CNN Architectures

EE807: Recent Advances in Deep Learning

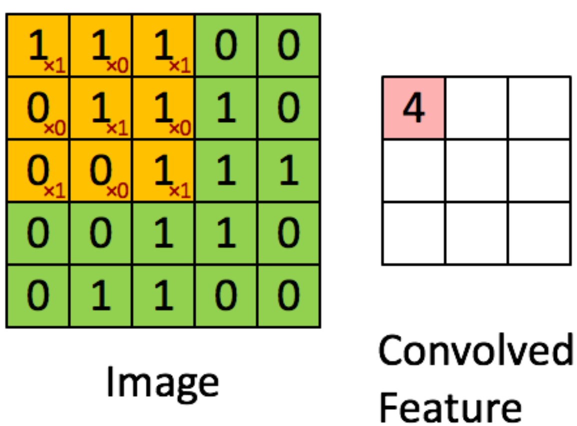
Lecture 4

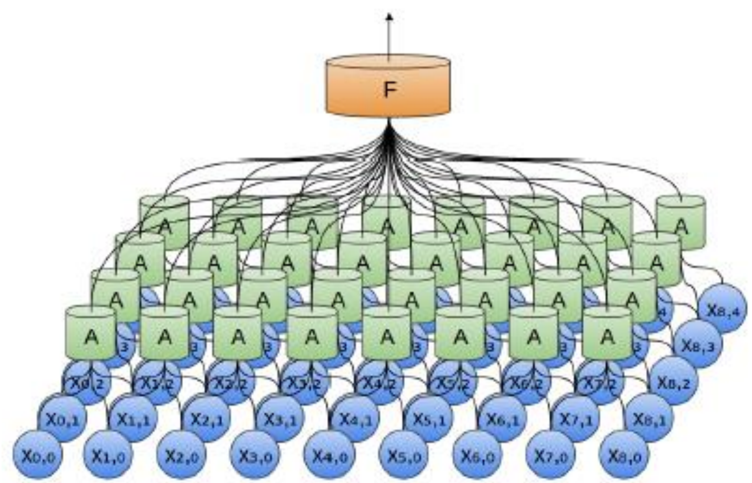
Slide made by

Jongheon Jeong and Hyungwon Choi

KAIST EE

- Neural networks that use convolution in place of general matrix multiplication
  - Sharing parameters across **multiple image locations**
  - Translation equivariant (invariant with **pooling**) operation
- Specialized for processing data that has a known, grid-like topology
  - e.g. time-series data (1D grid), image data (2D grid)

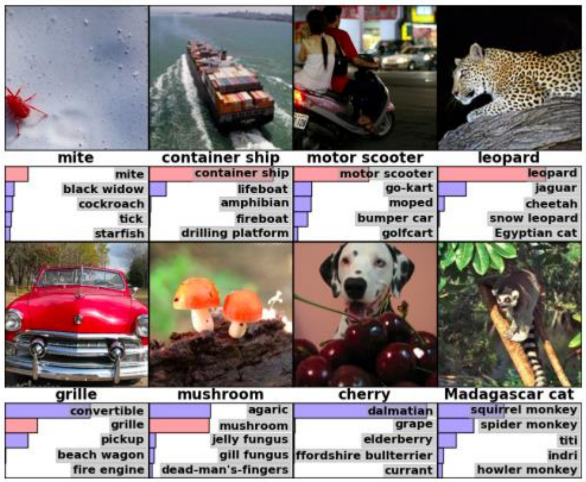


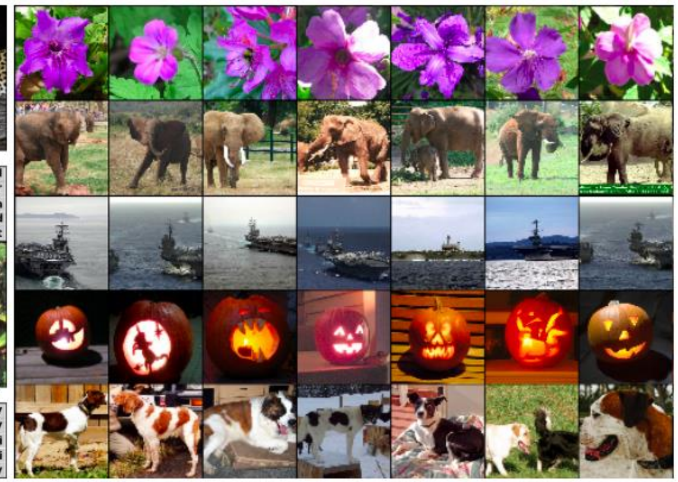


*sources :
- https://www.cc.gatech.edu/~san37/post/dlhc-cnn/
- http://colah.github.io/posts/2014-07-Conv-Nets-Modular/

- CNNs have been tremendously successful in practical applications

Classification and retrieval [Krizhevsky et al., 2012]





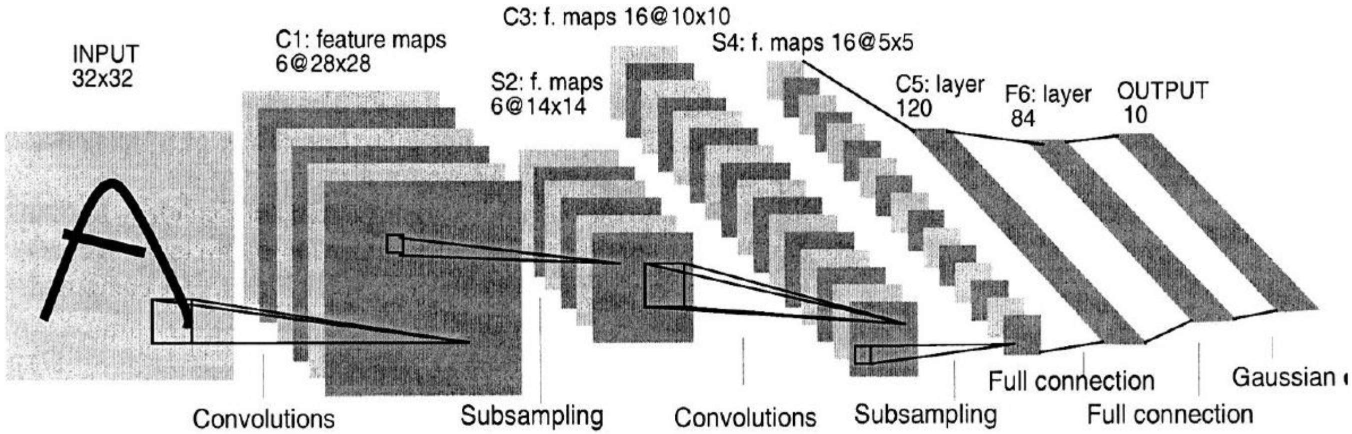
Detection [Ren et al., 2015]

Segmentation [Farabet et al., 2013]

Why do we develop CNN architectures?

