ICML | 2020
Thirty-seventh International Conference on Machine Learning

Breaking the Curse of Space Explosion: Towards Efficient NAS with Curriculum Search

Yong Guo*, Yaofo Chen*, Yin Zheng*, Peilin Zhao*, Jian Chen, Junzhou Huang, Mingkui Tan

Background and Motivation

Limitations of existing neural architecture search (NAS) methods:

- The search space is often **extremely large** (*e.g.*, billions of candidate architectures), resulting in the **space explosion issue**.
- NAS models become hard to train and often find **sub-optimal** architectures since they only receive **limited information** in the search process.

Contributions

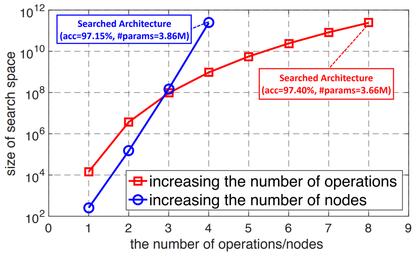
- We propose a novel **Curriculum Neural Architecture Search (CNAS)** method to alleviate the training difficulties of the NAS problem incurred by the extremely large search space.
- We propose a curriculum search method that **gradually incorporates the knowledge learned from a small search space**.
- Extensive experiments on several benchmark data sets show that the architectures found by our CNAS significantly **outperform the architectures obtained by state-of-the-art NAS methods**.

Search Space Size Analysis

Given B nodes and K candidate operations, the size of the search space Ω can be computed by

$$|\Omega| = K^{2(B-3)}\left((B-2)!\right)^2. \tag{1}$$

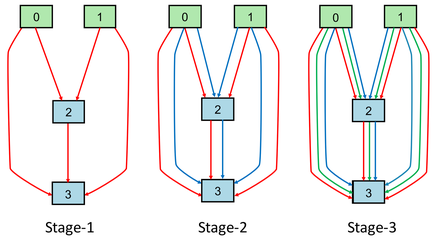
The search space can be **extremely large** when we have a large B or K.For example, ENAS has a search space of $|\Omega| \approx 5\times10^{12}$ with B=8 and K=5, and DARTS has a search space of $|\Omega| \approx 2\times10^{11}$ with B=7 and K=8.



Increasing the number of nodes make the size of search space **grow faster** than increasing the number of operations.

Curriculum Neural Architecture Search

We propose a novel **Curriculum Neural Architecture Search (CNAS)** to enlarge the search space by **gradually increasing the number of candidate operations** from 1 to K.



Curriculum Training Scheme

We progressively train the controller to solve the problems with **different search spaces**, which are corresponding to **different stages (from 1 to K)** in the training process. Let Ω_i be the search space of the i-th stage. The training objective in the i-th stage becomes

$$\begin{aligned} &\max_{\theta} \ \mathbb{E}_{\alpha\sim\pi(\cdot;\theta,\Omega_i)}\left[\mathcal{R}\left(\alpha, w^*(\alpha)\right)\right] + \lambda H\left(\pi\left(\cdot;\theta,\Omega_i\right)\right), \\ &\text{s.t. } w^*(\alpha) = \arg\min_{w} \mathcal{L}\left(\alpha, w\right), \end{aligned} \tag{2}$$

- $\pi(\cdot;\theta,\Omega_i)$ denotes the learned policy *w.r.t.* Ω_i, $H(\cdot)$ evaluates the entropy of the policy, and λ controls the strength of the entropy regularization term.
- This **entropy term** enables CNAS to explore the unseen areas of previous search stages and thus **escape from local optima**.

Operation Warmup

To address the issue that architectures with the **new operation** have **poor performance**, we propose an effective **operation warmup** method.

- When we **add a new operation**, we **fix the controller** model and only train the parameters of the super network.
- To improve the fairness of operations, we **uniformly** sample candidate architectures to **train each operation with equal probability**.
- The candidate architectures with the **newly added operation** achieve **comparable performance** with the existing ones.

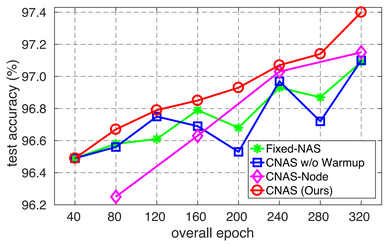
Training Method

Algorithm 1: Training method for CNAS.

