AutoGrow: automatic layer growing in deep convolutional networks

[1] Wei Wen, [2] Feng Yan, [1] Yiran Chen, [1] Hai Li

[1] Duke University, [2] University of Nevada - Reno

Code: https://github.com/wenwei202/autogrow

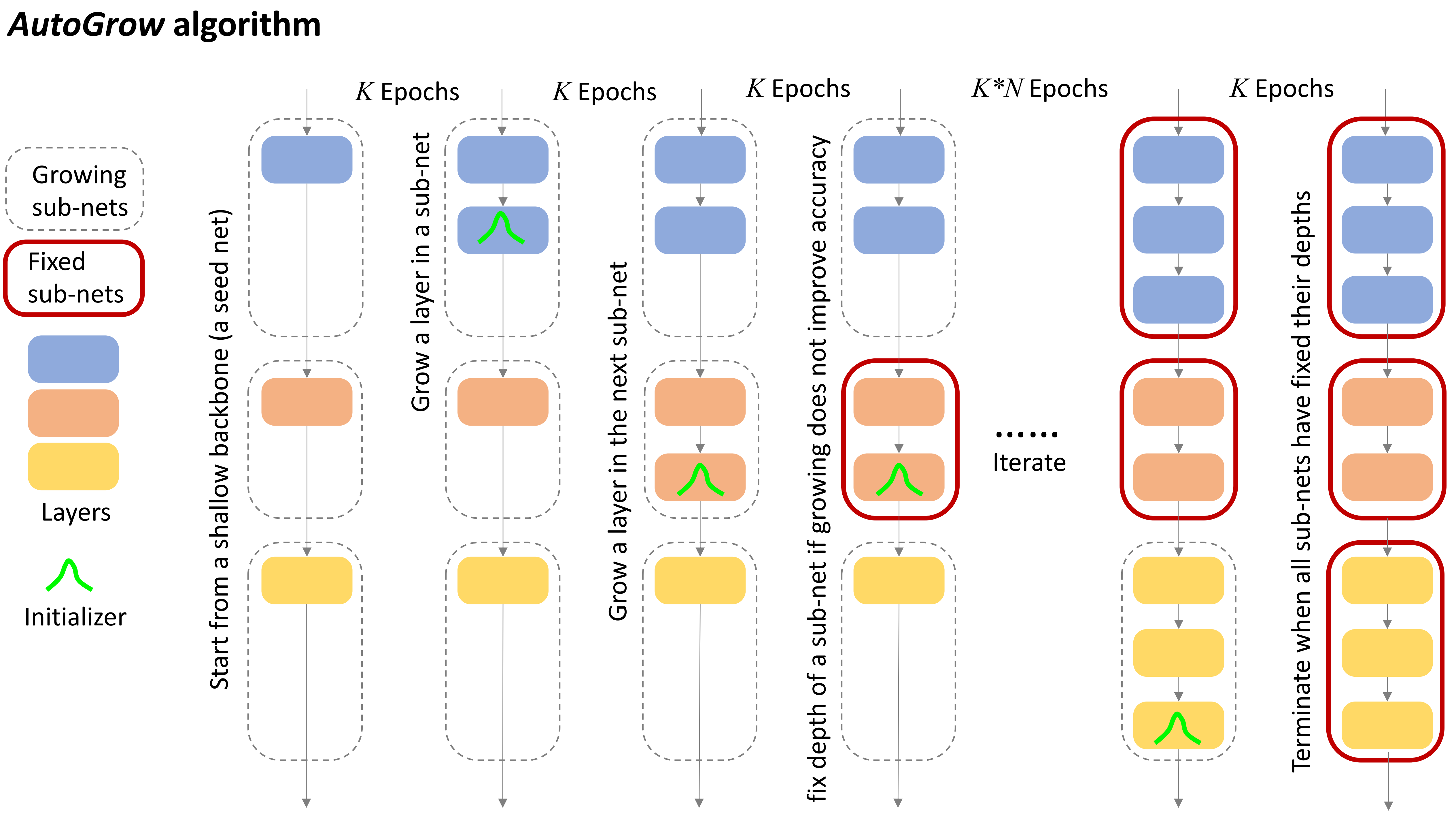
Duke UNIVERSITY

Eruditio et Religio

University of Nevada, Reno

Automated depth discovery in neural networks is essential

- Human designed and heuristic, thus time-consuming and sub-optimal
- Depth automation can save design time and find better models
- Applications
  - Automated machine learning
    - designing machine learning models by machine learning
  - Life-long learning
    - adapting neural architectures to dynamic domains