- Typically, **designing a CNN model** requires some effort
  - There are a lot of design choices: # layers, # filters, sizes of kernel, pooling, …
  - It is costly to measure the performance of each model and choose the best one

- Example: **LeNet** for handwritten digits recognition [LeCun et al., 1998]

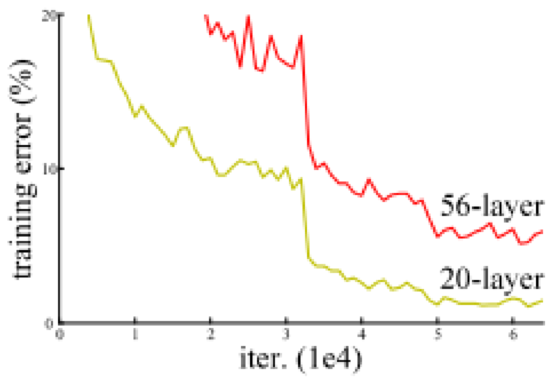


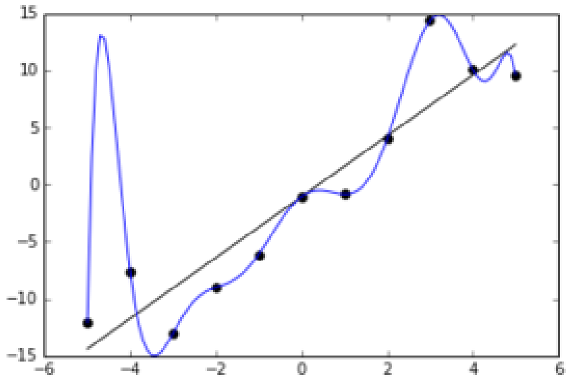
- However, LeNet is not enough to solve real-world problems in AI domain
  - CNNs are typically applied to extremely complicated domains, e.g. raw RGB images
  - We need to design **a larger model** to solve them adequately

Why do we develop CNN architectures?

- **Problem**: The larger the network, the more difficult it is to design
  1. **Optimization difficulty**
     - When the training loss is degraded
     - Deeper networks are typically much harder to optimize
     - Related to gradient vanishing and exploding
  2. **Generalization difficulty**
     - The training is done well, but the testing error is degraded
     - Larger networks are more likely to over-fit, i.e., regularization is necessary

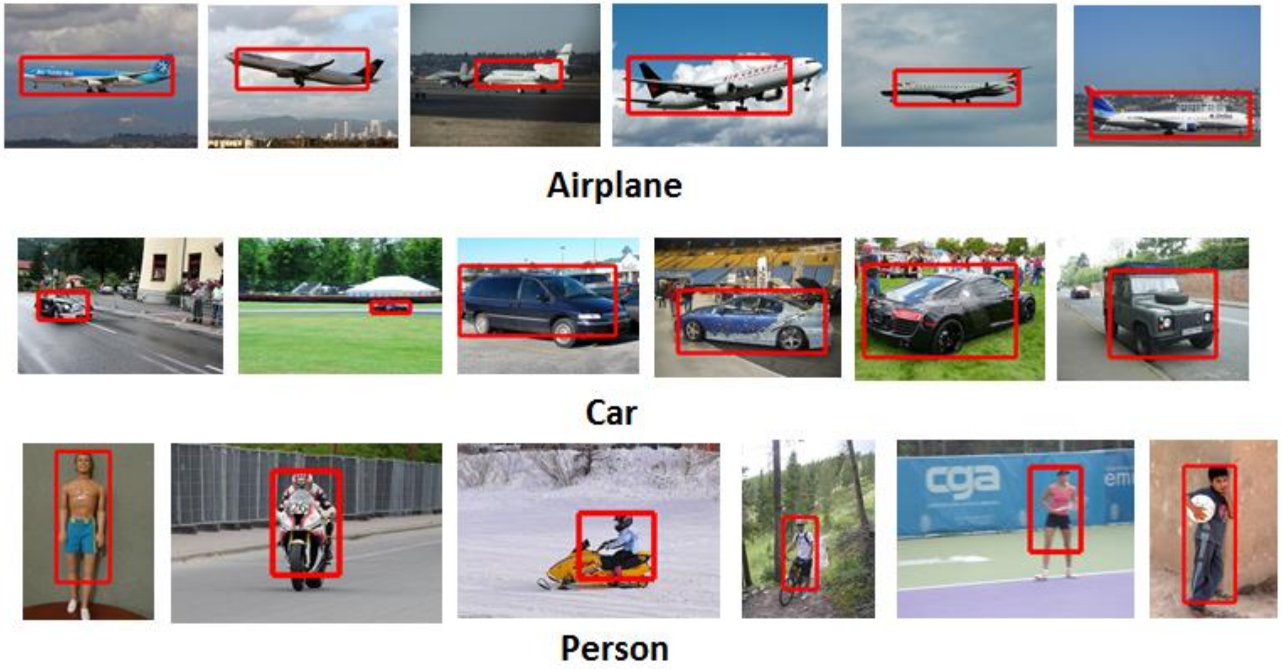
- Good architectures should be **scalable** that solves both of these problems





*sources :
- He et al. “Deep residual learning for image recognition”. CVPR 2016.
- https://upload.wikimedia.org/wikipedia/commons/thumb/6/68/Overfitted_Data.png/300px-Overfitted_Data.png

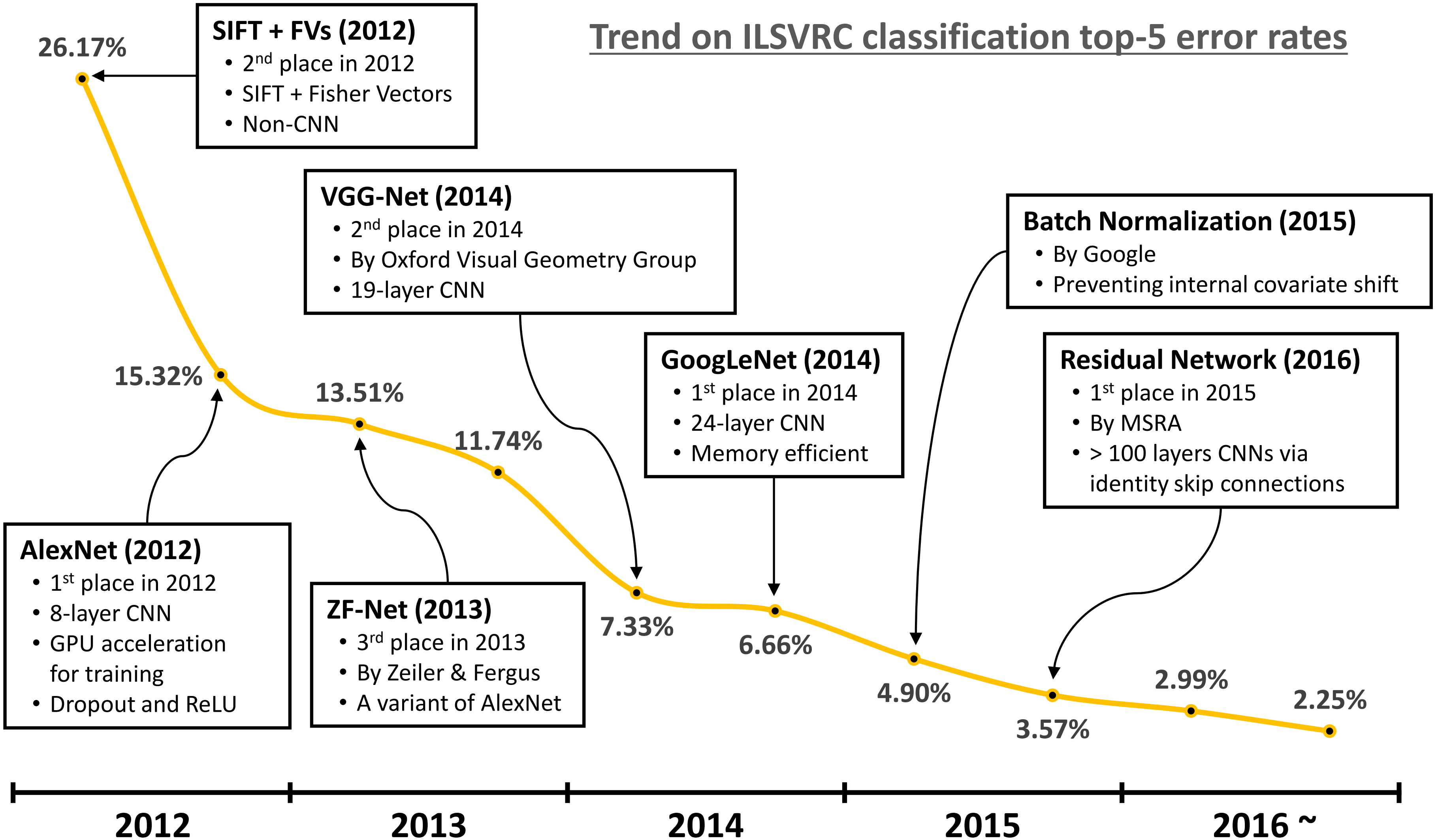
- **ImageNet Large Scale Visual Recognition Challenge** (ILSVRC)
  - ImageNet dataset: a large database of visual objects
    - ~14M labeled images, 20K classes
    - Human labels via Amazon MTurk
  - Classification: 1,281,167 images for training / 1,000 categories
  - Annually ran from 2010 to 2017, and now hosted by Kaggle
  - For details, see [Russakovsky et al., 2015]



