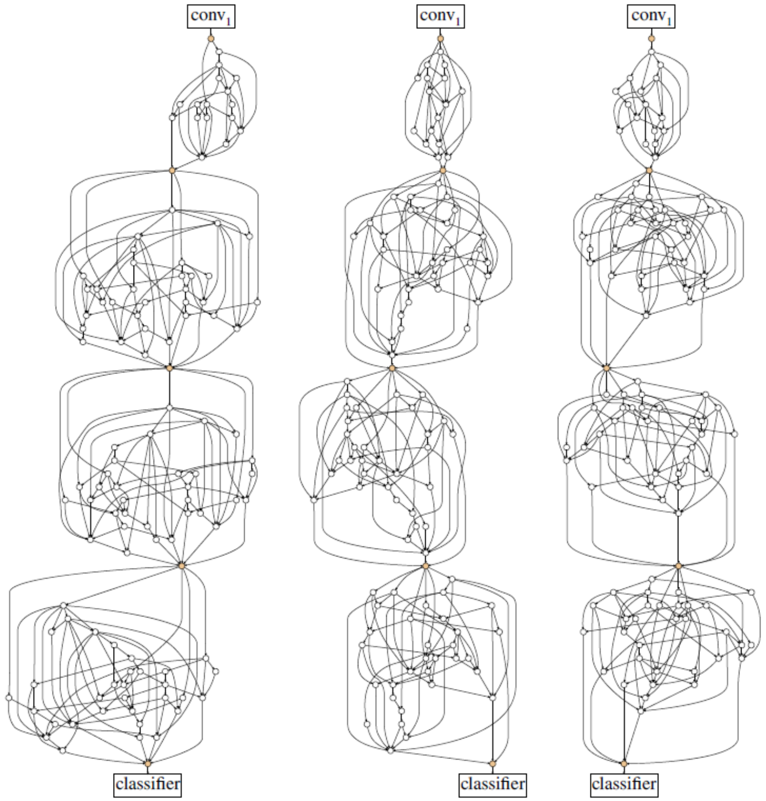
Exploring Randomly Wired Neural Networks for Image Recognition

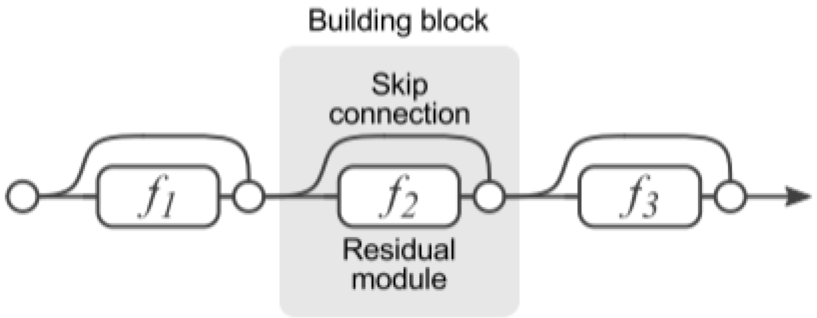


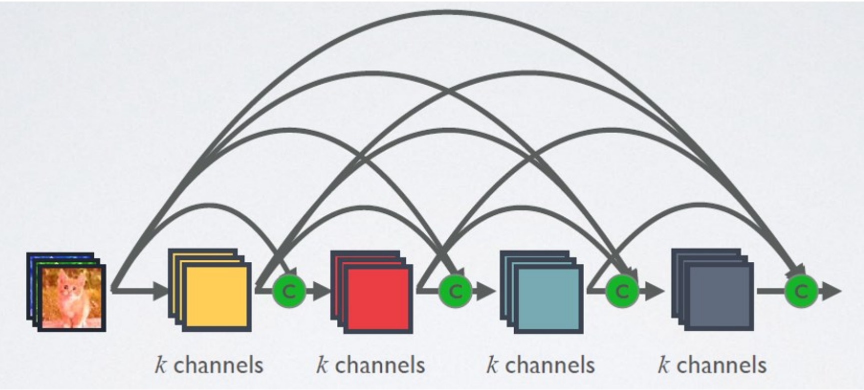
7^{th} April, 2019
PR12 Paper Review
JinWon Lee
Samsung Electronics

Saining Xie, et al., "Exploring Randomly Wired Neural Networks for Image Recognition", arxiv:1904.01569
Facebook AI Research(FAIR)

Introduction

- What we call deep learning today descends from the connectionist approach to cognitive science.
- How computational networks are wired is crucial for building intelligent machines.
- ResNet and DenseNet, that are effective in large part because of how they are wired.
- Advancing this trend, neural architecture search (NAS) has emerged as a promising direction for jointly searching wiring patterns and which operations to perform.

