Simple and Efficient Architecture Search for Convolution Neural Networks

—— ICLR 2018 Workshop

薛焕然 1701214297

2018.7.2

Background

Introduction

Peking University

Neural Networks have rapidly gained popularity over the last few years

However, In most cases Neural Networks are designed by hand, which is

exhausting

time-consuming

A Natural goal : design optimization algorithms to automate search process

Similar to Model Selection

Related Work

Peking University

Bayesian Optimization

Reinforcement Learning

Evolutionary Algorithms

However….
Most methods are very costly (requiring hundreds or thousands of GPU days)

Approach in this paper

Overview

Peking University

Hill Climbing Strategy(local search)

Starting with a small, (**possibly**) pretrained network, then apply **network morphisms** to generate larger ones that may perform better

Network Morphisms:

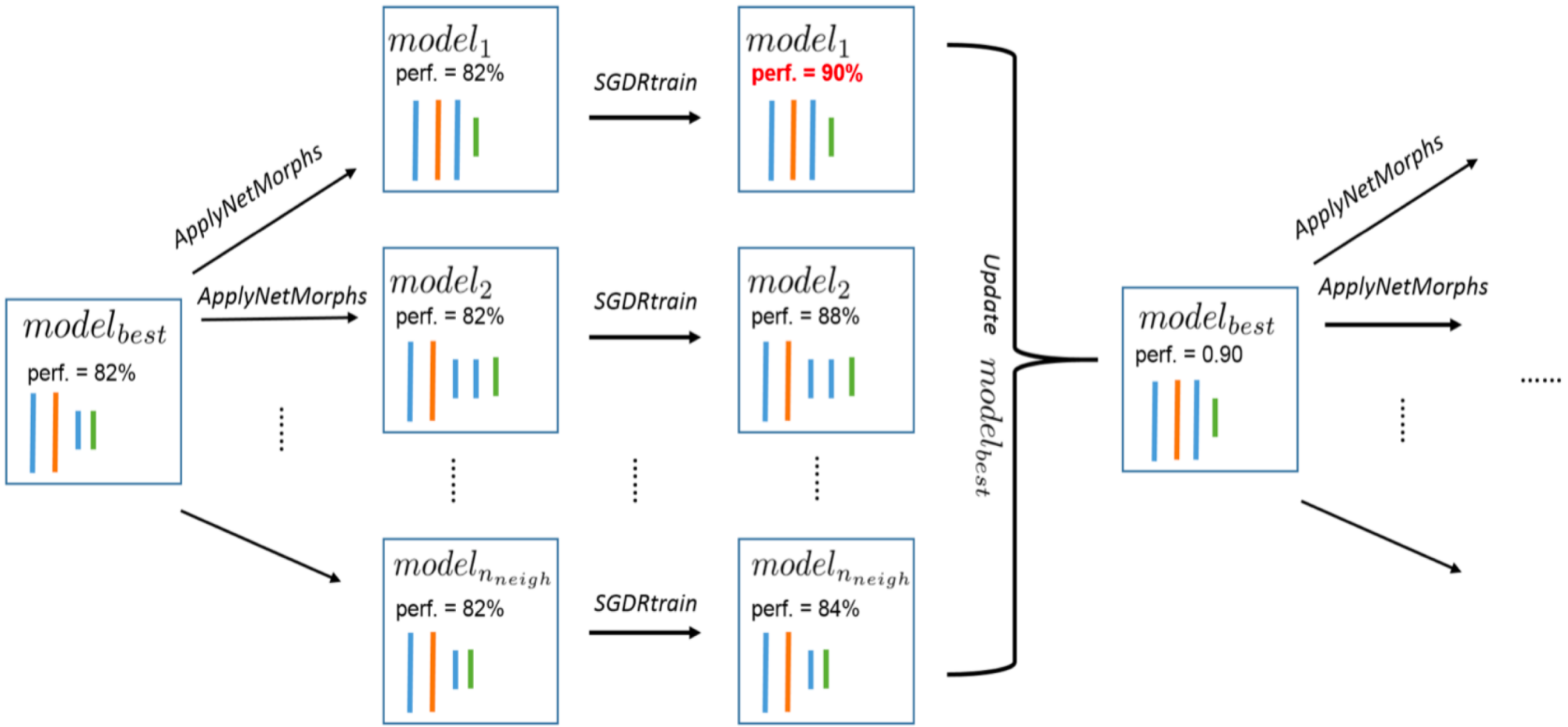
Make the network deeper

Make the network wider

Add a skip connection from layer i to layer j

Search Method

Peking University



Network Morphism

Peking University

Tianqi Chen, Ian J. Goodfellow, and Jonathon Shlens. Net2net: Accelerating learning via knowledge transfer. arXiv 2015, ICLR 2016.