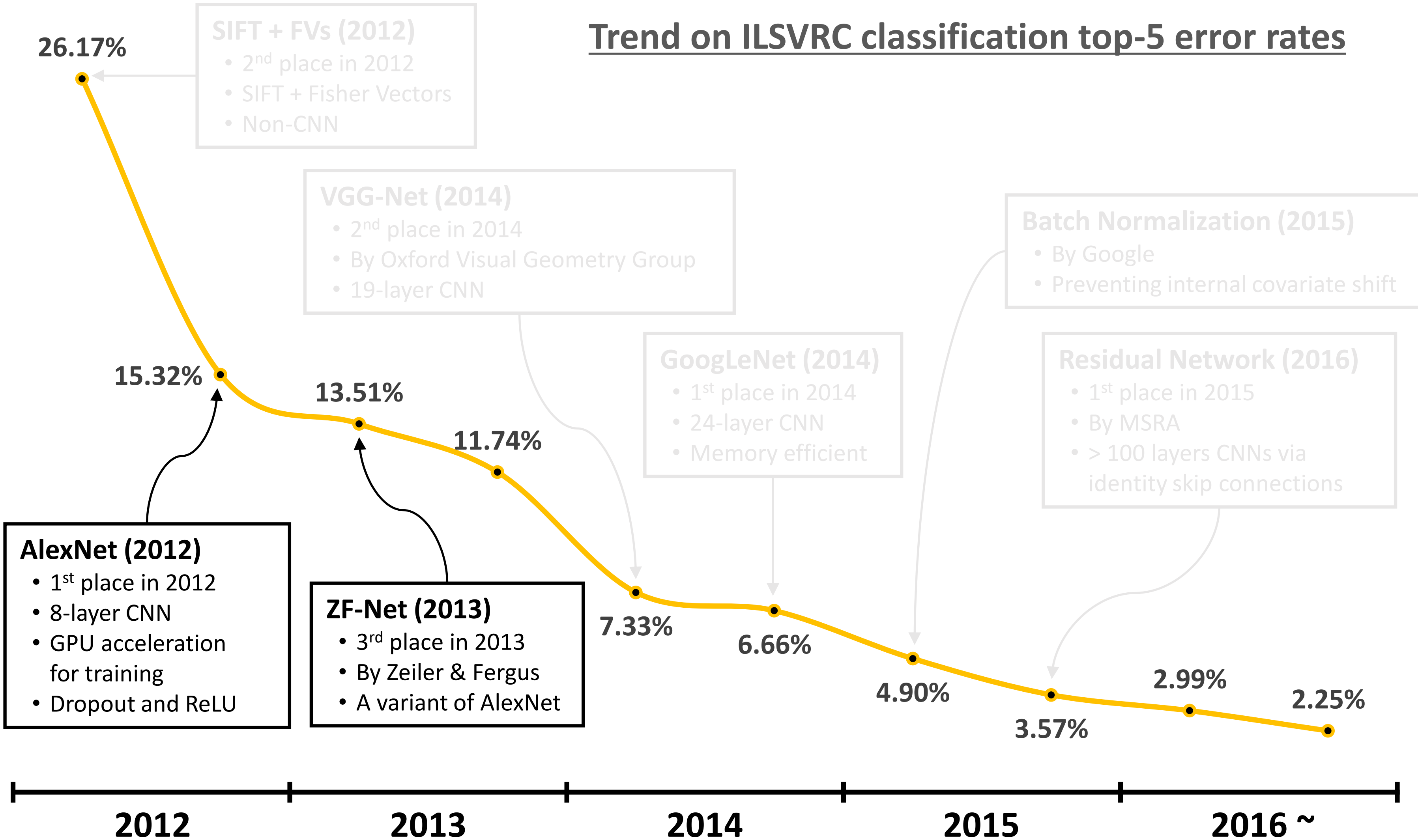
*source : http://visgraph.cse.ust.hk/ilsvrc/files/tesor.png

Evolution of CNN architectures

- ILSVRC contributed greatly to development of CNN architectures

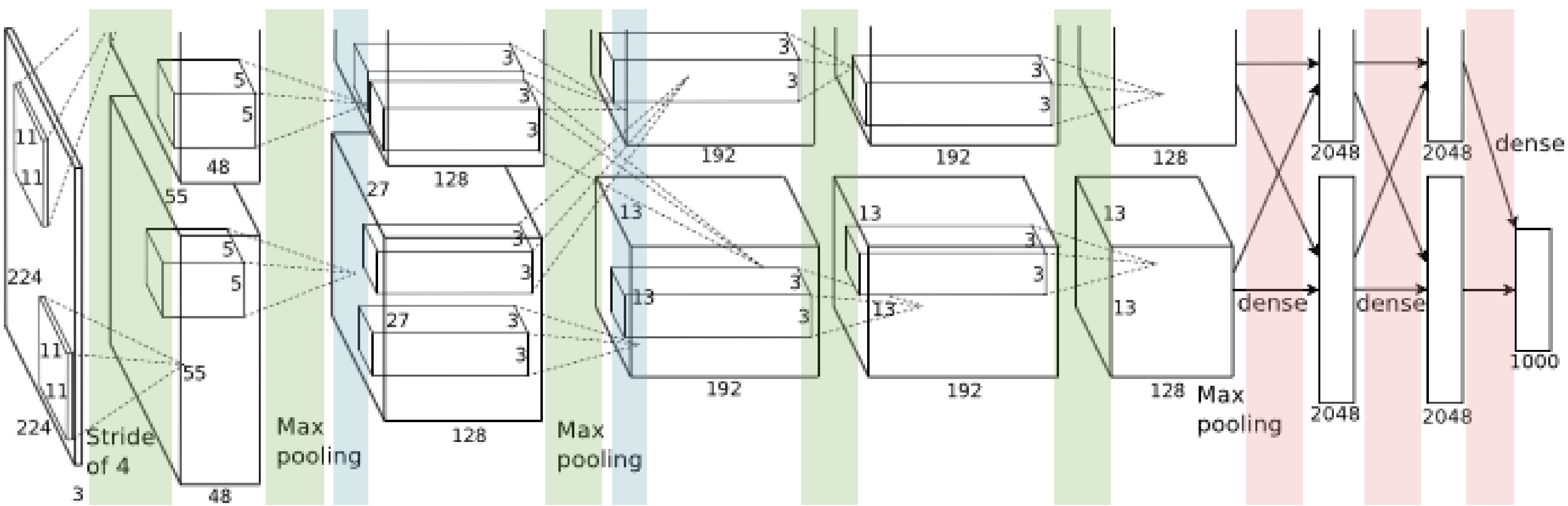


- ILSVRC contributed greatly to development of CNN architectures



Evolution of CNN architectures: AlexNet [Krizhevsky et al., 2012]