Require: The operation sequence O, learning rate η, the number of iterations for operation warmup M, the uniform distribution of architectures $p(\cdot)$, controller's policy $\pi(\cdot)$, supernet parameters w, controller parameters θ.

Initialize w and θ, $\Omega_0 = \emptyset$.
for i=1 to $|O|$ **do**
  Enlarge Ω_i by adding O_i to the set of candidate operations;
  // Operation warmup
  for j=1 to M **do**
    Sample $\alpha \sim p(\alpha;\Omega_i)$;
    $w \leftarrow w - \eta\nabla_w\mathcal{L}(\alpha, w)$;
  end for
  while not convergent **do**
    // Update θ by maximizing the reward
    for each iteration on validation data **do**
      Sample $\alpha \sim \pi(\alpha;\theta,\Omega_i)$;
      Update the controller by ascending its gradient:
        $\mathcal{R}(\alpha, w)\nabla_\theta \log\pi(\alpha;\theta,\Omega_i)+\lambda H(\pi(\cdot;\theta,\Omega_i))$;
    end for
    // Update w by minimizing the training loss
    for each iteration on training data **do**
      Sample $\alpha \sim \pi(\alpha;\theta,\Omega_i)$;
      $w \leftarrow w - \eta\nabla_w\mathcal{L}(\alpha, w)$.
    end for
  end while
end for

Ablation Study



- **Fixed-NAS**: For each stage, we keep the search space **fixed** and **train a controller from scratch**.
- **CNAS-Node**: We train the controller in a growing search space by **gradually adding new nodes**.
- **CNAS (Ours)**: We train the controller in a growing search space by **gradually adding new operations**.

Comparisons with SToA Methods

- **Comparisons with state-of-the-art methods on CIFAR-10**

Architecture	Test Accuracy (%)	Params (M)	Search Cost (GPU days)
DenseNet-BC	96.54	25.6	–
PyramidNet-BC	96.69	26.0	–
Random search baseline	96.71 ± 0.15	3.2	–
NASNet-A + cutout	97.35	3.3	1800
NASNet-B	96.27	2.6	1800
NASNet-C	96.41	3.1	1800
AmoebaNet-A + cutout	96.66 ± 0.06	3.2	3150
AmoebaNet-B + cutout	96.63 ± 0.04	2.8	3150
DSO-NAS	97.05	3.0	1
Hierarchical Evo	96.25 ± 0.12	15.7	300
SNAS	97.02	2.9	1.5
ENAS + cutout	97.11	4.6	0.5
NAONet	97.02	28.6	200
NAONet-WS	96.47	2.5	0.3
GHN	97.16 ± 0.07	5.7	0.8
PNAS + cutout	97.17 ± 0.07	3.2	225
DARTS + cutout	97.24 ± 0.09	3.4	4
CARS + cutout	97.38	3.6	0.4
CNAS + cutout	**97.40 ± 0.06**	3.7	0.3

- **Comparisons with state-of-the-art methods on ImageNet**

Architecture	Test Accuracy (%)		#Params (M)	#MAdds (M)	Search Cost (GPU days)
	Top-1	Top-5			
ResNet-18	69.8	89.1	11.7	1814	–
Inception-v1	69.8	89.9	6.6	1448	–
MobileNet	70.6	89.5	4.2	569	–
NASNet-A	74.0	91.6	5.3	564	1800
NASNet-B	72.8	91.3	5.3	488	1800
NASNet-C	72.5	91.0	4.9	558	1800
AmoebaNet-A	74.5	92.0	5.1	555	3150
AmoebaNet-B	74.0	92.4	5.3	555	3150
GHN	73.0	91.3	6.1	569	0.8
SNAS	72.7	90.8	4.3	522	1.5
DARTS	73.1	91.0	4.9	595	4
NAT-DARTS	73.7	91.4	4.0	441	-
PNAS	73.5	91.4	5.1	588	255
MnasNet-92	74.8	92.0	4.4	-	-
ProxylessNAS	75.1	92.5	7.1	-	8.3
CARS	75.2	92.5	5.1	591	0.4
CNAS	**75.4**	**92.6**	5.3	576	**0.3**

Contact Information and Code

- Email: mingkuitan@scut.edu.cn
- Code: https://github.com/guoyongcs/CNAS