Neural Architecture Search Using Deep Neural Networks and Monte Carlo Tree Search

Galina Boeva

MIPT, 2024

April 2, 2024

Motivation

Main idea

NAS has demonstrated great success in automating the design of neural networks, but the prohibitively large amount of computing underlying NAS methods requires additional research to improve sampling efficiency and the cost of evaluating the network to achieve better results in less time.

AlphaX: A Scalable MCTS Design Agent

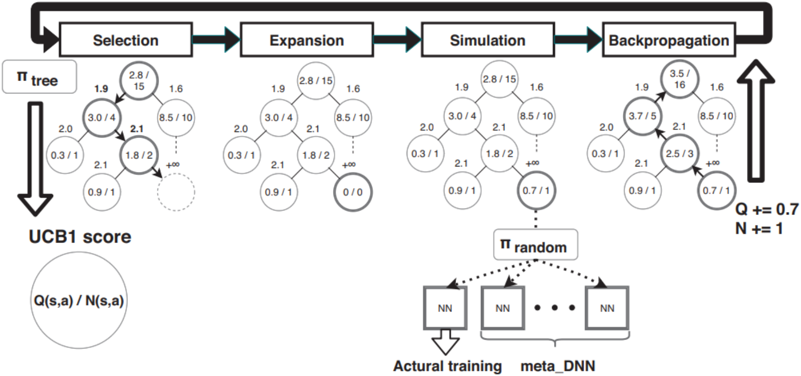
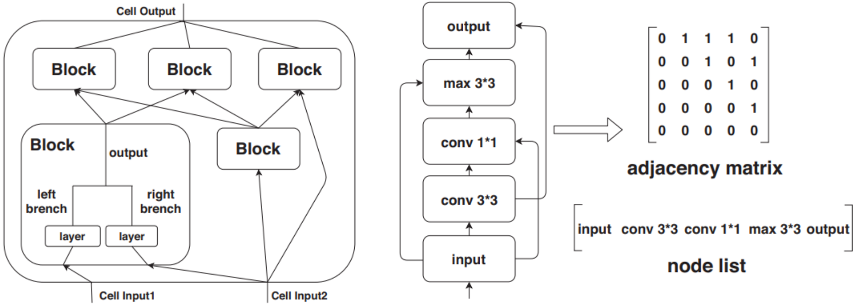


Figure: Overview of AlphaX search procedures.

Design, State and Action Space



(a) NASNet Cell (b) NASBench DAG

Figure: Design space: (a) the cell structure of NASNet and (b) the DAG structure of NASBench-101. Then the network is constructed by stacking multiple cells or DAGs.

Search Procedure

Selection

Moves down the search tree to trace the current most promising search path. It starts at the root and ends until it reaches the leaf. In the node, the agent selects actions based on UCB1:

$$\pi_{tree}(s) = \arg\max_{a \in A} \left(\frac{Q(s,a)}{N(s,a)} + 2c\sqrt{\frac{2\log N(s)}{N(s,a)}} \right),$$

where $N(s)$ is the number of visits to the state s (i.e. $N(s) = \sum_{a \in A} N(s,a)$), and c is a constant.

Expansion

Adds a new node into the tree. $Q(s,a)$ and $N(s,a)$ are initialized to zeros. $Q(s,a)$ will be updated in the simulation step.

Search Procedure

Meta-DNN assisted Simulation

Randomly samples the descendants of a new node to approximate $Q(s, a)$ of the node with their accuracies.

$$Q(s,a) \leftarrow \left(Acc(sim_0(s^{'})) + \frac{1}{k} \sum_{i=1\ldots k} Pred(sim_i(s^{'})) \right) /2,$$

where $s = s + a$, and $sim(s)$ represents a simulation starting from state s. Acc — trained accuracy in the first simulation, Pred — the predicted accuracy from Meta-DNN in subsequent k simulations.