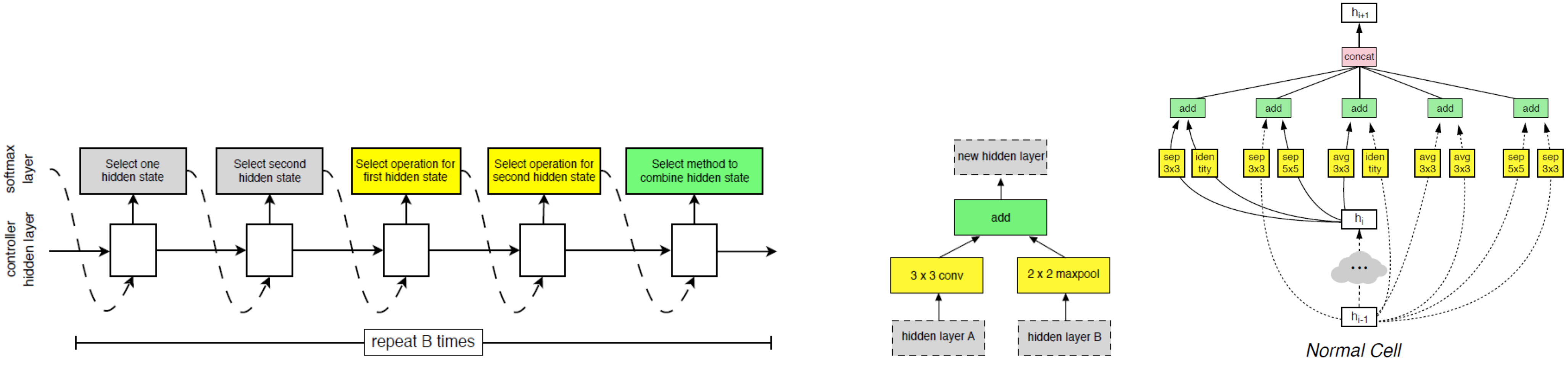




Introduction

- However, like the wiring patterns in ResNet and DenseNet, the NAS network generator is hand designed and the space of allowed wiring patterns is constrained in a small subset of all possible graphs.

- *What happens if we loosen this constraint and design novel network generators?*



<NASNet>

Main Topic

- *Design a Network Generator not an Individual Network!*

- The authors suggest a new transition from designing an individual network to designing a network generator may be possible, analogous to how our community have transitioned from designing features to designing a network that learns features.

- Rather than focusing primarily on search with a fixed generator, authors suggest designing new network generators that produce new families of models for searching.

Related Work

- Network Wiring
  - CNN & RNN use chain-like wiring patterns.
  - LSTM, Inception, ResNet, and DenseNet wiring patterns are effective in general.

- Neural Architecture Search(NAS)
  - Recent research on NAS mainly focuses on optimization methods, including RL, progressive, gradient-based, etc.
  - The search space in these NAS works is largely unchanged in these works.

Related Work

- Randomly Wired Machines
  - Turing suggested a concept of unorganized machines, which is a form of the earliest randomly connected neural networks.
  - Minsky and Rosenblatt also suggested randomly connected machines

- Relation to Neuroscience
  - Turing analogized the unorganized machines to an infant human's cortex.
  - Rosenblatt pointed out that "*at birth, the construction of the most important networks is largely random.*"

- Random Graphs in Graph Theory
  - Random graphs are widely studied in graph theory.
  - The definition of the random graph model determines the prior knowledge encoded in the resulting graphs and may connect them to naturally occurring phenomena.

Network Generator

- Defining ***a network generator*** as a mapping g from a parameter space Θ to a space of neural network architecture N.
  - The set N is typically a family of related networks, for example, VGG nets, ResNets, or DenseNets.

- The parameters $\theta \in \Theta$ specify the instantiated network and my contain diverse information.
  - For example, in a ResNet generator, θ can specify the number of stages, number of residual blocks for each stage, depth/width/filter sizes, activation types, etc.

Network Generator

- Stochastic Network Generator
  - The network generator $g(\theta)$ performs a *deterministic* mapping.
  - We can extend g to accept an additional argument s that is the *seed* of a pseudo-random number generator that is used internally by g.
  - We call generators of the form $g(\theta, s)$ *stochastic network generators*.

- NAS from the Network Generator Perspective
  - The weight matrices of the LSTM are the parameters θ of the generator.
  - Given the probability distribution conditioned on θ and the seed s, each step samples a construction action(e.g., insert an operator, connect two nodes).
  - NAS is the network generator that is hand-designed and encodes a prior from human knowledge.
  - The network space N has been restricted by hand-designed rules. (e.g., 5 nodes in a cell always have input degree 2 and output degree 1)

Randomly Wired Neural Networks

- To investigate how important the generator design is necessary to study new network generators that are substantially different from the NAS generator.

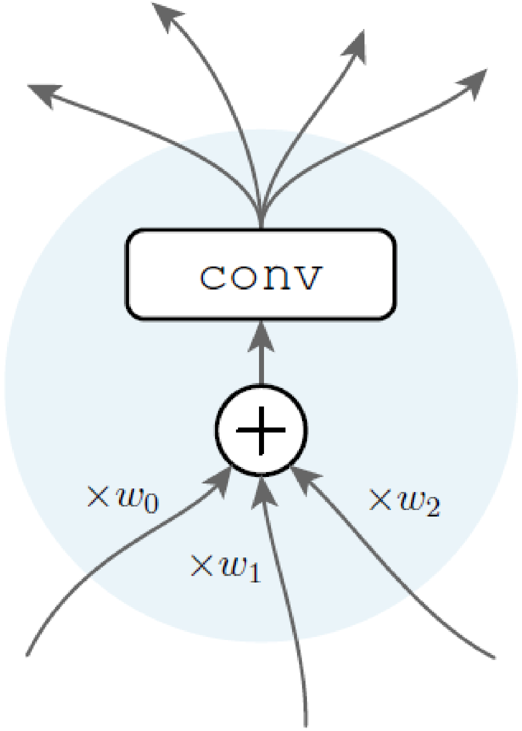
- So, authors define network generators that yield networks with random graphs, subject to different human-specific priors.
  - They use three classical random graph models – ER, BA, WS models

- Generating General Graphs
  - Starting by generating a general graph.
  - Without restricting how the graphs correspond to neural networks.
  - Once a graph is obtained, it is mapped to a computable neural networks.
  - The mapping is in itself arbitrary, and authors intentionally use a simple mapping.

