Progressive Neural Architecture Search

Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, Kevin Murphy
09/10/2018 @ECCV
Outline

Introduction and Background

Architecture Search Space

Progressive Neural Architecture Search Algorithm

Experiments and Results
Introduction and Background
AutoML

- Hit Enter, sit back and relax, come back the next day for a high-quality machine learning solution ready to be delivered
What Is Preventing Us?

<table>
<thead>
<tr>
<th>Machine Learning solution</th>
<th>Parameter</th>
<th>Hyperparameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural Network</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What Is Preventing Us?

<table>
<thead>
<tr>
<th>Machine Learning solution</th>
<th>Parameter</th>
<th>Hyperparameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural Network</td>
<td>Automated :)</td>
<td></td>
</tr>
</tbody>
</table>
## What Is Preventing Us?

<table>
<thead>
<tr>
<th>Machine Learning solution</th>
<th>Parameter</th>
<th>Hyperparameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural Network</td>
<td>Automated :)</td>
<td>Not quite automated :(</td>
</tr>
</tbody>
</table>
Where Are Hyperparameters?

- We usually think of those related to learning rate scheduling
- But for a neural network, many more lie in its architecture:

Neural Architecture Search (NAS)

- Can we design network architectures automatically, instead of relying on expert experience and knowledge?

- Broadly, existing NAS literatures fall into two main categories:
  - Evolutionary Algorithms (EA)
  - Reinforcement Learning (RL)
Evolutionary Algorithms for NAS

Best candidates

(0, 1, 0, 1): 0.85
(2, 0, 3, 1): 0.84
(5, 1, 3, 3): 0.91
(0, 2, 0, 6): 0.92
(0, 7, 3, 5): 0.82

String that defines network architecture
Accuracy on validation set
Evolutionary Algorithms for NAS

Best candidates

(0, 1, 0, 1): 0.85
(2, 0, 3, 1): 0.84
(5, 1, 3, 3): 0.91
(0, 2, 0, 6): 0.92
...
(0, 7, 3, 5): 0.82

New candidates

(0, 1, 0, 2): ??
(2, 0, 4, 1): ??
(5, 5, 3, 3): ??
(0, 2, 1, 6): ??
...
(0, 6, 3, 5): ??
Evolutionary Algorithms for NAS

Best candidates

(0, 1, 0, 1): 0.85
(2, 0, 3, 1): 0.84
(5, 1, 3, 3): 0.91
(0, 2, 0, 6): 0.92
...
(0, 7, 3, 5): 0.82

New candidates

(0, 1, 0, 2): 0.86
(2, 0, 4, 1): 0.83
(5, 5, 3, 3): 0.90
(0, 2, 1, 6): 0.91
...
(0, 6, 3, 5): 0.80
Evolutionary Algorithms for NAS

Best candidates

(5, 5, 3, 3): 0.90
(0, 2, 1, 6): 0.91
(5, 1, 3, 3): 0.91
(0, 2, 0, 6): 0.92
...
(0, 1, 0, 2): 0.86

New candidates

(0, 1, 0, 2): 0.86
(2, 0, 4, 1): 0.83
(5, 5, 3, 3): 0.90
(0, 2, 1, 6): 0.91
...
(0, 6, 3, 5): 0.80
Reinforcement Learning for NAS

LSTM Agent

0, 1, 0, 2!

computing...

GPU/TPU
Reinforcement Learning for NAS

LSTM Agent

updating...

0.86!

GPU/TPU
Reinforcement Learning for NAS

5, 5, 3, 3!

computing...

LSTM Agent

GPU/TPU
Reinforcement Learning for NAS

LSTM Agent

updating...

0.90!