Backpropagation

Backtracks the search path from the new node to the root to update visiting statistics. With the estimated q for the new node, we iteratively backpropagate the information to its ancestral as:

$$Q(s,a) \leftarrow Q(s,a) + q, N(s,a) \leftarrow N(s,a) + 1, s \leftarrow parent(s), a \leftarrow \pi_{tree}(s)$$

The design of Meta-DNN and its related issues

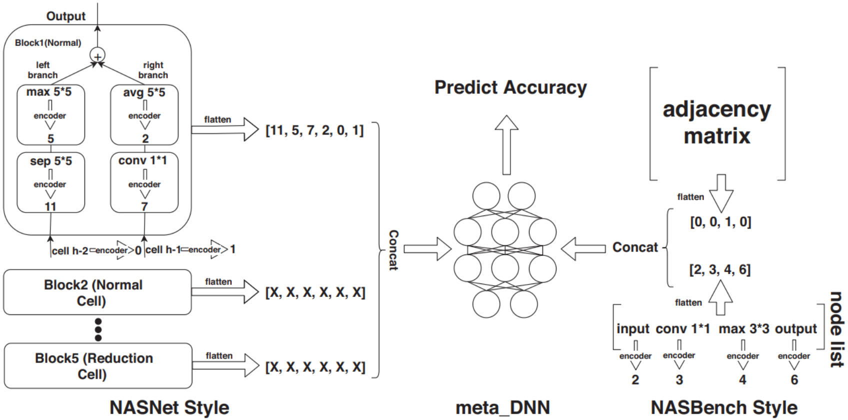


Figure: Encoding scheme of NASBench and NASNet.

Distributed AlphaX

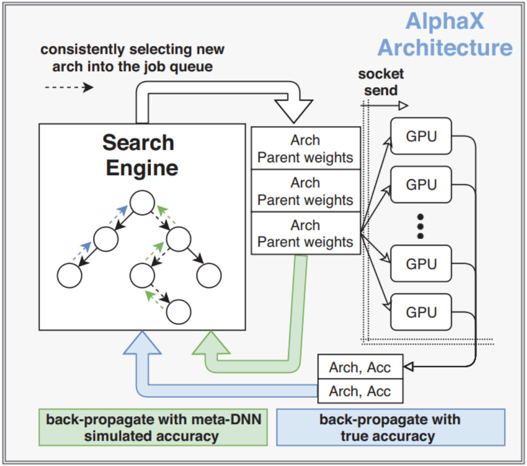


Figure: Distributed AlphaX: we decouple the original back-propagation into two parts: one uses predicted accuracy (green arrow), while the other uses the true accuracy (blue arrow).

Evaluations of architecture search. Open domain search.

Model	Params	Err	GPU days	M
NASNet-A+cutout (Zoph et al. 2017)	3.3M	2.65	2000	20000
AmoebaNet-B+cutout (Real et al. 2018)	2.8M	$2.50_{\pm 0.05}$	3150	27000
DARTS+cutout (Liu et al. 2018)	3.3M	$2.76_{\pm 0.09}$	4	-
RENASNet+cutout (Chen et al. 2019)	3.5M	$2.88_{\pm 0.02}$	6	4500
AlphaX+cutout (32 filters)	2.83M	$2.54_{\pm 0.06}$	12	1000
PNAS (Liu et al. 2017a)	3.2M	$3.41_{\pm 0.09}$	225	1160
ENAS (Pham et al. 2018)	4.6M	3.54	0.45	-
NAONet (Luo et al. 2018)	10.6M	3.18	200	1000
AlphaX (32 filters)	2.83M	$3.04_{\pm 0.03}$	12	1000
NAS v3(Zoph and Le 2016)	7.1M	4.47	22400	12800
Hier-EA (Liu et al. 2017c)	15.7M	$3.75_{\pm 0.12}$	300	7000
AlphaX+cutout (128 filters)	31.36M	$2.16_{\pm 0.04}$	12	1000

Figure: The comparisons of our NASNet search results to other state-of-the-art results on CIFAR-10. M is the number of sampled architectures in the search.

Evaluations of architecture search. Searching on NAS dataset.

