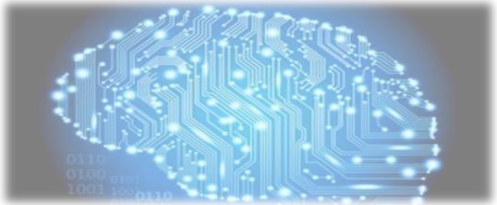

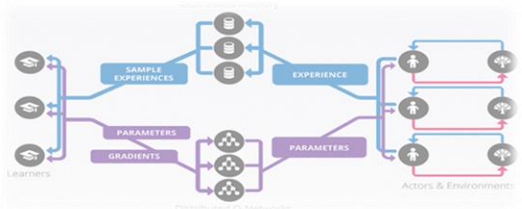

One-Shot Neural Network Architecture Search via Self-Evaluated Template Network

AutoML 연구실
석사과정 문아성

21.04.20

One-Shot Neural Network Architecture Search via Self-Evaluated Template Network

- Typical one-shot NAS paradigm

(1) Randomly sample hundreds of architecture candidates from the parameter-shared network.

(2) evaluate sampled candidates

(3) find the candidate with the highest validation accuracy.

- Self-Evaluated Template Network

  - Template Network

  - Estimator

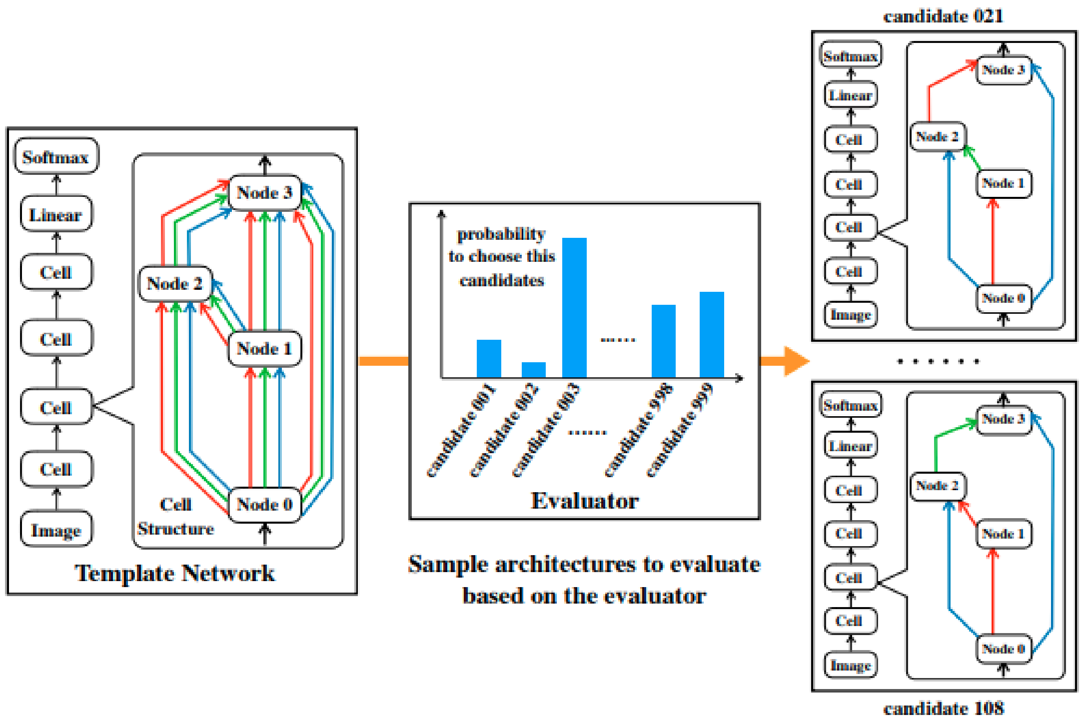

Figure 1. An overview of the self-evaluated template network (SETN). SETN consists of a template network and an evaluator. Architectures for evaluation are generated by sampling candidates in the template network using the evaluator. The template network shares the parameters for different candidates as indicated by connections of arrow lines in different colors. The evaluator learns the distribution of the architectures being likely to have a lower validation loss.