Randomly Wired Neural Networks

- Edge Operation
  - The edges are data flow – a directed edge sends data from one node to another node.

- Node Operation
  - Aggregation
    - The input data to a node are combined via a weighted sum; the weights are learnable and positive.
  - Transformation
    - ReLU-convolution-BN triplet
    - The same type of convolution is used for all nodes. – 3x3 separable convolution (3x3 depthwise conv → 1x1 pointwise conv)
  - Distribution
    - The same copy of the transformed data is sent out by the output edges of the node.

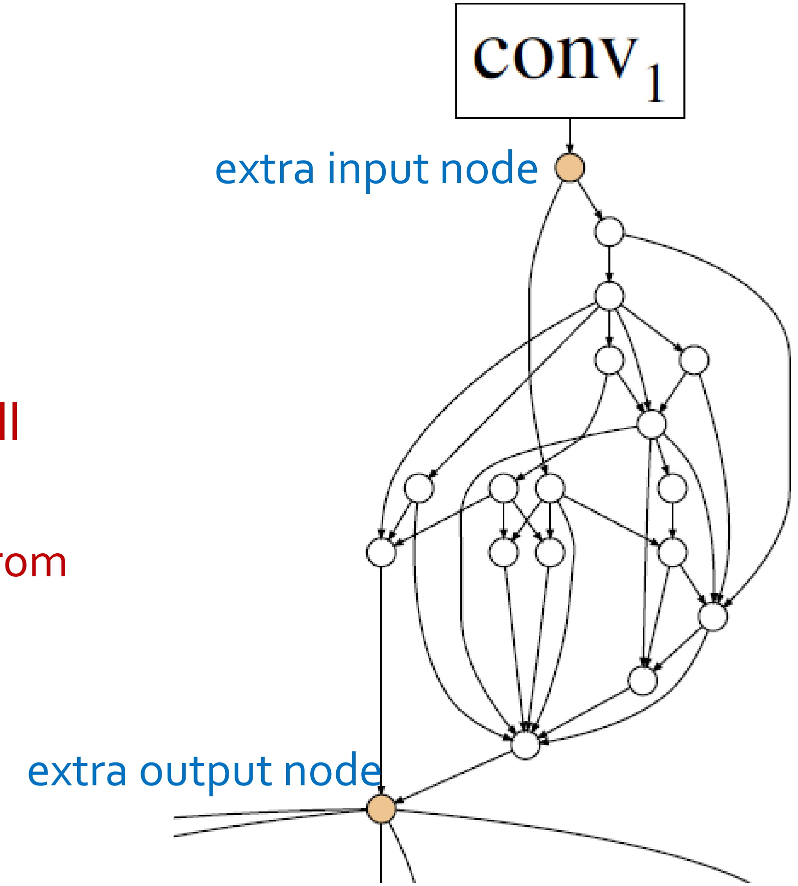


Randomly Wired Neural Networks

- Nice Properties of Node Operation
  - Additive aggregation maintains the same number of output channels as input channels.
  - Transformed data can be combined with the data from any other nodes.
  - Fixing the channel count then keeps the FLOPs and parameter count unchanged for each node, regardless of its input and output degrees.
  - Aggregation and distribution are almost parameter-free, regardless of input and output degrees.
  - Also, the overall FLOPs and parameter count of a graph are roughly proportional to the number of nodes and nearly independent of the number of edges
  - This enables the comparison of different graphs without inflating/deflating model complexity. **Differences in task performance are therefore reflective of *the properties of the wiring pattern*.**

Randomly Wired Neural Networks

- Input and Output Nodes
  - A general graph is not yet a valid neural network.
    - ➢ It may have multiple input and output nodes.
  - Creating a single extra node that is connected to all original input nodes.
    - ➢ Unique input node.
  - Similarly, creating a single extra node that is connected to all original output nodes.
    - ➢ Unique output node & computing the average(unweighted) from all original output nodes.



Randomly Wired Neural Networks

- Stages
  - In image classification in particular, it is common to divide a network into *stages* that progressively down-sample feature maps.
  - Simple strategy
    - An entire network consists of multiple stages.
    - One random graph represents one stage, and it is connected to its preceding/succeeding stage by its unique input/output node.
    - For all nodes that are directly connected to the input node, their transformations are modified to have a stride 2.
    - The channel count in a random graph is increased by 2x when going from one stage to the next stage.

<*RandWire* Architectures>

stage	output	*small regime*	*regular regime*
conv_1	112×112	3×3 conv, $C/2$	
conv_2	56×56	3×3 conv, C	***random wiring*** $N/2, C$
conv_3	28×28	***random wiring*** N, C	***random wiring*** $N, 2C$
conv_4	14×14	***random wiring*** $N, 2C$	***random wiring*** $N, 4C$
conv_5	7×7	***random wiring*** $N, 4C$	***random wiring*** $N, 8C$
classifier	1×1	1×1 conv, 1280-d global average pool, 1000-d *fc*, softmax	