Suppose a parent network is $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$, where $\boldsymbol{x}$ is the input to the network, $\boldsymbol{y}$ is the output of the network, $\boldsymbol{\theta}$ is the parameters of the network

The strategy of network morphism is to choose a new set of parameters $\boldsymbol{\theta}'$ for a child network, such that

$$\forall \boldsymbol{x}, f(\boldsymbol{x}; \boldsymbol{\theta}) = g(\boldsymbol{x}, \boldsymbol{\theta}')$$

Two basic operations: Net2WiderNet Net2DeeperNet

Example: Net2WiderNet

Suppose layer i and layer $i+1$ are both fully-connected

Weight matrix $W^{(i)} \in \mathrm{R}^{m \times n}, W^{(i+1)} \in \mathrm{R}^{n \times p}$

Replace layer i with a new layer with q outputs, with $q > n$

Introduce a random mapping function:

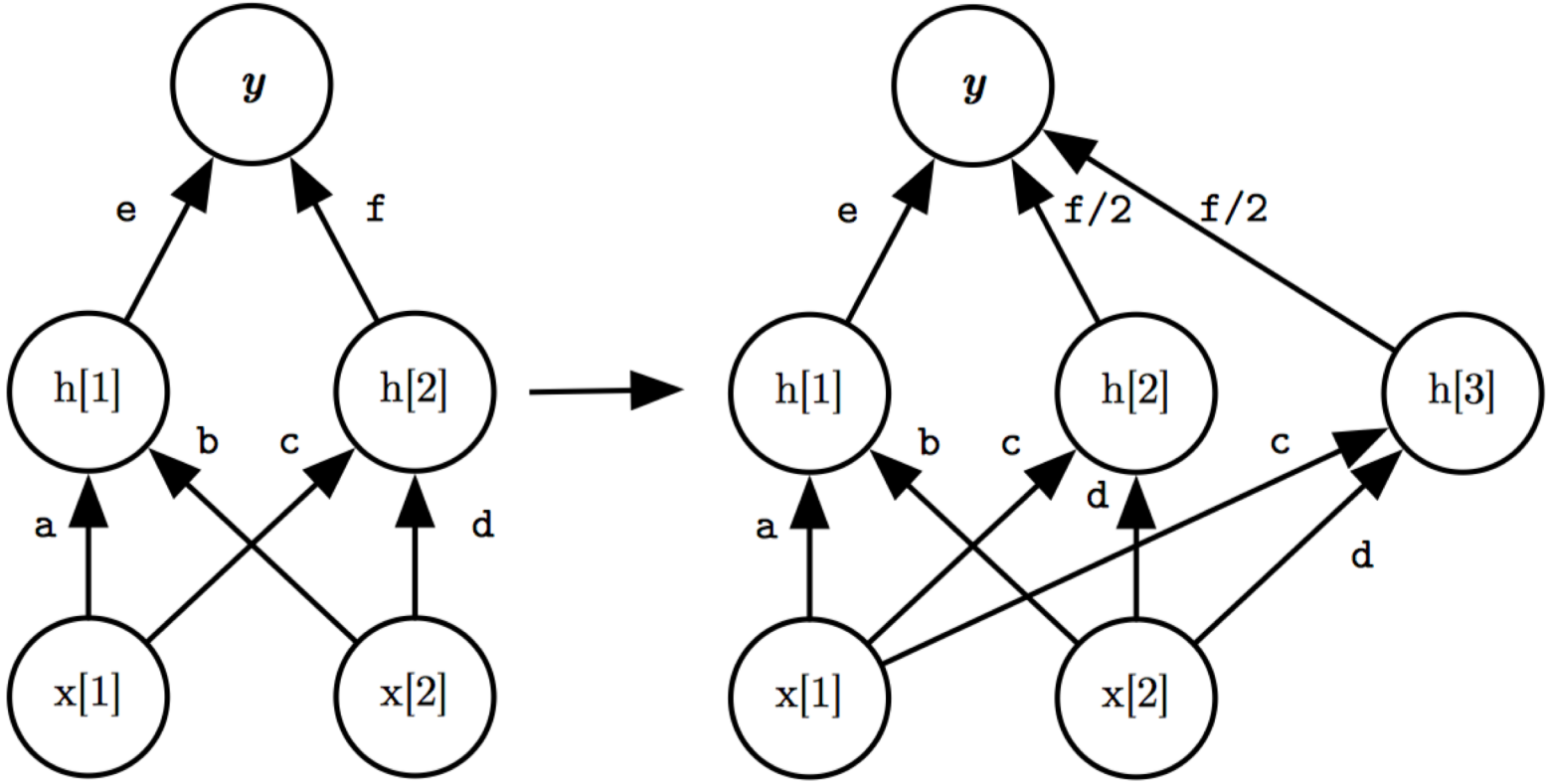
$$g(j) = \begin{cases} j & j \leq n \\ \text{random sample from } \{1, 2, \cdots n\} & j > n \end{cases}$$

$U^{(i)}$ and $U^{(i+1)}$ are new weight matrices, with

$$\boldsymbol{U}_{k,j}^{(i)} = \boldsymbol{W}_{k,g(j)}^{(i)}, \quad \boldsymbol{U}_{j,h}^{(i+1)} = \frac{1}{|\{x|g(x) = g(j)\}|} \boldsymbol{W}_{g(j),h}^{(i+1)}.$$