- **The first winner to use CNN** in ILSVRC, with an astounding improvement
  - Top-5 error is largely improved: 25.8% → **15.3%**
  - The 2^{nd} best entry at that time was **26.2%**
- 8-layer CNN (5 Conv + 3 FC)
- Utilized 2 GPUs (GTX-580 × 2) for training the network
  - Split a single network into 2 parts to distribute them into each GPU

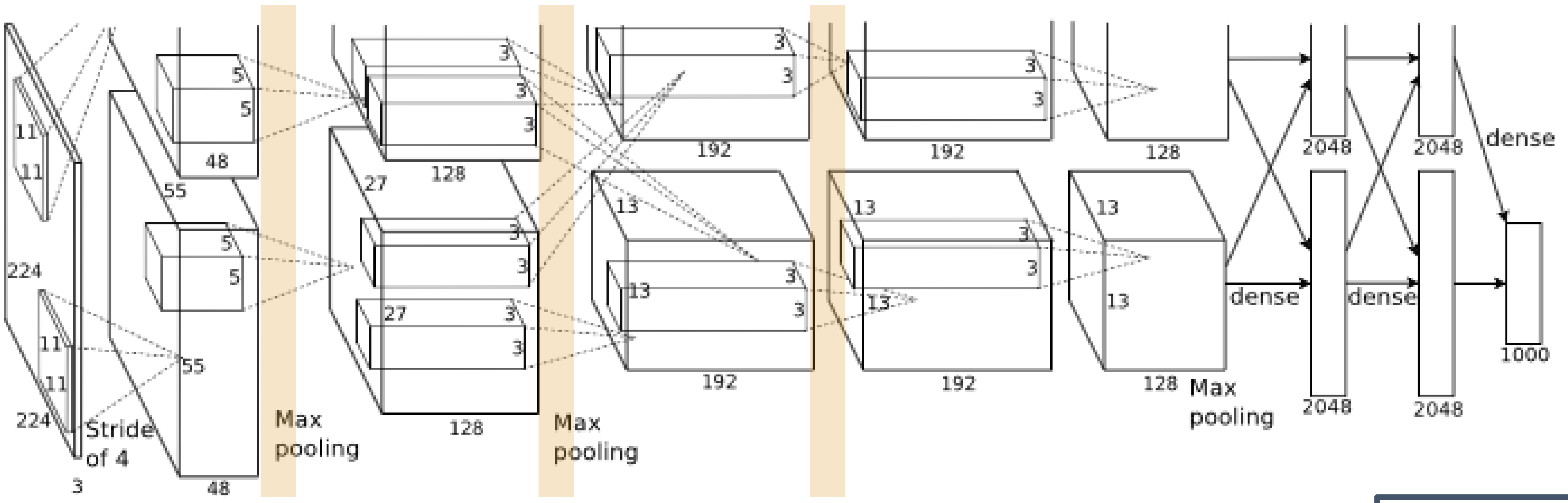


Convolutional layer Max pooling Fully-connected layers

*source : Krizhevsky et al. "Imagenet classification with deep convolutional neural networks". NIPS 2012

- **Local response normalization layers** (LRN)
  - Detects high-frequency features with a big neuron response
  - Dampens responses that are uniformly large in a local neighborhood
- Useful when using neurons with **unbounded** activations (e.g. ReLU)

$$b^i_{x,y} = a^i_{x,y} \Big/ \left(k + \alpha \sum_{j=\max(0, i-\frac{n}{2})}^{\min(N-1, i+\frac{n}{2})} (a^j_{x,y})^2 \right)^\beta$$

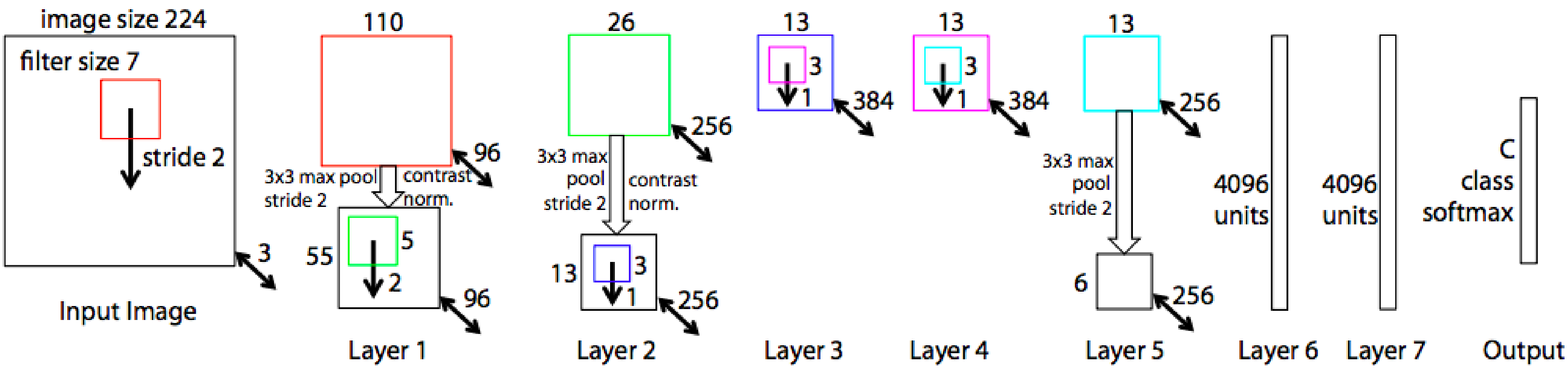


Next, ZFNet

*source : Krizhevsky et al. “Imagenet classification with deep convolutional neural networks”. NIPS 2012