One-Shot Neural Network Architecture Search via Self-Evaluated Template Network

Template Network

- The inputs I_1, I_2
- The candidate function set $\mathcal{O}$
  - 3x3 max pooling
  - 3x3 avg pooling
  - skip connection
  - 3x3 separable conv
  - 5x5 separable conv
  - 1x3 & 3x1 conv
- The Output H_i^c

$$H_i^c = f_1(I_1) + f_2(I_2) \qquad f_1, f_2 \in \mathcal{O} \tag{1}$$

$$s.t.\ \{I_1, I_2\} = UniformSample(I_i^c, 2), \tag{2}$$

$$\{f_1, f_2\} = OrderedUniformSample(\mathcal{O}, 2) \tag{3}$$

Estimator

$$\hat{\boldsymbol{f}} = \mathrm{softmax}(\boldsymbol{f})\ ;\ \hat{\boldsymbol{g}} = \mathrm{softmax}(\boldsymbol{g}), \tag{4}$$

$$t \sim \mathcal{T}(\hat{\boldsymbol{f}})\ \ ;\ \ u \sim \mathcal{T}(\hat{\boldsymbol{g}}), \tag{5}$$

$$\boldsymbol{I}_1 = \mathcal{I}_{(t)}\ \ ;\ \ \boldsymbol{I}_2 = \mathcal{I}_{(u)}, \tag{6}$$

$$\hat{\boldsymbol{h}} = \mathrm{softmax}(\boldsymbol{h})\ \ \rightarrow\ \ r \sim \mathcal{T}(\hat{\boldsymbol{h}}), \tag{7}$$

$$r1 = \min\left\{n \in [\,|\mathcal{I}|\,] : \sum_{k=1}^{n} k \geq r\right\}\ ;\ r2 = r - \sum_{k=1}^{r1\text{-}1} k, \tag{8}$$

$$f_1 = \mathcal{O}_{(r1)}\ ;\ f_2 = \mathcal{O}_{(r2)}, \tag{9}$$

$$\boldsymbol{H}_i^c = \sum_{r=1}^{\frac{(|\mathcal{O}||\mathcal{O}|+1)}{2}} \hat{\boldsymbol{h}}_{(r)} \times \left(\sum_{t=1}^{|\mathcal{I}|} \hat{\boldsymbol{f}}_{(r1)} \mathcal{O}_{(r1)}(\mathcal{I}_{(t)}) + \sum_{u=1}^{|\mathcal{I}|} \hat{\boldsymbol{g}}_{(u)} \mathcal{O}_{(r2)}(\mathcal{I}_{(u)})\right), \tag{10}$$

One-Shot Neural Network Architecture Search via Self-Evaluated Template Network

Algorithm 1 The Searching Algorithm of SETN

Input: the whole available training data
    a template network with ω and an estimator with α
  Split the whole available training data into the training set $\mathcal{D}_{train}$ and the validation set $\mathcal{D}_{val}$ for searching
  while not converge **do** ▷ Optimize ω and α
    Sample training batch $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{batch}$ from $\mathcal{D}_{train}$
    Calculate $\ell_{train} = \sum_{\mathcal{D}_t} \ell(x_i, y_i)$ based on Eq. (1)
    Update ω via gradients from the training loss ℓ_{train}
    Sample validation batch $\mathcal{D}_v = \{(x_i, y_i)\}_{i=1}^{batch}$ from $\mathcal{D}_{val}$
    Calculate $\ell_{val} = \sum_{\mathcal{D}_v} \ell(x_i, y_i)$ based on Eq. (10)
    Update α via gradients from the validation loss ℓ_{val}
  end while
  After the above steps, we obtain the optimized ω and α.
  Initialize $\mathcal{A} = \phi$ ▷ Obtain Low-Validation-Loss Candidates
  for i=1; i $\leq T$; i++ **do**
    Sample an architecture $\boldsymbol{a}$ using Eq. (4) to Eq. (9)
    $\mathcal{A} = \mathcal{A} \supset \{\boldsymbol{a}\}$
  end for
  Evaluate all candidates in $\mathcal{A}$ with parameters extracted from ω
  Select the candidate with the lowest validation loss
Output: the final selected candidate