AutoGrow algorithm
Growing sub-nets
Fixed sub-nets
Layers
Initializer
K Epochs
K Epochs
K Epochs
K*N Epochs
K Epochs
Start from a shallow backbone (a seed net)
Grow a layer in a sub-net
Grow a layer in the next sub-net
fix depth of a sub-net if growing does not improve accuracy
……
Iterate
Terminate when all sub-nets have fixed their depths

Growth policy and initializers

Growth policy
Option 1: Grow after a shallow net converges
Option 2: Grow every K=3 epochs

Initializer of new layers
Option 1: Network Morphism
Option 2: Adam pre-training
Option 3: Uniform noise
Option 4: Gaussian noise

Findings contradicting previous intuitions:

- *AutoGrow* should grow rapidly ***before*** a shallow net converges using a small period of K epochs (K = 3);
- Simple random initializers ***outperform*** complex Network Morphism [3–5, 9, 36, 37]

Experimental results

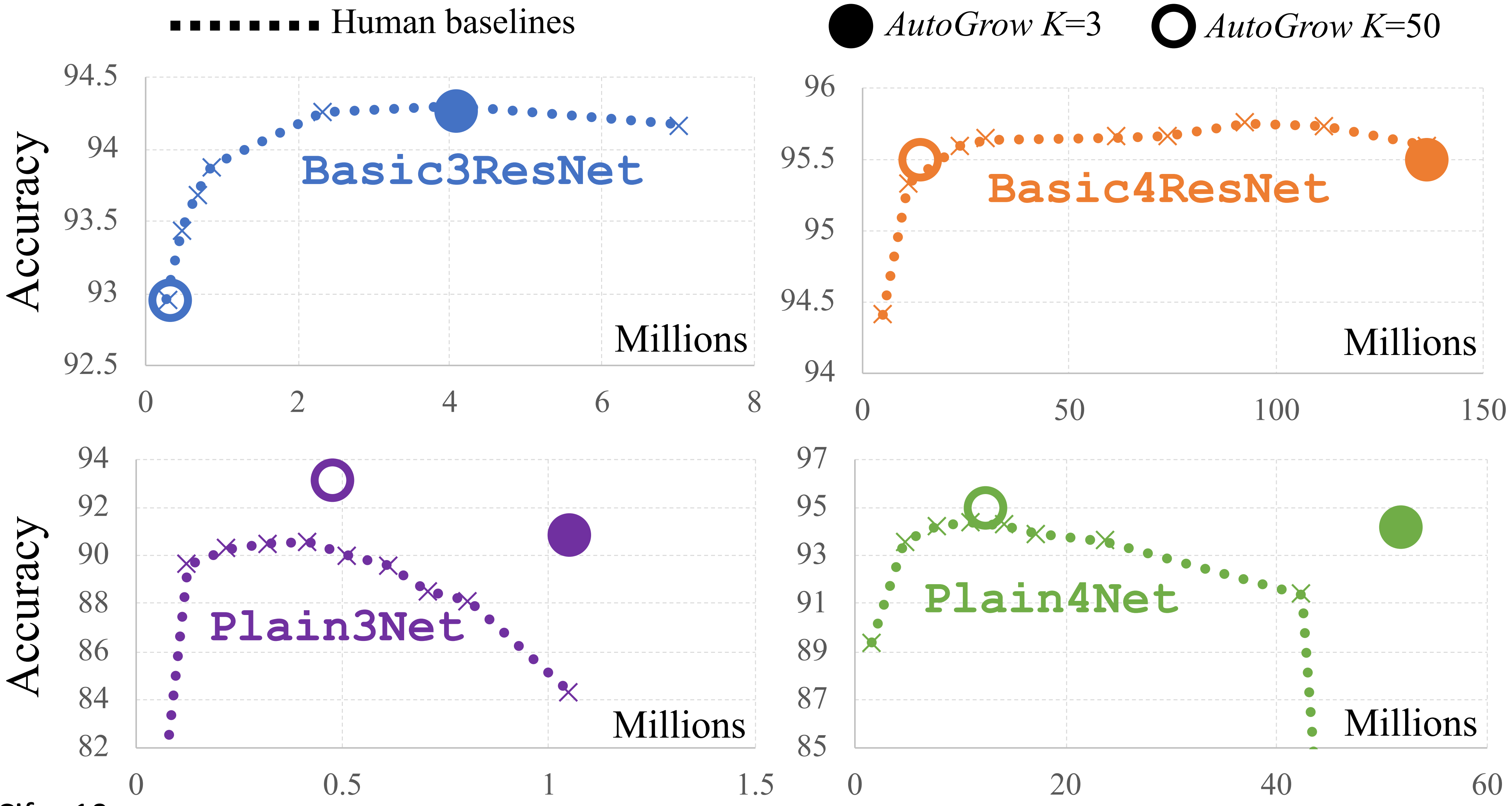
- ***AutoGrow* (with K=3) adapts to 5 datasets and 4 network architectures without any human tuning**
- **Significantly improves the accuracy of plain (non-residual) nets**