Randomly Wired Neural Networks

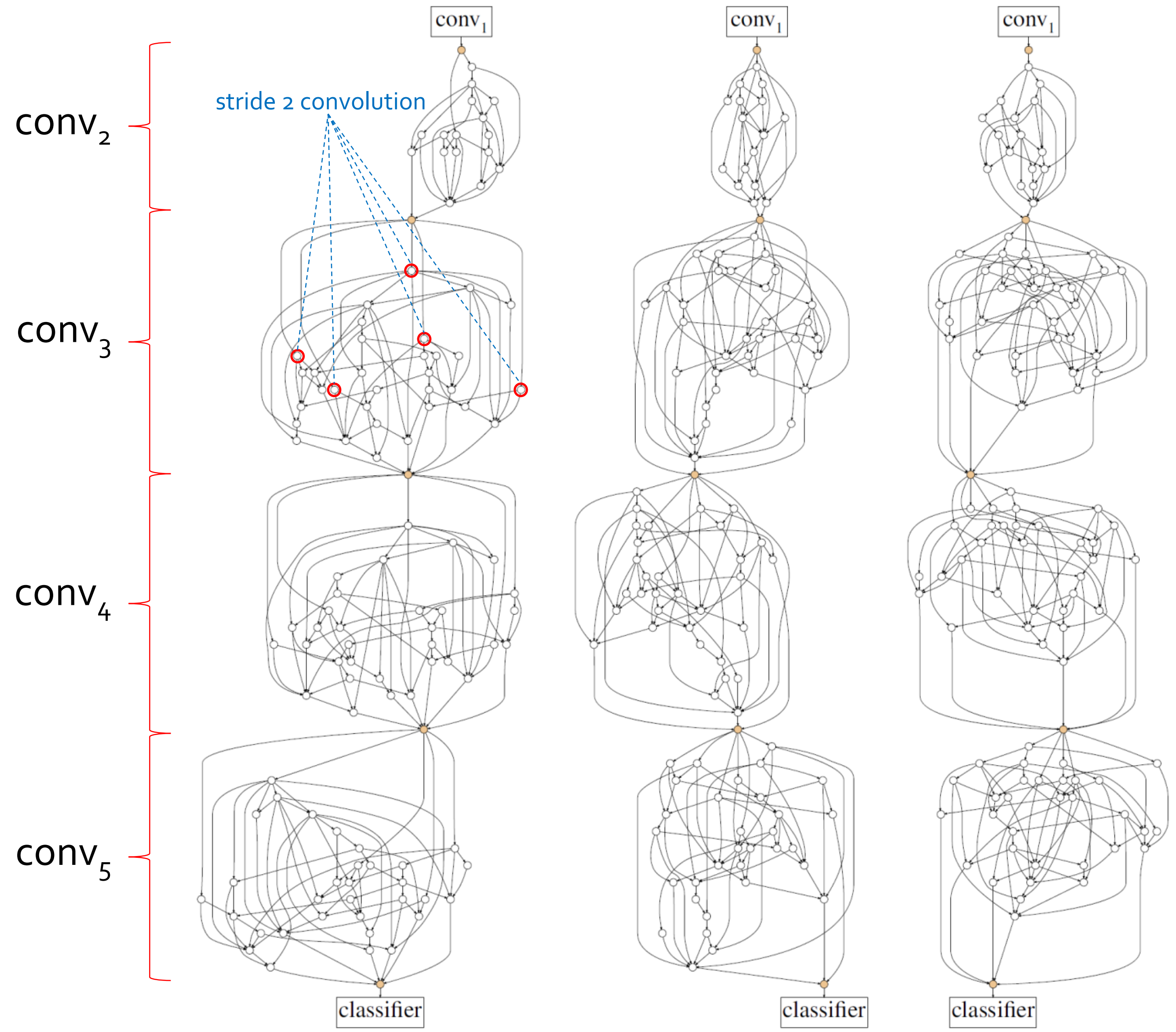
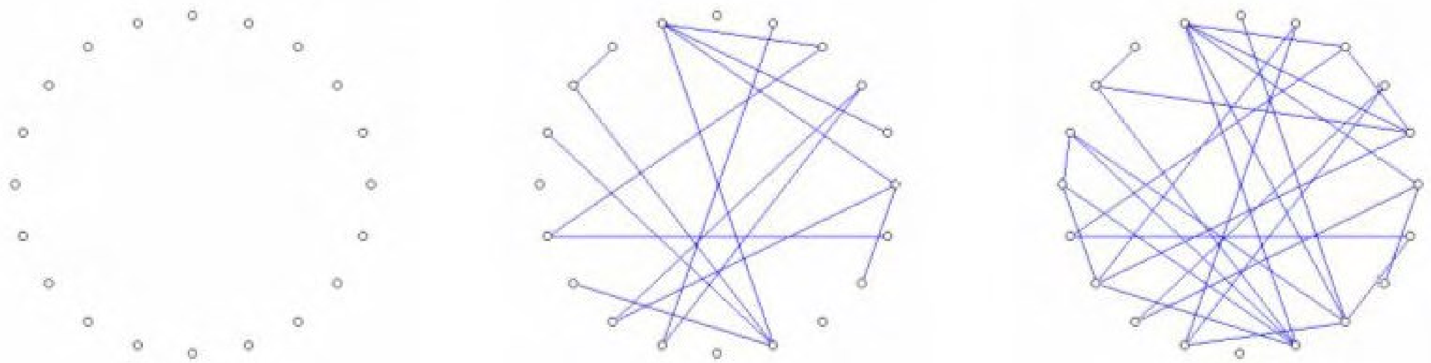


Figure 1. **Randomly wired neural networks** generated by the classical Watts-Strogatz (WS) [50] model: these three instances of random networks achieve (left-to-right) 79.1%, 79.1%, 79.0% classification accuracy on ImageNet under a similar computational budget to ResNet-50, which has 77.1% accuracy.

Random Graph Models

- Erdös-Rényi (ER), 1959.
  - Has N nodes
  - An edge between two nodes is connected with probability P.
  - The ER generation model has only a single parameter P, and is denoted as ER(P).
  - Any graph with N nodes has non-zero probability of being generated by the ER model.