One-Shot Neural Network Architecture Search via Self-Evaluated Template Network

Experiment Results

Dataset: CIFAR- 10, CIFAR- 100

Method	GPU Days	M	C	Parameters	Error on CIFAR-10 (%)	Error on CIFAR-100 (%)
DenseNet-BC [19]	–	–	–	25.6 MB	3.46	17.18
PyramidNet [17]	–	–	–	26.0 MB	3.31	16.35
MetaQNN [1]	>80	–	–	11.2 MB	6.92	27.14
Net Transformation [7]	10	–	–	19.7 MB	5.70	–
SMASH [6]	1.5	–	–	16.0 MB	4.03	–
Hierarchical NAS [24]	300	6	64	–	3.75	20.3
Progressive NAS [23]	150	11	48	3.2 MB	3.63	19.53
NASNet-A [43]	2000	20	32	3.3 MB	3.41	19.70
AmoebaNet-A [30]	3150	20	36	3.2 MB	3.34	–
ENAS [29]	0.45	20	36	4.6 MB	3.54	19.43
NAONet [28]	200	20	36	10.6 MB	3.18	–
NASNet-A + CutOut [43]	2000	20	32	3.3 MB	**2.65**	17.81†
PNAS + CutOut [23]	150	11	48	3.2 MB	–	17.63
DARTS + CutOut [25]	4	20	36	3.4 MB	2.83	–
GHN + CutOut [40]	0.84	18	32	5.7 MB	2.84	–
ENAS + CutOut [29]	0.45	20	36	4.6 MB	2.89	18.91†
GDAS + CutOut [14]	0.84	20	36	3.4 MB	2.93	18.38
SETN-LR (T=1K) + CutOut	1.8	20	36	5.5 MB	2.81	17.88
SETN-NON (T=1K) + CutOut	1.8	20	36	3.7 MB	3.12	18.27
SETN (T=1) + CutOut	1.7	20	36	4.5 MB	3.41	18.12
SETN (T=1K)	1.8	20	36	4.6 MB	3.56	19.38
SETN (T=1K) + CutOut	1.8	20	36	4.6 MB	2.69	**17.25**

Table 2. We compare SETN and other algorithms on CIFAR-10 and CIFAR-100. The top block presents state-of-the-art architectures designed by human experts. The bottom block presents architectures that are automatically discovered by machine. "M" indicates the total number of cells in the CNN, and "C" denotes the number of the filter channel in the first cell. "CutOut" indicates the data argumentation approach [11]. † denotes the results reproduced by ourself. The bottom five lines show results for different variants of our approach. We run each model three times and report the mean error (lower is better).

One-Shot Neural Network Architecture Search via Self-Evaluated Template Network

Experiment Results

Dataset: ImageNet

	Method	GPU days	Parameters	$+\times$ (million)	Top-1 Accuracy	Top-5 Accuracy
Human Expert	Inception-v1 [35]	–	6.6 MB	1448	69.8%	89.9%
	ResNet [18]	–	11.7 MB	1814	69.8%	89.1%
	MobileNet-v2 [33]	–	3.4 MB	300	72.0%	–
	ShuffleNet [41]	–	$\sim$5 MB	524	73.7%	–
NAS with more than 100 GPU days	Progressive NAS [23]	150	5.1 MB	588	74.2%	91.9%
	NASNet-A [43]	2000	5.3 MB	564	74.0%	91.6%
	NASNet-B [43]	2000	5.3 MB	488	72.8%	91.3%
	NASNet-C [43]	2000	4.9 MB	558	72.5%	91.0%
NAS with less than 5 GPU days	DARTS [25]	4	4.9 MB	595	73.1%	91.0%
	GHN [40]	0.84	6.1 MB	569	73.0%	91.3%
	SNAS [38]	1.5	4.3 MB	522	72.7%	90.8%
	GDAS [14]	0.84	5.3 MB	581	74.0%	91.5%
	SETN (N=1 & C=73)	1.8	5.2 MB	597	73.3%	91.4%
	SETN (N=2 & C=58)	1.8	5.3 MB	600	74.3%	91.6%
	SETN (N=3 & C=49)	1.8	5.3 MB	584	74.1%	91.9%
	SETN (N=4 & C=44)	1.8	5.4 MB	599	74.3%	92.0%