GPU/TPU
Success and Limitation

- NASNet from Zoph et al. (2018) already surpassed human designs on ImageNet under the same # Mult-Add or # Params

- But very computationally intensive:
  - Zoph & Le (2017): 800 K40 for 28 days
  - Zoph et al. (2018): 500 P100 for 5 days

Zoph, Barret, and Quoc V. Le. "Neural architecture search with reinforcement learning." In ICLR. 2017.
Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. "Learning transferable architectures for scalable image recognition." In CVPR. 2018.
Our Goal

- NASNet from Zoph et al. (2018) already surpassed human designs on ImageNet under the same # Mult-Add or # Params

- But very computationally intensive:
  - Zoph & Le (2017): 800 K40 for 28 days
  - Zoph et al. (2018): 500 P100 for 5 days

- **Our goal:** Speed up NAS by proposing an alternative algorithm
Architecture
Search Space
Similar to Zoph et al. (2018)

Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. "Learning transferable architectures for scalable image recognition." In CVPR. 2018.
Cell -> Network

- Once we have a cell structure, we stack it up using a predefined pattern
- A network is fully specified with:
  - Cell structure
  - $N$ (number of cell repetition)
  - $F$ (number of filters in the first cell)
- $N$ and $F$ are selected by hand to control network complexity
Block -> Cell

- Each cell consists of $B=5$ blocks
- The cell’s output is the concatenation of the 5 blocks’ outputs
Within a Block

- Input 1 is transformed by Operator 1
- Input 2 is transformed by Operator 2
- Combine to give block’s output
Within a Block

- **Input 1** and **Input 2** may select from:
  - Previous cell’s output
  - Previous-previous cell’s output
  - Previous blocks’ output in current cell
Within a Block

- **Operator 1** and **Operator 2** may select from:
  - 3x3 depth-separable convolution
  - 5x5 depth-separable convolution
  - 7x7 depth-separable convolution
  - 1x7 followed by 7x1 convolution
  - Identity
  - 3x3 average pooling
  - 3x3 max pooling
  - 3x3 dilated convolution
Within a Block

- **Combination** is element-wise addition
Architecture Search Space Summary

- One cell may look like:

- $2^2 \times 8^2 \times 1 \times 3^2 \times 8^2 \times 1 \times 4^2 \times 8^2 \times 1 \times 5^2 \times 8^2 \times 1 \times 6^2 \times 8^2 \times 1 = 10^{14}$ possible combinations!
Progressive Neural Architecture Search Algorithm
Main Idea: Simple-to-Complex Curriculum

- Previous approaches directly work with the $10^{14}$ search space

- Instead, what if we progressively work our way in:
  - Begin by training all 1-block cells. There are only 256 of them!
  - Their scores are going to be low, because of they have fewer blocks...
  - But maybe their relative performances are enough to show which cells are promising and which are not.
  - Let the $K$ most promising cells expand into 2-block cells, and iterate!
Progressive Neural Architecture Search: First Try

● **Problem**: for a reasonable $K$, too many 2-block candidates to train
  ○ It is “expensive” to obtain the performance of a cell/string
  ○ Each one takes hours of training and evaluating
  ○ Maybe can afford $10^2$, but definitely cannot afford $10^5$
Performance Prediction with Surrogate Model

- **Solution**: train a “cheap” surrogate model that predicts the final performance simply by reading the string
  - The data points collected in the “expensive” way are exactly training data for this “cheap” surrogate model

- The two assessments are in fact used in an alternate fashion:
  - Use “cheap” assessment when candidate pool is large (~$10^5$)
  - Use “expensive” assessment when it is small (~$10^2$)
Performance Prediction with Surrogate Model

- Desired properties of this surrogate model/predictor:
  - Handle variable-size input strings
  - Correlate with true performance
  - Sample efficient

- We try both a MLP-ensemble and a RNN-ensemble as predictor
  - MLP-ensemble handles variable-size by mean pooling
  - RNN-ensemble handles variable-size by unrolling a different number of times
Progressive Neural Architecture Search

predictor

$B_1(256)$

enumerate and train all 1-block cells
Progressive Neural Architecture Search

- Train predictor
- Enumerate and train all 1-block cells
Progressive Neural Architecture Search

