
AmoebaNet-A (Real et al., 2018)	5.1	555	25.5
PNAS (Liu et al., 2017a)	5.1	588	25.8
DARTS (Liu et al., 2018)	4.9	595	26.9
RandGrow macro	6.88	830	27.2

A Transfer of Macro-search Model to ILSVRC2012

For ILSVRC2012 (Russakovsky et al., 2015), we take the standard data augmentation approach of (He et al., 2016), and use 224x224 input images. Since our macro-search is performed on CIFAR10, which consists of 32x32 images, the found models are not directly applicable to ILSVRC2012. Instead we add three reductions at the start of the model so the 224x224 images of ILSVRC2012 are reduced to 28x28. First, a 3x3 conv with stride of two projects the RGB-color feature into 12 channels. Then we apply two simple reduction cells, each of which is the same as the reduction cells in our initial model. At each reduction cell, the channel size is doubled throughout the network. So after these initial reductions, the channel size is 48 at the start of the macro-search. The top-1 error rate, the number of model parameters and the test-time computational cost in terms of mult-adds are shown in Table 3. We observe that RandGrow macro-search can learn models that have comparable results against to state-of-the-art results on ILSVRC2012 using similar computation. This is one of the first results to showcase that macro-search models can be transferred onto larger data-set, suggesting that the previous believed gap between macro and cell searches are not as wide as it seemed.

B Found Model in RandGrow macro-search

We describe the found model as a list json description of the layers in their toplogical order. The layers are uniquely identified by the field "id". `id=0` and `id=1` are the same, and are the result of the stem convolution on the input image. The field "inputs" is a list of id of the input layers. The field "ops" is a list of operation type for the input layers, followed by a merge operation type to combine the results. The mapping between operation type and operation names is after the layer descriptions. The down-sampling happens at layer `id=8` and `id=15`. When a layer has an input that is of a different channel size or resolution, the input is first projected with 1x1 conv, and down-sampled with factorized reduction if needed.

```
{'ops': [], 'inputs': [], 'id': 0}
{'ops': [], 'inputs': [], 'id': 1}
{'ops': [8, 23, 9, 22], 'inputs': [0, 0, 1], 'id': 40}
{'ops': [24, 8, 23, 22], 'inputs': [0, 1, 1], 'id': 58}
{'ops': [23, 1, 14, 14, 11], 'inputs': [1, 1, 40, 58], 'id': 2}
{'ops': [24, 8, 24, 22], 'inputs': [0, 0, 2], 'id': 22}
{'ops': [9, 9, 24, 22], 'inputs': [0, 0, 2], 'id': 41}
{'ops': [8, 9, 8, 22], 'inputs': [0, 0, 2], 'id': 59}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [2, 2, 22, 41, 59], 'id': 3}
{'ops': [24, 8, 23, 22], 'inputs': [0, 1, 3], 'id': 23}
{'ops': [23, 9, 23, 22], 'inputs': [0, 0, 3], 'id': 42}
{'ops': [24, 23, 24, 22], 'inputs': [0, 1, 3], 'id': 60}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [3, 3, 23, 42, 60], 'id': 4}
{'ops': [24, 23, 24, 22], 'inputs': [0, 0, 4], 'id': 24}
{'ops': [23, 8, 23, 22], 'inputs': [0, 0, 4], 'id': 61}
{'ops': [23, 1, 14, 14, 11], 'inputs': [4, 4, 24, 61], 'id': 5}
{'ops': [8, 24, 23, 22], 'inputs': [1, 3, 5], 'id': 25}
{'ops': [24, 8, 8, 22], 'inputs': [0, 22, 5], 'id': 43}
```