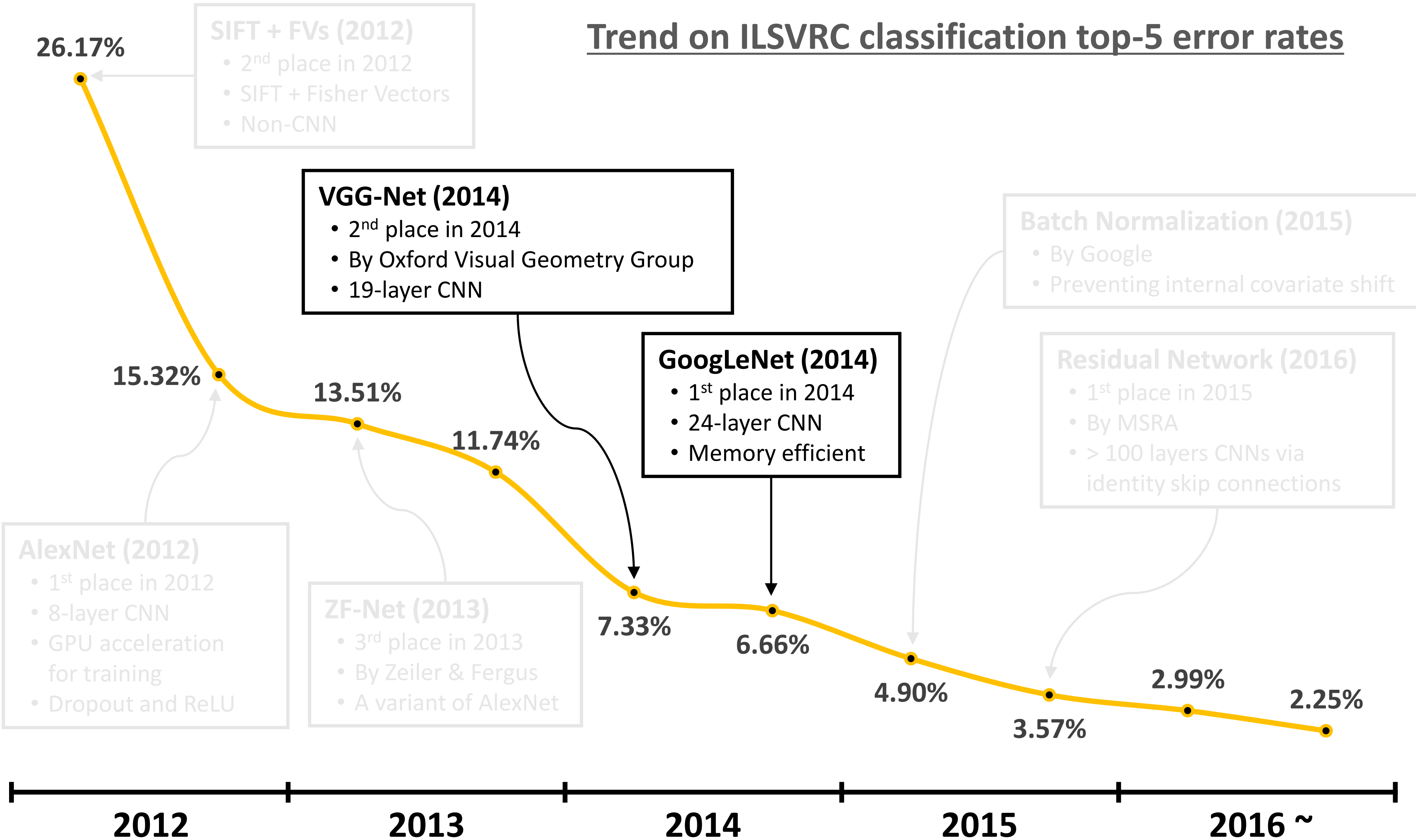
- A simple variant of AlexNet, placing the 3rd in ILSVRC'13 (15.3% → **13.5%**)
  - Smaller kernel at input: 11 × 11 → 7 × 7
  - Smaller stride at input: 4 → 2
  - The # of hidden filters are doubled

- **Lessons**:
  1. Design principle: Use smaller kernel, and smaller stride
  2. CNN architectures can be very sensitive on hyperparameters



*source : Zeiler et al., "Visualizing and understanding convolutional networks". ECCV 2014

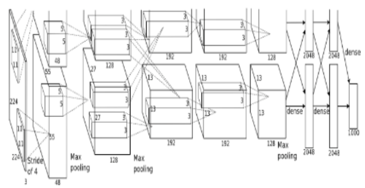
- ILSVRC contributed greatly to development of CNN architectures



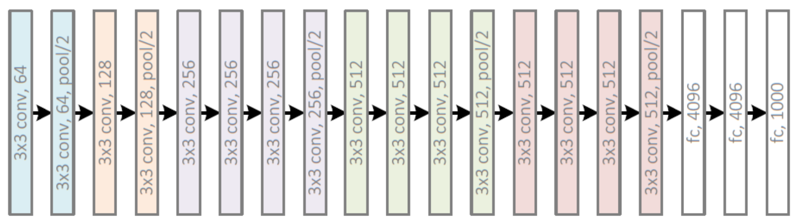
Evolution of CNN architectures: VGGNet and GoogleNet

- **Networks were getting deeper**
  - AlexNet: 8 layers
  - VGGNet: **19** layers
  - GoogleNet: **24** layers
- Both focused on parameter efficiency of each block
  - Mainly to allow larger networks computable at that time

AlexNet



VGGNet



GoogLeNet

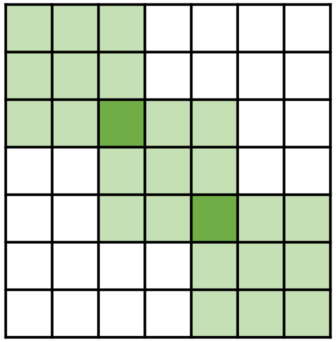
Next, VGGNet

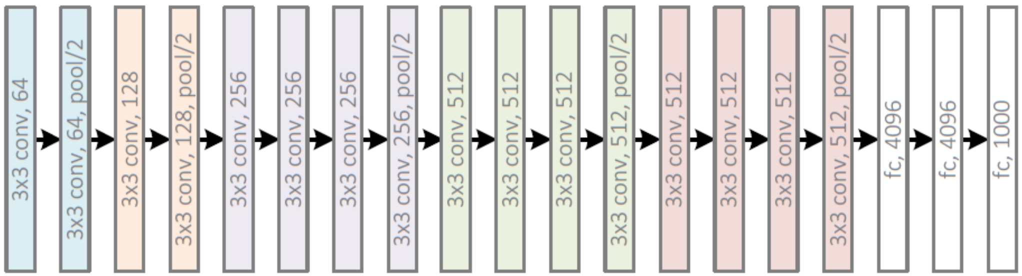
*sources :
- Krizhevsky et al. “Imagenet classification with deep convolutional neural networks”. NIPS 2012
- Simonyan et al., “Very deep convolutional networks for large-scale image recognition”. arXiv 2014.
- Szegedy et al., “Going deeper with convolutions”. CVPR 2015

Evolution of CNN architectures: VGGNet [Simonyan et al., 2014]

- The 2^{nd} place in ILSVRC'14 (11.7% → **7.33%**)
- Designed using only 3×3 kernels for convolutions

- **Lesson**: **Stacking multiple 3×3** is advantageous than using other kernels
- **Example**: $((3 \times 3) \times 3)$ v.s. (7×7)
  - Essentially, they get the same receptive field
  - $((3 \times 3) \times 3)$ have less # parameters
    - $3 \times \left(C \times ((3 \times 3) \times C)\right) = \mathbf{27C^2}$
    - $C \times ((7 \times 7) \times C) = \mathbf{49C^2}$
  - $((3 \times 3) \times 3)$ gives more non-linearities



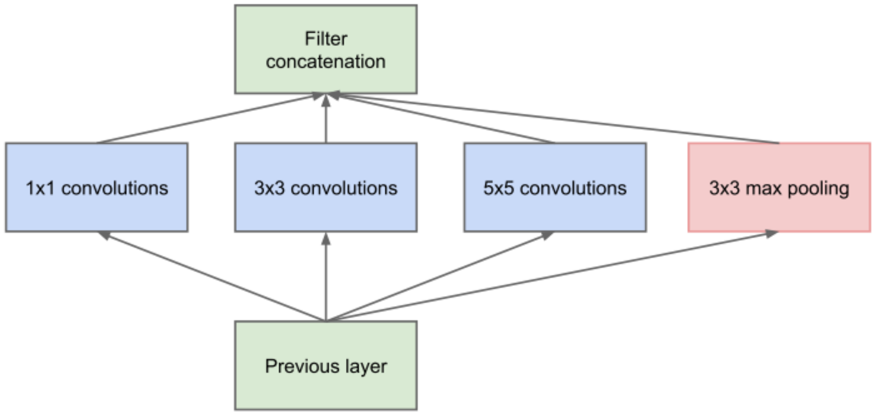


Next, GoogLeNet

*source : Simonyan et al., "Very deep convolutional networks for large-scale image recognition". arXiv 2014.