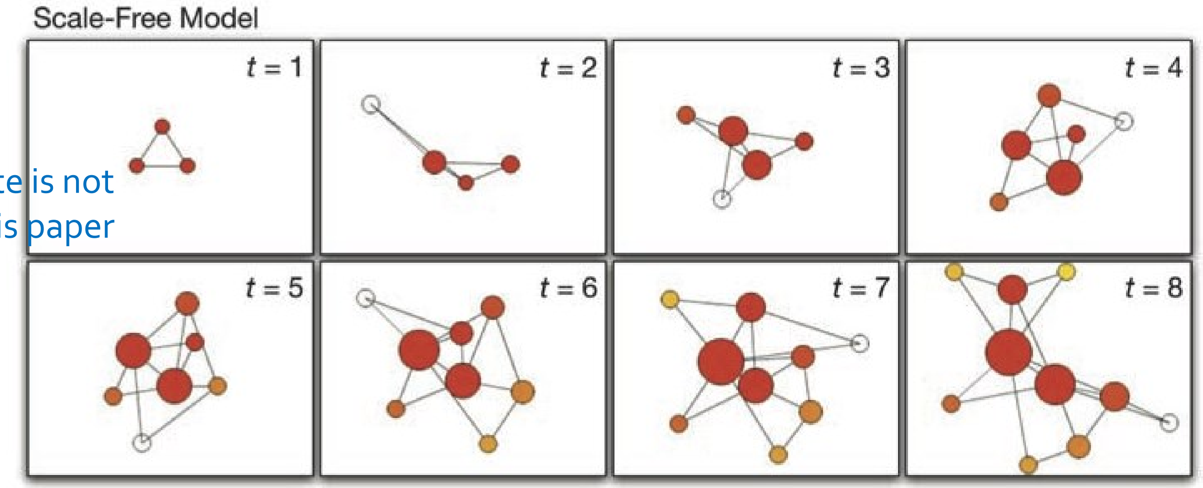


$p = 0$ (a) $p = 0.1$ (b) $p = 0.2$ (c)

Random Graph Models

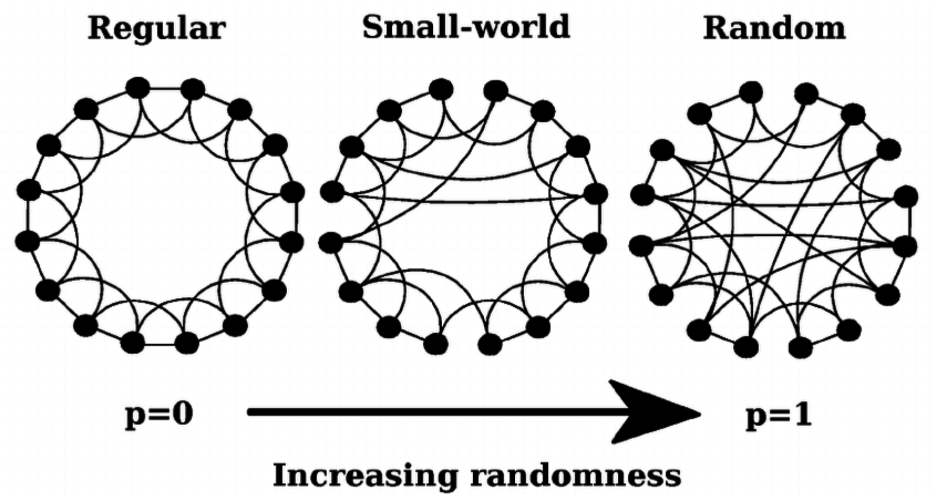
- Barabási-Albert (BA), 1999.
  - Generates a random graph by sequentially adding new nodes.
  - The initial state is M nodes without any edges ($1 \leq M < N$).
  - Sequentially adds a new node with M new edges.
  - For a node to be added, it will be connected to an existing node v with probability proportional to v's degree.
  - The new node repeatedly adds non-duplicate edges in this way until it has M edges.
  - Any graph generated by BA(M) has exactly M(N-M) edges. This gives one example on how an underlying prior can be introduced by the graph generator in spite of randomness.