{'ops': [8, 8, 24, 22], 'inputs': [0, 1, 5], 'id': 62}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [5, 5, 25, 43, 62], 'id': 6}
{'ops': [23, 8, 8, 22], 'inputs': [0, 0, 6], 'id': 26}
{'ops': [9, 24, 9, 22], 'inputs': [1, 5, 6], 'id': 44}
{'ops': [9, 24, 8, 22], 'inputs': [1, 1, 6], 'id': 63}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [6, 6, 26, 44, 63], 'id': 7}
{'ops': [8, 8, 9, 22], 'inputs': [0, 3, 7], 'id': 27}
{'ops': [8, 24, 24, 22], 'inputs': [0, 1, 7], 'id': 64}
{'ops': [23, 1, 14, 14, 11], 'inputs': [7, 7, 27, 64], 'id': 8}
{'ops': [24, 9, 24, 22], 'inputs': [1, 1, 8], 'id': 28}
{'ops': [8, 24, 8, 22], 'inputs': [23, 27, 8], 'id': 45}
{'ops': [23, 24, 23, 22], 'inputs': [0, 1, 8], 'id': 65}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [8, 8, 28, 45, 65], 'id': 9}
{'ops': [23, 23, 8, 22], 'inputs': [1, 1, 9], 'id': 29}
{'ops': [23, 9, 24, 22], 'inputs': [0, 0, 9], 'id': 46}
{'ops': [9, 24, 8, 22], 'inputs': [0, 0, 9], 'id': 66}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [9, 9, 29, 46, 66], 'id': 10}
{'ops': [9, 8, 23, 22], 'inputs': [0, 0, 10], 'id': 30}
{'ops': [23, 24, 9, 22], 'inputs': [0, 28, 10], 'id': 47}
{'ops': [23, 1, 14, 14, 11], 'inputs': [10, 10, 30, 47], 'id': 11}
{'ops': [9, 9, 8, 22], 'inputs': [0, 8, 11], 'id': 31}
{'ops': [23, 24, 24, 22], 'inputs': [0, 2, 11], 'id': 48}
{'ops': [23, 1, 14, 14, 11], 'inputs': [11, 11, 31, 48], 'id': 12}
{'ops': [9, 9, 23, 22], 'inputs': [1, 5, 12], 'id': 32}
{'ops': [8, 24, 24, 22], 'inputs': [0, 23, 12], 'id': 49}
{'ops': [9, 9, 23, 22], 'inputs': [0, 1, 12], 'id': 67}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [12, 12, 32, 49, 67], 'id': 13}
{'ops': [24, 23, 9, 22], 'inputs': [0, 6, 13], 'id': 33}
{'ops': [8, 23, 9, 22], 'inputs': [0, 10, 13], 'id': 50}
{'ops': [9, 8, 9, 22], 'inputs': [0, 23, 13], 'id': 68}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [13, 13, 33, 50, 68], 'id': 14}
{'ops': [23, 9, 9, 22], 'inputs': [0, 12, 14], 'id': 51}
{'ops': [8, 8, 24, 22], 'inputs': [0, 0, 14], 'id': 69}
{'ops': [23, 1, 14, 14, 11], 'inputs': [14, 14, 51, 69], 'id': 15}
{'ops': [24, 9, 8, 22], 'inputs': [1, 5, 15], 'id': 34}
{'ops': [9, 8, 23, 22], 'inputs': [0, 33, 15], 'id': 52}
{'ops': [24, 9, 23, 22], 'inputs': [1, 1, 15], 'id': 70}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [15, 15, 34, 52, 70], 'id': 16}
{'ops': [8, 24, 24, 22], 'inputs': [0, 8, 16], 'id': 35}
{'ops': [8, 24, 23, 22], 'inputs': [30, 16, 16], 'id': 53}
{'ops': [24, 9, 9, 22], 'inputs': [1, 6, 16], 'id': 71}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [16, 16, 35, 53, 71], 'id': 17}
{'ops': [9, 8, 23, 22], 'inputs': [2, 4, 17], 'id': 36}
{'ops': [23, 24, 24, 22], 'inputs': [0, 0, 17], 'id': 54}
{'ops': [8, 8, 9, 22], 'inputs': [0, 51, 17], 'id': 72}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [17, 17, 36, 54, 72], 'id': 18}
{'ops': [9, 9, 9, 22], 'inputs': [0, 5, 18], 'id': 37}
{'ops': [23, 8, 9, 22], 'inputs': [0, 13, 18], 'id': 55}
{'ops': [24, 9, 9, 22], 'inputs': [0, 14, 18], 'id': 73}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [18, 18, 37, 55, 73], 'id': 19}
{'ops': [23, 23, 23, 22], 'inputs': [13, 15, 19], 'id': 38}
{'ops': [9, 8, 24, 22], 'inputs': [0, 3, 19], 'id': 56}
{'ops': [23, 8, 24, 22], 'inputs': [1, 53, 19], 'id': 74}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [19, 19, 38, 56, 74], 'id': 20}
{'ops': [8, 23, 9, 22], 'inputs': [0, 8, 20], 'id': 39}
{'ops': [8, 24, 23, 22], 'inputs': [0, 37, 20], 'id': 57}
{'ops': [9, 23, 24, 22], 'inputs': [0, 2, 20], 'id': 75}
{'ops': [23, 1, 14, 14, 14, 11], 'inputs': [20, 20, 39, 57, 75], 'id': 21}

```
IDENTITY = 1
SEPARABLE_CONV_3_2 = 23
SEPARABLE_CONV_5_2 = 24
MAXPOOL_3x3 = 8
AVGPOOL_3x3 = 9
MERGE_WITH_SUM = 11
MERGE_WITH_CAT_PROJ = 22
MULTIPLY_SCALAR = 14
```