- The winner of ILSVRC'14 (11.7% → **6.66%**)
- Achieved 12× fewer parameters than AlexNet

- **Inception module**
  - Multiple operation paths with different receptive fields
  - Each of the outputs are **concatenated** in filter-wise
  - Capturing sparse patterns in a stack of features



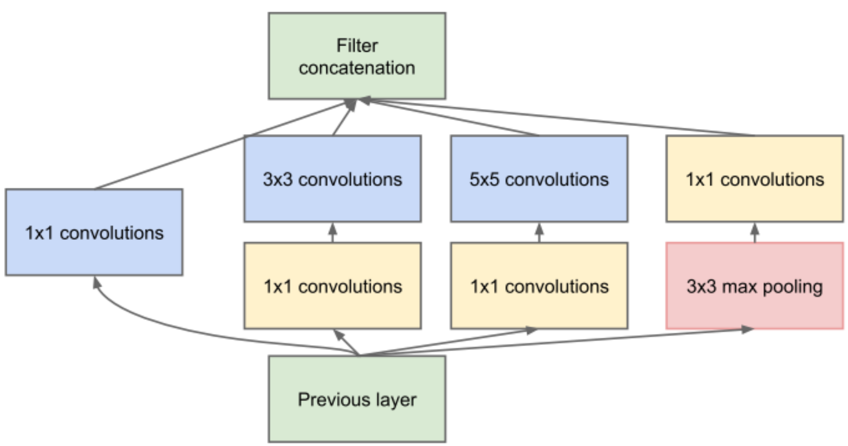
(a) Inception module, naïve version

*source : Szegedy et al., "Going deeper with convolutions". CVPR 2015

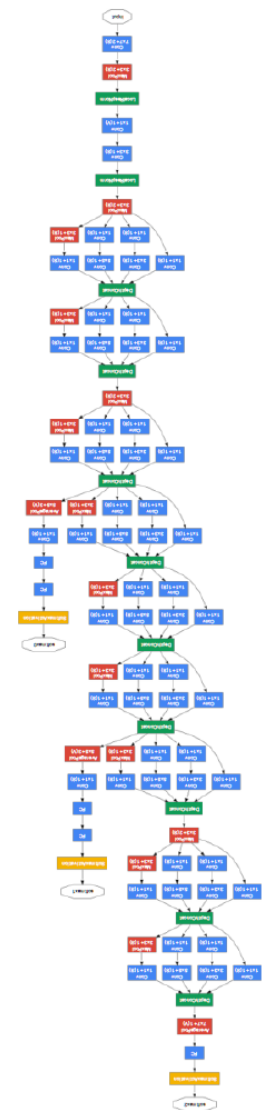
Evolution of CNN architectures: GoogleNet [Szegedy et al., 2015]

- The winner of ILSVRC'14 (11.7% → **6.66%**)
- Achieved 12× fewer parameters than AlexNet

- **Use of 1 × 1 convolutions**
  - Naïve inceptions can be too expensive to scale up
  - **Dimension reduction** before expensive convolutions
  - They also gives more non-linearities



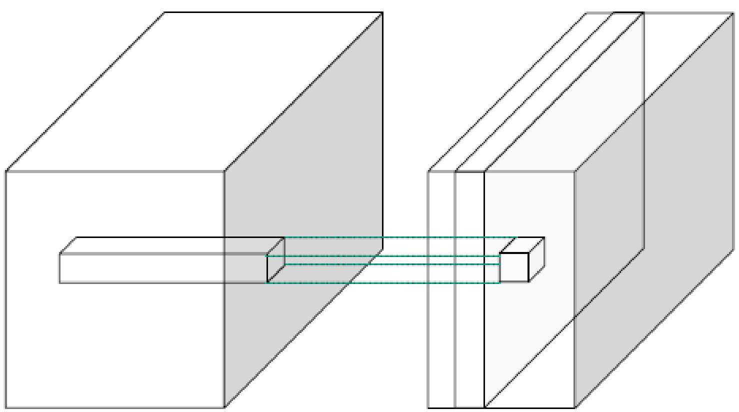
(b) Inception module with dimensionality reduction

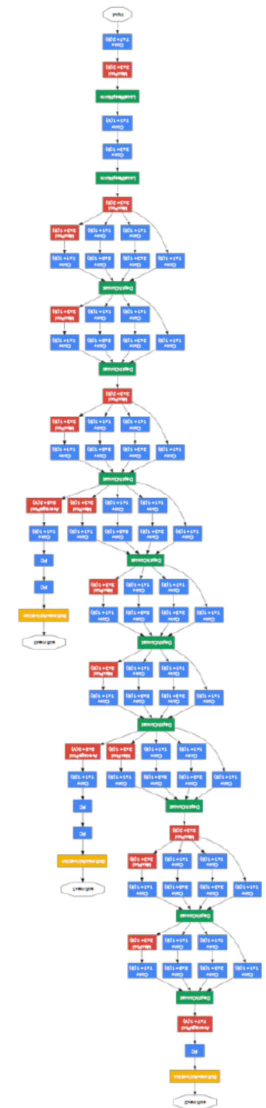


*source : Szegedy et al., "Going deeper with convolutions". CVPR 2015

- The winner of ILSVRC'14 (11.7% → **6.66%**)
- Achieved 12× fewer parameters than AlexNet