Initial state is not the same as this paper



Random Graph Models

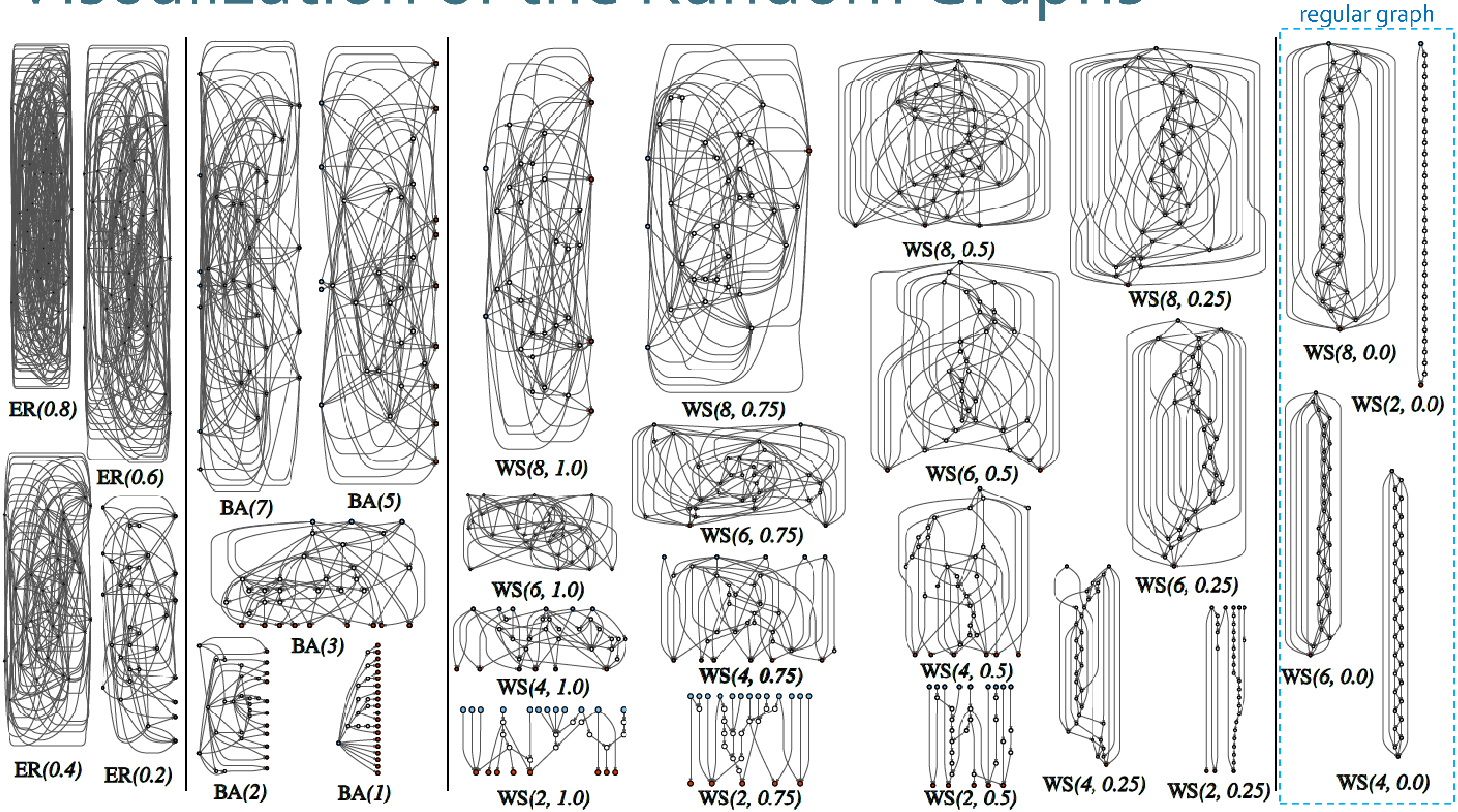
- Watts-Strogatz (WS), 1998.
  - "Small World" model.
    - Stanley Milgram experiment, Erdös number, Bacon number(http://oracleofbacon.org/).
    - High clustering, small diameter.
  - N nodes are regularly placed in a ring and each node is connected to its K=2 neighbors on both sides – starts from regular graph.
  - Then, in a clockwise loop, for every node v, the edge that connects v to its clockwise i-th next node is rewired with probability P.
  - "Rewiring" is defined as uniformly choosing a random node that is not v and that is not a duplicate edge.



Converting Undirected Graphs into DAGs

- Assign indices to all nodes in a graph and set the direction of every edge as pointing from the smaller-index node to the larger-index one. → no cycle.
- Indexing
  - ER – indices are assigned in a random order.
  - BA – the initial M nodes are assigned indices 1 to M, and all other nodes are indexed following their order of adding to the graph.
  - WS – indices are assigned sequentially in the clockwise order.

Visualization of the Random Graphs



Design and Optimization

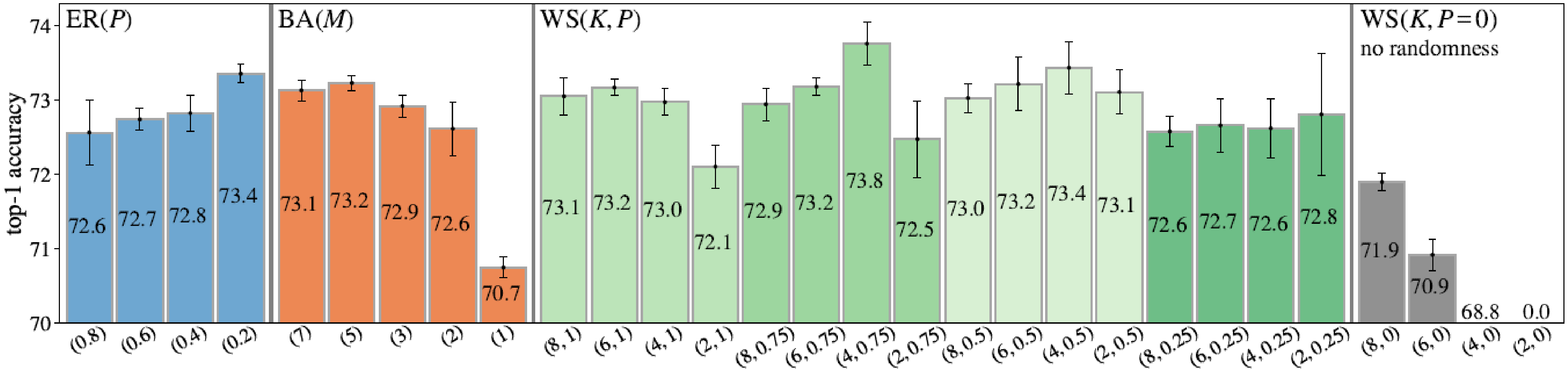
- Randomly wired neural networks are generated by a stochastic network generator $g(\theta, s)$.
- The random graph parameters, P, M, (K; P) in ER, BA, WS respectively, are part of the parameters θ.
- The "optimization" of such a 1- or 2-parameter space is essentially done by trial-and-error by human designers. – line/grid search
- The accuracy variation of our networks is small for different seeds s so they perform *no random search* and report *mean* accuracy of multiple random network instances.