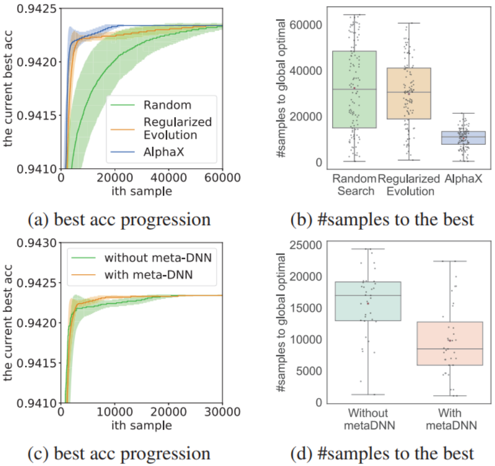


Figure: Finding the global optimum on NASBench-101: AlphaX is 3x, 2.8x faster than Random Search and Regularized Evolution on NASBench-101 (nodes 6).

Evaluations of architecture search. Qualitative evaluations of AlphaX.

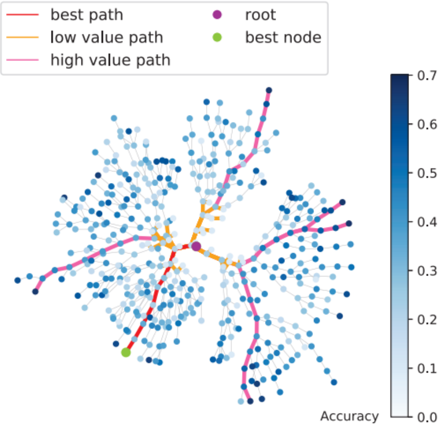
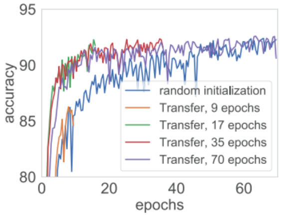
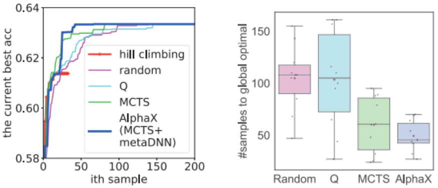


Figure: AlphaX search visualization:each node represents a MCTS state; the node color reflects its value, i.e. accuracy, indicating how promising a search branch.

Component evaluations. Transfer learning.





(a) best acc progression (b) #samples to the best

Figure: **1.** Validation of transfer learning: transferring weights significantly reduces the number of epochs in reaching the same accuracy of random initializations (Transfer 17 $\rightarrow$ 70 epochs v.s. random initialization), but insufficient epochs loses accuracy (Transfer, 9 epochs). **2.** Algorithmic comparisons: AlphaX is consistently the fastest algorithm to reach the global optimal on another simplified search domain, while Hill Climbing can easily trap into a local optimal.

Algorithm Comparisons

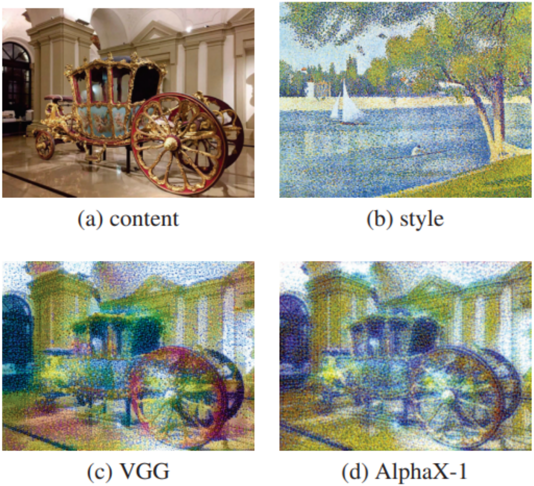


Figure: Neural Style Transfer: AlphaX-1 v.s. VGG.

Literature

1. **Main article** Neural Architecture Search Using Deep Neural Networks and Monte Carlo Tree Search.