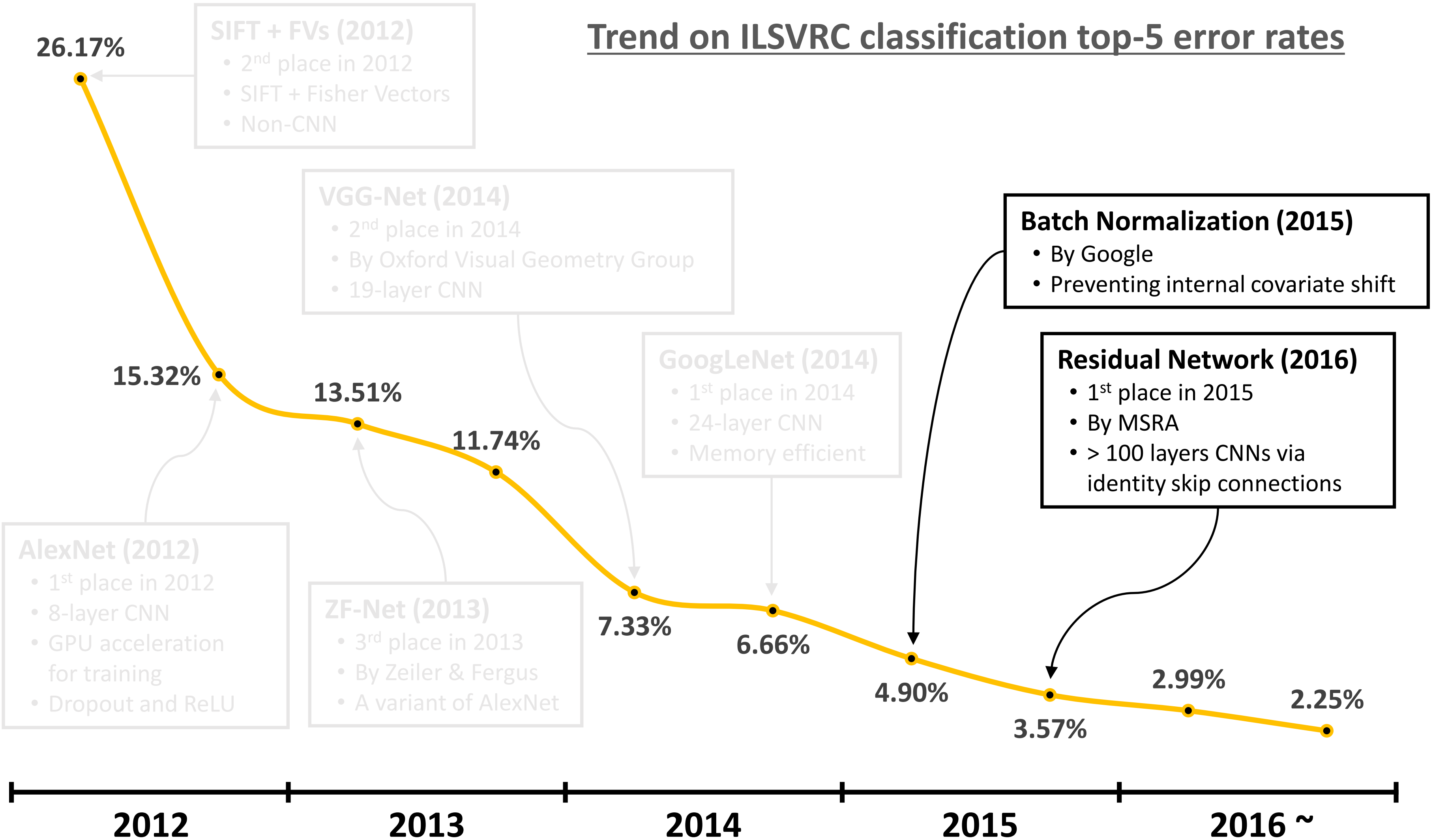
- *cf*. **1 × 1 convolutions**
  - Linear transformation done in pixel-wise
  - Can be represented by a matrix
  - Useful for changing # channels efficiently





*sources :
- Szegedy et al., "Going deeper with convolutions". CVPR 2015
- Lana Lazebnik, "Convolutional Neural Network Architectures: from LeNet to ResNet".

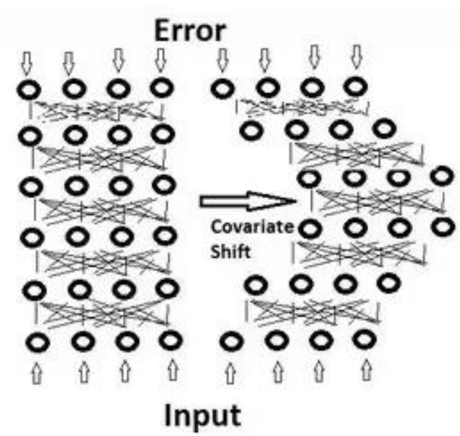
- ILSVRC contributed greatly to development of CNN architectures



Evolution of CNN architectures: Batch normalization [Ioffe et al., 2015]

- **Training a deep network well** had been a delicate task
  - It requires a careful initialization, with adequately low learning rate
  - **Gradient vanishing**: networks containing saturating non-linearity

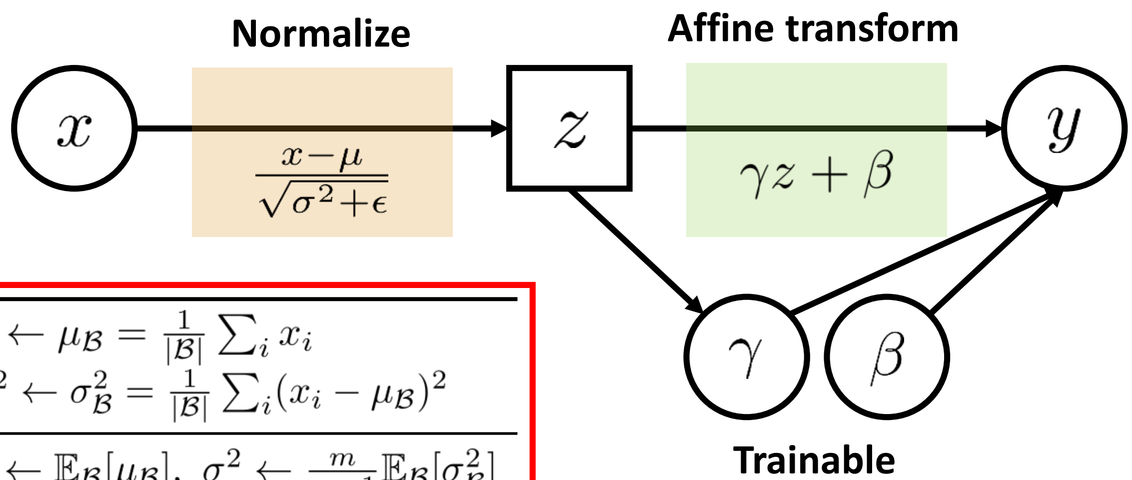
- Ioffe et al. (2015): Such difficulties are come from **internal covariate shift**
- **Motivation**: “The cup game analogy”



  - Similar problem happens during training of deep neural networks
  - Updates in early layers may shift the inputs of later layers too much

*sources :
- Ioffe et al., “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. ICML 2015
- http://pages.cs.wisc.edu/~shavlik/cs638/lectureNotes/Batch_Normalization.pptx
- https://www.quora.com/Why-does-batch-normalization-help

- **Batch normalization** (BN) accelerates neural network training by eliminating internal covariate shift inside the network
- **Idea**: A normalization layer that behaves differently in training and testing



Training	$\mu \leftarrow \mu_{\mathcal{B}} = \frac{1}{\lvert\mathcal{B}\rvert}\sum_i x_i$ $\sigma^2 \leftarrow \sigma^2_{\mathcal{B}} = \frac{1}{\lvert\mathcal{B}\rvert}\sum_i (x_i - \mu_{\mathcal{B}})^2$
Testing	$\mu \leftarrow \mathbb{E}_{\mathcal{B}}[\mu_{\mathcal{B}}],\ \sigma^2 \leftarrow \frac{m}{m-1}\mathbb{E}_{\mathcal{B}}[\sigma^2_{\mathcal{B}}]$

1. During training, input distribution of y only depends on γ and β
   - Training mini-batches are always normalized into mean 0, variance 1
2. There is some gap between $\mu_{\mathcal{B}}$ and $\mathbb{E}[\mu_{\mathcal{B}}]$ ($\sigma^2_{\mathcal{B}}$, resp.)
   - Noise injection effect for each mini-batch ⇒ Regularization effect

*source : Ioffe et al., “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. ICML 2015

- **Batch normalization** (BN) accelerates neural network training by eliminating internal covariate shift inside the network
  - BN allows much **higher learning rates**, i.e. faster training
  - BN **stabilizes** gradient vanishing on saturating non-linearities
  - BN also has its own **regularization effect**, so that it allows to reduce weight decay, and to remove dropout layers

- BN makes GoogLeNet much easier to train with great improvements

Model	Resolution	Crops	Models	Top-1 error	Top-5 error
GoogLeNet ensemble	224	144	7	-	6.67%
Deep Image low-res	256	-	1	-	7.96%
Deep Image high-res	512	-	1	24.88	7.42%
Deep Image ensemble	variable	-	-	-	5.98%
BN-Inception single crop	224	1	1	25.2%	7.82%
BN-Inception multicrop	224	144	1	21.99%	5.82%
BN-Inception ensemble	224	144	6	20.1%	**4.9%***