Experiments

- Architecture Details
  - A small computation regime – MobileNet & ShuffleNet
  - A regular computation regime – ResNet-50/101
  - N nodes, C channels determine network complexity.
    - N = 32, C = 79 for the small regime.
    - N = 32, C = 109 or 154 for the regular regime.
- Random Seeds
  - Randomly sample 5 network instances, train them from scratch.
  - Report the classification accuracy with "mean±std" for all 5 network instances.
- Implementation Details
  - Train for 100 epochs
  - Half-period-cosine shaped learning rate decay and initial learning rate 0.1
  - The weight decay is 5e-5
  - Momentum 0.9
  - Label smoothing regularization with a coefficient of 0.1

Analysis Experiments

- All random generators provide decent accuracy over all 5 random network instances. – no fails to converge.
- ER, BA, and WS all have certain setting that yield mean accuracy > 73%, with in a <1% gap from the best accuracy from WS(4, 0.75)
- The variation among the random network instances is low. (std : 0.2~0.4%)
- Different random generators may have a gap between their mean accuracies. It means that random generator design plays an important role in the accuracy.



Analysis Experiments

- Graph Damage.
  - Graph damage by randomly removing one node or edge.
  - For networks generated by WS, the mean degradation of accuracy is larger when the output degree of the removed node is higher(hub nodes).
  - The accuracy loss is generally decreasing along the x-asis in the right figure(bottom).

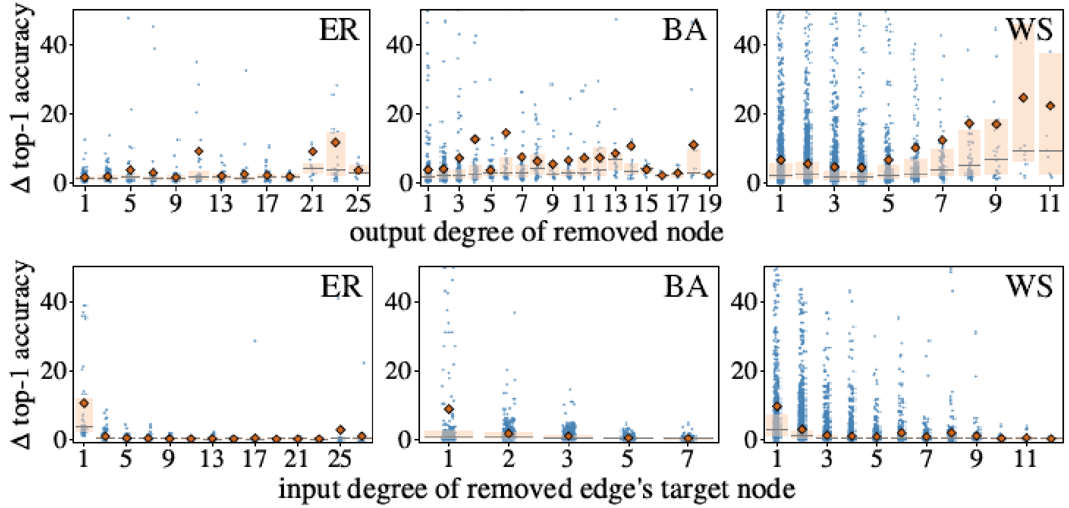
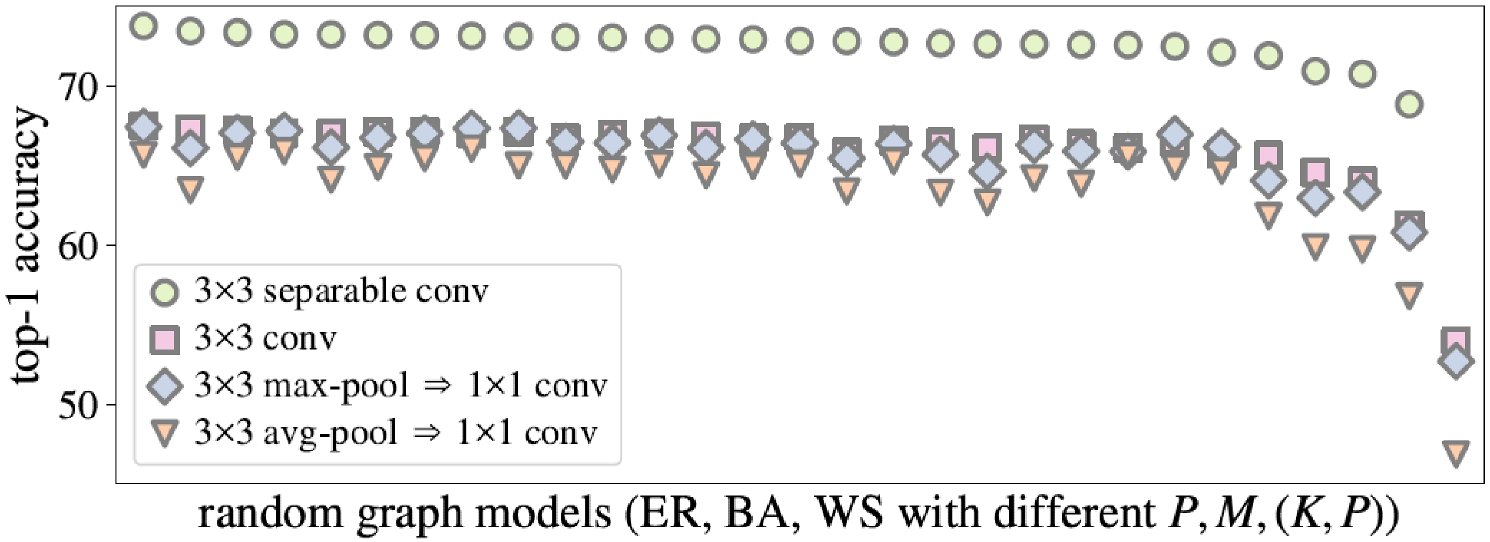