net	dataset	found net	accu %	Δ*
Basic3ResNet	CIFAR10	42-42-42	94.27	-0.03
	CIFAR100	54-53-53	74.72	**-0.95**
	SVHN	34-34-34	97.22	0.04
	FashionMNIST	30-29-29	94.57	-0.06
	MNIST	33-33-33	99.64	-0.03
Basic4ResNet	CIFAR10	22-22-22-22	95.49	-0.10
	CIFAR100	17-51-16-16	79.47	**1.22**
	SVHN	20-20-19-19	97.32	-0.08
	FashionMNIST	27-27-27-26	94.62	**-0.17**
	MNIST	11-10-10-10	99.66	0.01

* Δ = (accuracy of *AutoGrow*) − (accuracy of training from scratch)

net	dataset	found net	accu %	Δ*
Plain3Net	CIFAR10	23-22-22	90.82	**6.49**
	CIFAR100	28-28-27	66.34	**31.53**
	SVHN	36-35-35	96.79	**77.20**
	FashionMNIST	17-17-17	94.49	**0.56**
	MNIST	20-20-20	99.66	**0.12**
Plain4Net	CIFAR10	17-17-17-17	94.20	**5.72**
	CIFAR100	16-15-15-15	73.91	**29.34**
	SVHN	12-12-12-11	97.08	**0.32**
	FashionMNIST	13-13-13-13	94.47	**0.72**
	MNIST	13-12-12-12	99.57	0.03

Depth is near-optimal & tuning K can make trade-off



Dataset: Cifar-10

AutoGrow finds shallower nets when datasets are smaller

Table 9: The adaptability of *AutoGrow* to dataset sizes

Basic3ResNet on CIFAR10		
dataset size	found net	accu %
100%	42-42-42	94.27
75%	32-31-31	93.54
50%	17-17-17	91.34
25%	21-12-7	88.18

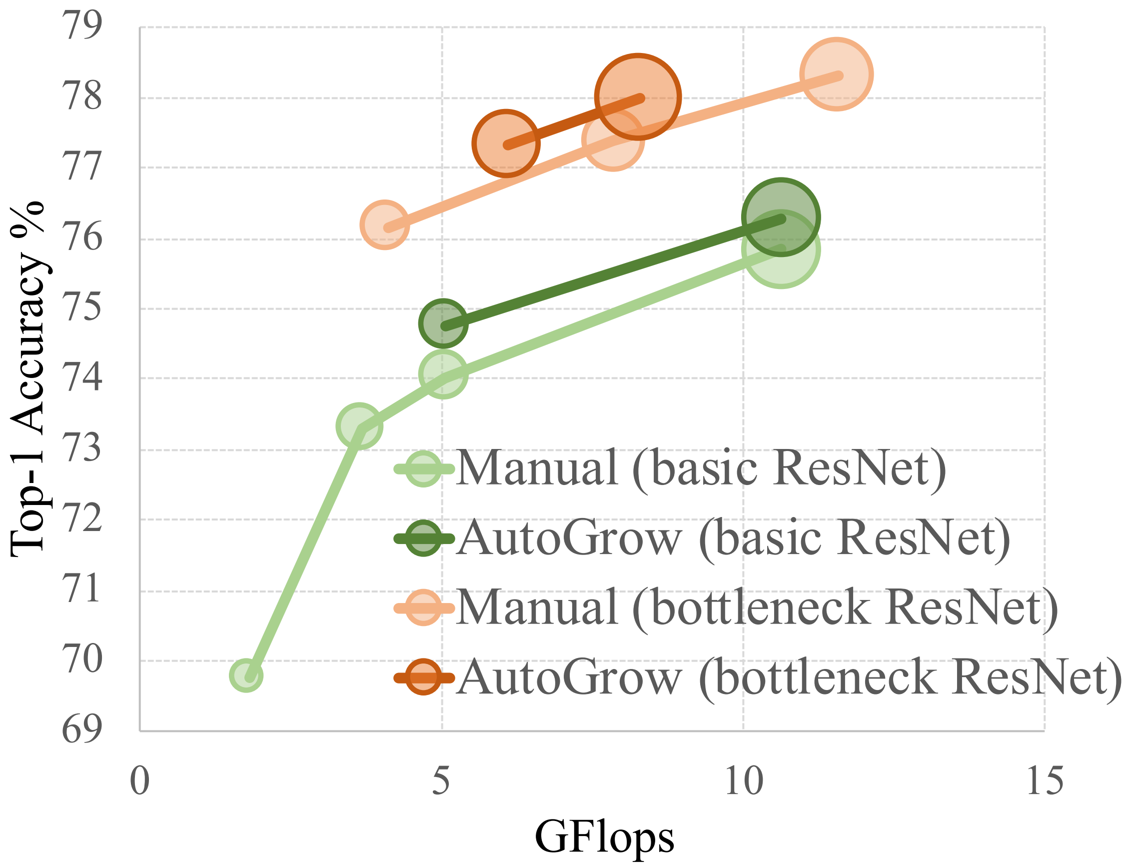
Plain3Net on MNIST		
dataset size	found net	accu %
100%	20-20-20	99.66
75%	12-12-12	99.54
50%	12-11-11	99.46
25%	10-9-9	99.33