- Predict \( K \times B_2 (\sim 10^5) \)
- \( B_1 (256) \)
- Expand promising 2-block cells
- Train predictor
- Enumerate and train all 1-block cells
Progressive Neural Architecture Search

- enumerate and train all 1-block cells
- expand promising 2-block cells
- train predictor
- apply predictor to select top $K$
- enumerate and train all 1-block cells
Progressive Neural Architecture Search

- enumerate and train all 1-block cells
- expand promising 2-block cells
- train predictor
- train the selected 2-block cells
- apply predictor to select top $K$
- enumerate and train all 1-block cells
Progressive Neural Architecture Search

- enumerate and train all 1-block cells
- expand promising 2-block cells
- train predictor
- apply predictor to select top $K$
- train the selected 2-block cells
- finetune predictor
- enumerate and train all 1-block cells

$K \sim 10^2$

$K \cdot B_2 \sim 10^5$

$B_1 = 256$
Progressive Neural Architecture Search

- Enumerate and train all 1-block cells
- Expand promising 2-block cells
- Train predictor
- Apply predictor to select top K
- Expand promising 3-block cells
- Fine-tune predictor
- Train the selected 2-block cells
- Train predictor
- Enumerate and train all 1-block cells
Progressive Neural Architecture Search

apply predictor to select top $K$
expand promising 3-block cells
finetune predictor
train the selected 2-block cells
apply predictor to select top $K$
expand promising 2-block cells
train predictor
enumerate and train all 1-block cells
Experiments and Results
The Search Process

- We performed Progressive Neural Architecture Search ($K = 256$) on CIFAR-10
- Each model ($N = 2, F = 24$) was trained for 20 epochs with cosine learning rate
- First big question: Is our search more efficient?
The Search Process: 5x Speedup
The Search Process: PNASNet-1, 2, 3
The Search Process: PNASNet-4
The Search Process: PNASNet-5

$H^c$

concat

$H^{c-1}$

$H^{c-2}$

sep 7x7  max 3x3  sep 5x5  sep 3x3

max 3x3

sep 3x3

iden

ty

sep 3x3  sep 5x5  max 3x3
After The Search

- Select the best 5-block cell structure; increase $N$ and $F$
- Train and evaluate on both CIFAR-10 and ImageNet
- Second big question: How competitive is the found cell structure on benchmark datasets?
## After The Search: CIFAR-10

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th>Error Rate</th>
<th>Method</th>
<th>Search Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>NASNet-A [1]</td>
<td>3.3M</td>
<td>3.41</td>
<td>RL</td>
<td>21.4 - 29.3B</td>
</tr>
<tr>
<td>NASNet-B [1]</td>
<td>2.6M</td>
<td>3.73</td>
<td>RL</td>
<td>21.4 - 29.3B</td>
</tr>
<tr>
<td>NASNet-C [1]</td>
<td>3.1M</td>
<td>3.59</td>
<td>RL</td>
<td>21.4 - 29.3B</td>
</tr>
<tr>
<td>Hier-EA [2]</td>
<td>15.7M</td>
<td>3.75 ± 0.12</td>
<td>EA</td>
<td>35.8B</td>
</tr>
<tr>
<td>AmoebaNet-B [3]</td>
<td>2.8M</td>
<td>3.37 ± 0.04</td>
<td>EA</td>
<td>63.5B</td>
</tr>
<tr>
<td>AmoebaNet-A [3]</td>
<td>3.2M</td>
<td>3.34 ± 0.06</td>
<td>EA</td>
<td>25.2B</td>
</tr>
<tr>
<td>PNASNet-5</td>
<td>3.2M</td>
<td>3.41 ± 0.09</td>
<td>SMBO</td>
<td>1.0B</td>
</tr>
</tbody>
</table>