Table 3. We compare networks found by SETN and other approaches on ImageNet. We report the model size, the computation cost, the top-1 accuracy, and the top-5 accuracy. The top block shows the manually designed CNNs. The bottom two blocks indicate the automatically design CNNs. $+\times$ indicates the number of multiply-add operations.

Experiment Results

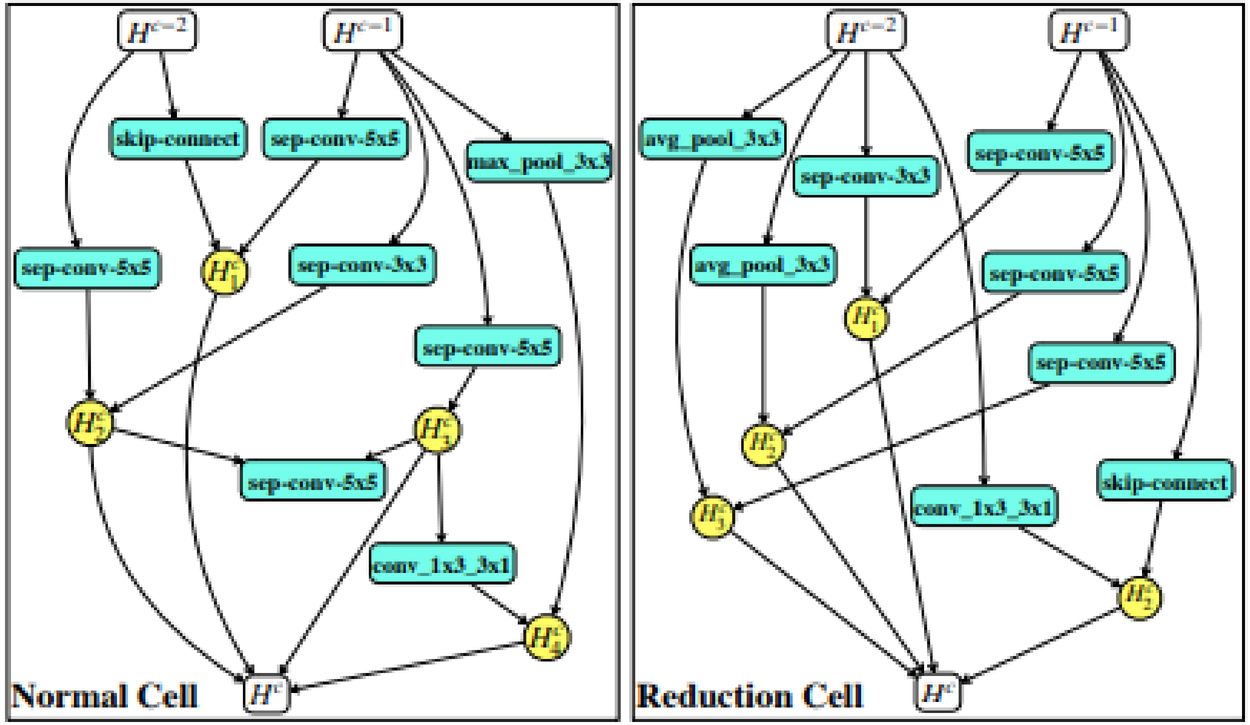

Figure 3. The left figure is the normal CNN cell that SETN discovered on CIFAR-10. The right figure is the reduction CNN cell that SETN discovered on CIFAR-10.