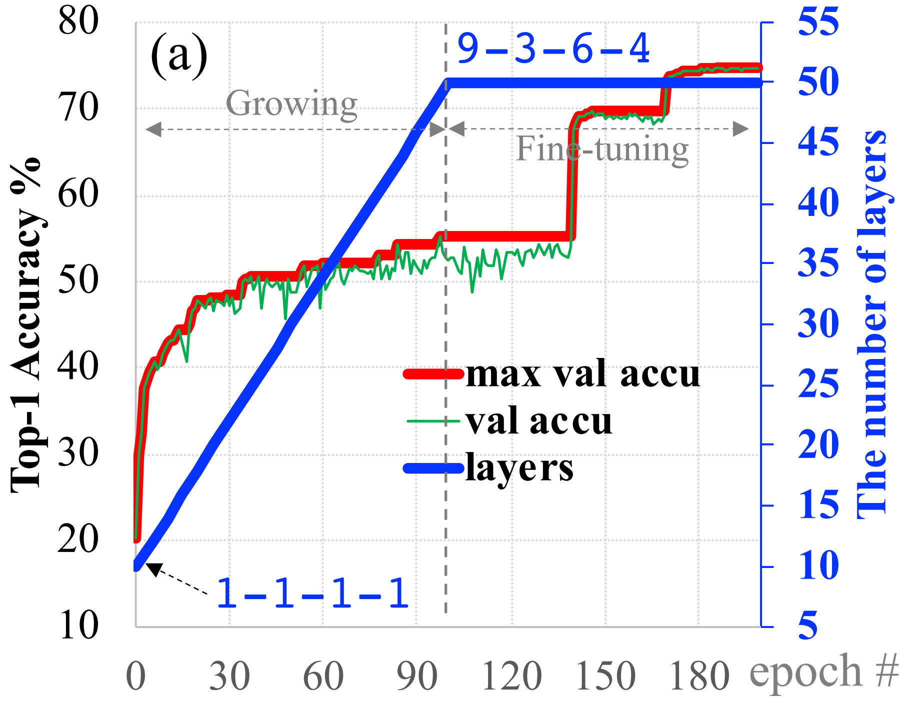
Basic4ResNet on CIFAR100		
dataset size	found net	accu %
100%	17-51-16-16	79.47
75%	17-17-16-16	77.26
50%	12-12-12-11	72.91
25%	6-6-6-6	62.53

Plain4Net on SVHN		
dataset size	found net	accu %
100%	12-12-12-11	97.08
75%	9-9-9-9	96.71
50%	8-8-8-8	96.37
25%	5-5-5-5	95.68

AutoGrow scales efficiently to large-scale problems and outperform human

Dataset: ImageNet





Takeaways

- *AutoGrow* can ***adapt*** depth of a specific network to a specific dataset without any tuning
- *AutoGrow* can ***scale*** to large dataset efficiently thanks to the rapid growth and appropriate stop policy
- New layers should be grown rapidly ***before*** shallow nets converge to avoid local minima
  - Check trajectory visualization in the paper
- Complex initializers (e.g. Network Morphism [3–5, 9, 36, 37]) are ***unnecessary*** and underperform random initializers.

Future directions

- Neural Architecture Search (NAS) with architecture growth
    - searches in an open-ended space
    - can be more efficient by starting from a small net
        - Current NAS tries to find a small sub-net by training a gigantic super-net (inefficient)
- Lifelong learning with neural growth
    - Adapts neural architectures to domains
- Integration of neural growing and neural pruning [7, 38-40]

Thanks!