### After The Search: ImageNet (Mobile)

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th># Mult-Add</th>
<th>Top 1</th>
<th>Top 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobileNet [1]</td>
<td>4.2M</td>
<td>569M</td>
<td>70.6</td>
<td>89.5</td>
</tr>
<tr>
<td>ShuffleNet [2]</td>
<td>5M</td>
<td>524M</td>
<td>70.9</td>
<td>89.8</td>
</tr>
<tr>
<td>NASNet-A [3]</td>
<td>5.3M</td>
<td>564M</td>
<td>74.0</td>
<td>91.6</td>
</tr>
<tr>
<td>AmoebaNet-B [4]</td>
<td>5.3M</td>
<td>555M</td>
<td>74.0</td>
<td>91.5</td>
</tr>
<tr>
<td>AmoebaNet-A [4]</td>
<td>5.1M</td>
<td>555M</td>
<td>74.5</td>
<td>92.0</td>
</tr>
<tr>
<td>AmoebaNet-C [4]</td>
<td>6.4M</td>
<td>570M</td>
<td>75.7</td>
<td>92.4</td>
</tr>
<tr>
<td>PNASNet-5</td>
<td>5.1M</td>
<td>588M</td>
<td>74.2</td>
<td>91.9</td>
</tr>
</tbody>
</table>

### After The Search: ImageNet (Large)

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th># Mult-Add</th>
<th>Top 1</th>
<th>Top 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNeXt-101 [1]</td>
<td>83.6M</td>
<td>31.5B</td>
<td>80.9</td>
<td>95.6</td>
</tr>
<tr>
<td>Squeeze-Excite [2]</td>
<td>145.8M</td>
<td>42.3B</td>
<td>82.7</td>
<td>96.2</td>
</tr>
<tr>
<td>NASNet-A [3]</td>
<td>88.9M</td>
<td>23.8B</td>
<td>82.7</td>
<td>96.2</td>
</tr>
<tr>
<td>AmoebaNet-B [4]</td>
<td>84.0M</td>
<td>22.3B</td>
<td>82.3</td>
<td>96.1</td>
</tr>
<tr>
<td>AmoebaNet-A [4]</td>
<td>86.7M</td>
<td>23.1B</td>
<td>82.8</td>
<td>96.1</td>
</tr>
<tr>
<td>AmoebaNet-C [4]</td>
<td>155.3M</td>
<td>41.1B</td>
<td>83.1</td>
<td>96.3</td>
</tr>
<tr>
<td>PNASNet-5</td>
<td>86.1M</td>
<td>25.0B</td>
<td>82.9</td>
<td>96.2</td>
</tr>
</tbody>
</table>

Conclusion

- We propose to search neural network architectures in order of increasing complexity, while simultaneously learning a surrogate function to guide the search.

- PNASNet-5 achieves state-of-the-art level accuracies on CIFAR-10 and ImageNet, while being 5 to 8 times more efficient than leading RL and EA approaches during the search process.
Code and Model Release

- We have released PNASNet-5 trained on ImageNet
  - Both *Mobile* and *Large*
  - Both *TensorFlow* and *PyTorch*
  - SOTA on ImageNet amongst all publicly available models

https://github.com/tensorflow/models/tree/master/research/slim
https://github.com/chenxi116/PNASNet.TF
https://github.com/chenxi116/PNASNet.pytorch
Extensions

- Our PNAS algorithm has been applied on related tasks:
  - PPP-Net [1] and DPP-Net [2]: Pareto-optimal architectures
  - Auto-Meta [3]: Meta-learning

- PNAS did not address sharing among child models:
  - ENAS [4] and DARTS [5] showed its importance to speedup
  - EPNAS [6] combined ENAS and PNAS for further speedup

Thank You

Poster session 3B (Wednesday, September 12, 2:30pm - 4:00pm)
@chenxi116    https://cs.jhu.edu/~cxliu/