Example: Net2WiderNet

Peking University



Network Search by hill climbing

Peking University

function NASH($model_0, n_{steps}, n_{neigh}, n_{NM}, epoch_{neigh}, epoch_{final}, \lambda_{end}, \lambda_{start}$)

$model_0 \triangleq$ model to start with, $n_{steps} \triangleq$ number of hill climbining steps
$n_{neigh} \triangleq$ number of neighbours, $n_{NM} \triangleq$ number of net. morph. applied
$epoch_{neigh} \triangleq$ number of epochs for training every neighbour
$epoch_{final} \triangleq$ number of epochs for final training
initial LR λ_{start} is annealed to λ_{end} during SGDR training

$model_{best} \leftarrow model_0$
# start hill climbing
for $i \leftarrow 1, \ldots, n_{steps}$ **do**
  #get n_{neigh} neighbors of $model_0$ by applying n_{NM} network morphisms to $model_{best}$
  for $j \leftarrow 1, \ldots, n_{neigh} - 1$ **do**
    $model_j \leftarrow ApplyNetMorphs(model_{best}, n_{NM})$
    # train for a few epochs on training set with SGDR
    $model_j \leftarrow \text{SGDRtrain}(model_j, epoch_{neigh}, \lambda_{start}, \lambda_{end})$
  end for
  # in fact, last neighbor is always just the current best
  $model_{n_{neigh}} \leftarrow \text{SGDRtrain}(model_{best}, epoch_{neigh}, \lambda_{start}, \lambda_{end})$
  # get best model on validation set
  $model_{best} \leftarrow \underset{j=1,\ldots,n_{neigh}}{arg\,max} \{performance_{vali}(model_j)\}$
end for
# train the final model on training and validation set
$model_{best} \leftarrow \text{SGDRtrain}(model_{best}, epoch_{final}, \lambda_{start}, \lambda_{end})$
return $model_{best}$
end function

Experiments

Experiments Setting

Peking University

Dataset:

Cifar10/Cifar100

Standard Data Augmentation Scheme

40000 training samples, 10000 valid samples, 10000 test samples

Nvidia Titan X

Baseline

Peking University

algorithm setting	runtime (hrs)	# params (mil.)	error $\pm$ std. (%)
$n_{neigh} = 8$	12.8	5.7	5.7 ± 0.35
Random networks ($n_{neigh} = 1$)	4.5	4.4	6.5 ± 0.76
models from line 1 retrained from scratch	5.3	5.7	6.1 ± 0.92
$n_{neigh} = 8$, no SGDR	10.6	5.8	6.4 ± 0.70
$n_{neigh} = 8$, no net. morph.	6.6	2.9	6.1 ± 0.30

Baseline experiments. Runtime, # params and error rates are averaged over 10 runs and 30 runs respectively.

Comparison on Cifar10

Peking University

model	resources spent	# params (mil.)	error (%)
Shake-Shake (Gastaldi, 2017)	4 GPU days, 2 GPUs	26	2.9
WRN 28-10 (Loshchilov & Hutter, 2017)	1 GPU day	36.5	3.86
Baker et al. (2016)	80-100 GPU days	11	6.9
Cai et al. (2017)	15 GPU days	19.7	5.7
Zoph & Le (2017)	16.000-24.000 GPU days	37.5	3.65
Real et al. (2017)	2500 GPU days	5.4	5.4
Saxena & Verbeek (2016)	?	21	7.4
Brock et al. (2017)	3 GPU days	16.0	4.0
Ours (random networks, $n_{steps} = 5, n_{neigh} = 1$)	0.2 GPU days	4.4	6.5
Ours ($n_{steps} = 5, n_{neigh} = 8$, 10 runs)	0.5 GPU days	5.7	5.7
Ours ($n_{steps} = 8, n_{neigh} = 8$, 4 runs)	1 GPU day	19.7	5.2
Ours (snapshot ensemble, 4 runs)	2 GPU days	57.8	4.7
Ours (ensemble across runs)	4 GPU days	88	4.4

Generate competitive network architectures in only 12 hours

Comparison on Cifar100

Peking University

model	resources spent	# params (mil.)	error (%)
Shake-Shake (Gastaldi, 2017)	14 GPU days	34.4	15.9
WRN 28-10 (Loshchilov & Hutter, 2017)	1 GPU day	36.5	19.6
Real et al. (2017)	250 GPUs	40.4	23.7
Brock et al. (2017)	3 GPU days	16.0	20.6
Ours ($n_{steps} = 8, n_{neigh} = 8$, 5 runs)	1 GPU day	22.3	23.4
Ours (snapshot ensemble, 5 runs)	2 GPU days	73.3	20.9
Ours (ensemble across runs)	5 GPU days	111.5	19.6

Baseline experiments. Runtime, # params and error rates are averaged over 10 runs and 30 runs respectively.

Concolusion

Conclusion

Peking University

Less computational resources than most alternative approaches

How to escape the local optima

Serve as a basis for the development of more sophisticated methods that yield further improvements of performance

Peking University

Q & A