Figure 5. **Graph damage ablation**. We randomly ***remove one node*** (top) or ***remove one edge*** (bottom) from a graph after the network is trained, and evaluate the loss (Δ) in accuracy on ImageNet. From left to right are ER, BA, and WS generators. Red circle: *mean*; gray bar: *median*; orange box: *interquartile range*; blue dot: *an individual damaged instance*.

Analysis Experiments

- Node Operations
  - Almost all networks still converge to non-trivial results.
  - The Pearson correlation between any two series in the below figure is 0.91~0.98.
  - This suggests that the network wiring plays a role somewhat orthogonal to the role of the chosen operations.



Comparisons

- Small Computation Regime
  - 250 epochs for fair comparisons.
  - The mean accuracy achieved by RandWire is a competitive result, especially considering that they perform no random search, and use a single operation type for all nodes.

network	top-1 acc.	top-5 acc.	FLOPs (M)	params (M)
MobileNet [15]	70.6	89.5	569	4.2
MobileNet v2 [40]	**74.7**	-	585	6.9
ShuffleNet [54]	70.9	89.8	524	$\sim$5
ShuffleNet v2 [30]	73.7	-	524	$\sim$5
NASNet-A [56]	74.0	91.6	564	5.3
NASNet-B [56]	72.8	91.3	488	5.3
NASNet-C [56]	72.5	91.0	558	4.9
Amoeba-A [34]	74.5	92.0	555	5.1
Amoeba-B [34]	74.0	91.5	555	5.3
Amoeba-C [34]	**75.7**	**92.4**	570	6.4
PNAS [26]	74.2	91.9	588	5.1
DARTS [27]	73.1	91.0	595	4.9
RandWire-WS	$\mathbf{74.7}_{\pm 0.25}$	$\mathbf{92.2}_{\pm 0.15}$	$583_{\pm 6.2}$	$5.6_{\pm 0.1}$

Comparisons

- Regular Computation Regime
  - Use a regularization method inspired by edge removal analysis. Randomly remove one edge whose target node has input degree > 1 with probability of 0.1.
  - Mean accuracies are respectively 1.9% and 1.3% higher than the ResNet-50 and ResNet-101 and are 0.6% higher than the ResNeXt.

network	top-1 acc.	top-5 acc.	FLOPs (B)	params (M)
ResNet-50 [11]	77.1	93.5	4.1	25.6
ResNeXt-50 [52]	78.4	94.0	4.2	25.0
RandWire-WS, C=109	**79.0**$_{\pm 0.17}$	**94.4**$_{\pm 0.11}$	4.0$_{\pm 0.09}$	31.9$_{\pm 0.66}$
ResNet-101 [11]	78.8	94.4	7.8	44.6
ResNeXt-101 [52]	79.5	94.6	8.0	44.2
RandWire-WS, C=154	**80.1**$_{\pm 0.19}$	**94.8**$_{\pm 0.18}$	7.9$_{\pm 0.18}$	61.5$_{\pm 1.32}$

Comparisons

- Larger Computation
  - Same trained networks as regular computation regime, but only increase the test image size to 320x320 without retraining.
  - Mean accuracy is 0.7%~1.3% lower than the most accurate NAS results, but use only ~2/3 FLOPs and ~3/4 parameters. Networks are trained for 100 epochs and not on the target image size.
- Object Detection
  - The features learned by randomly wired networks can also transfer.

network	test size	epochs	top-1 acc.	top-5 acc.	FLOPs (B)	params (M)
NASNet-A [56]	331^2	>250	82.7	96.2	23.8	88.9
Amoeba-B [34]	331^2	>250	82.3	96.1	22.3	84.0
Amoeba-A [34]	331^2	>250	82.8	96.1	23.1	86.7
PNASNet-5 [26]	331^2	>250	82.9	96.2	25.0	86.1
RandWire-WS	320^2	100	$81.6_{\pm 0.13}$	$95.6_{\pm 0.07}$	$16.0_{\pm 0.36}$	$61.5_{\pm 1.32}$

backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50 [11]	37.1	58.8	39.7	21.9	40.8	47.6
ResNeXt-50 [52]	38.2	60.5	41.3	23.0	41.5	48.8
RandWire-WS, C=109	**39.9**	**61.9**	**43.3**	**23.6**	**43.5**	**52.7**
ResNet-101 [11]	39.8	61.7	43.3	23.7	43.9	51.7
ResNeXt-101 [52]	40.7	62.9	44.5	24.4	44.8	52.7
RandWire-WS, C=154	**41.1**	**63.1**	**44.6**	**24.6**	**45.1**	**53.0**

COCO object detection

Conclusion

- Exploring randomly wired neural networks by three classical random graph models from graph theory.
- The result were surprising: the mean accuracy of these models is competitive with hand-designed and optimized from NAS.
- The authors hope that future work exploring new generator designs may yield new, powerful networks designs.