Next, ResNet

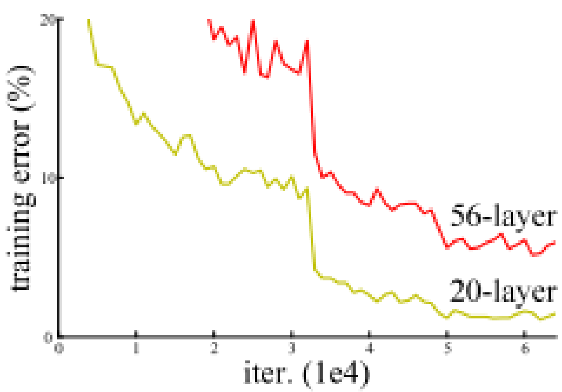
*source : Ioffe et al., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". ICML 2015

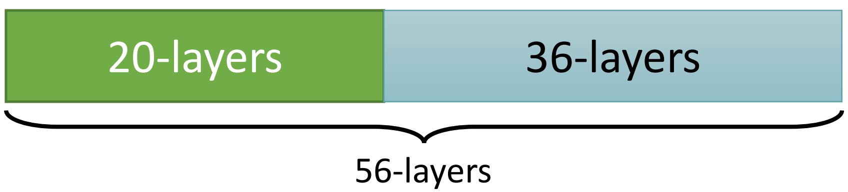
Evolution of CNN architectures: ResNet [He et al., 2016a]

- The winner of ILSVRC'15 (6.66% → **3.57%**)
- **ResNet** is the first architecture succeeded to train >100-layer networks
  - Prior works could until ~30 layers, but failed for the larger nets

What was the problem?

- 56-layer net gets higher training error than 20-layers network
- Deeper networks are much harder to optimize even if we use BNs
- It's not due to overfitting, but optimization difficulty
  - **Quiz**: Why is that?



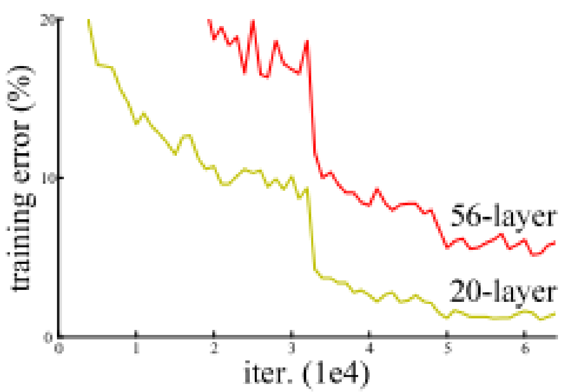


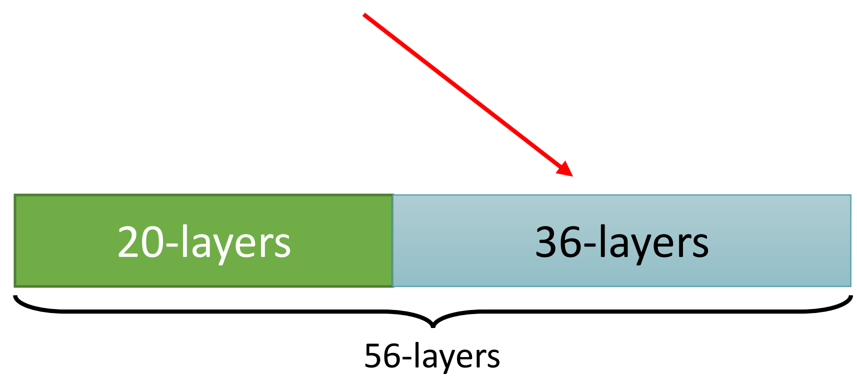
*source : He et al., "Deep residual learning for image recognition". CVPR 2016

- The winner of ILSVRC'15 (6.66**%** → **3.57%**)
- **ResNet** is the first architecture succeeded to train >100-layer networks
  - Prior works could until ~30 layers, but failed for the larger nets

What was the problem?

- It's not due to overfitting, but optimization difficulty
  - **Quiz**: Why is that?
- If the 56-layer model optimized well, then it **must be better** than the 20-layer
  - There is a trivial solution for the 36-layer: identity

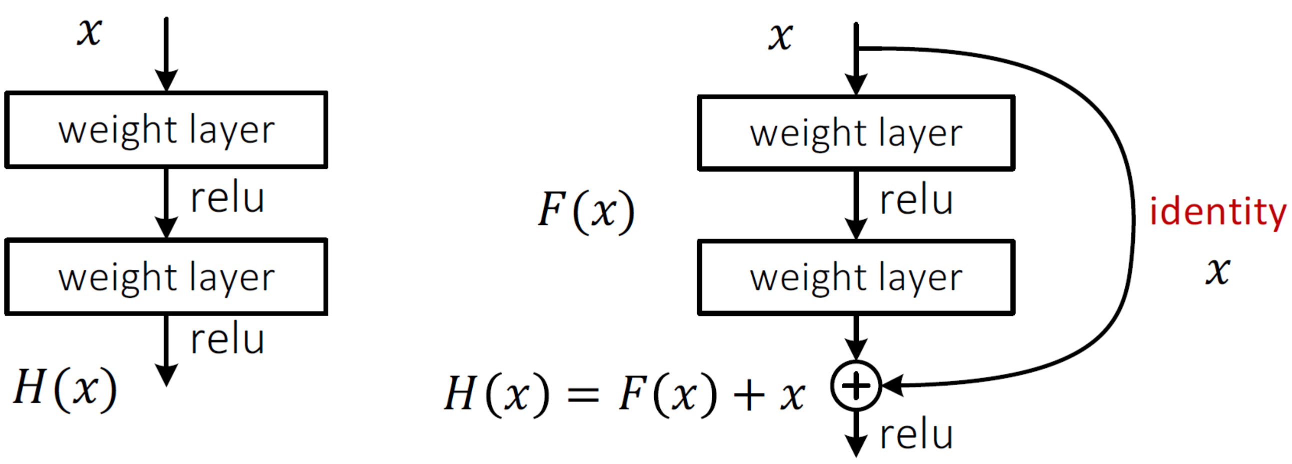




*source : He et al., "Deep residual learning for image recognition". CVPR 2016

- **Motivation:** A non-linear layer may struggle to represent an identity function
  - Due to its internal non-linearities, e.g. ReLU
  - This may cause the optimization difficulty on large networks

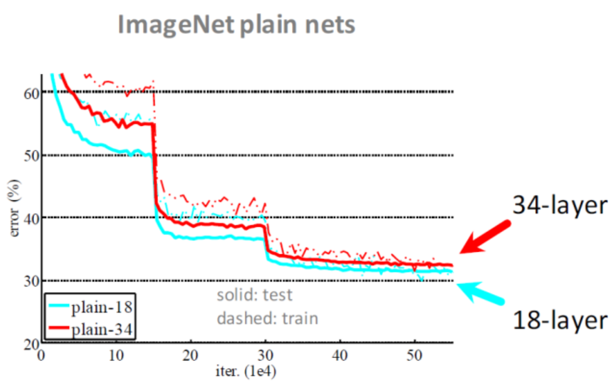
- **Idea:** Reparametrize each layer to make them easy to represent an *identity*
  - When all the weights are set to zero, the layer represents an identity

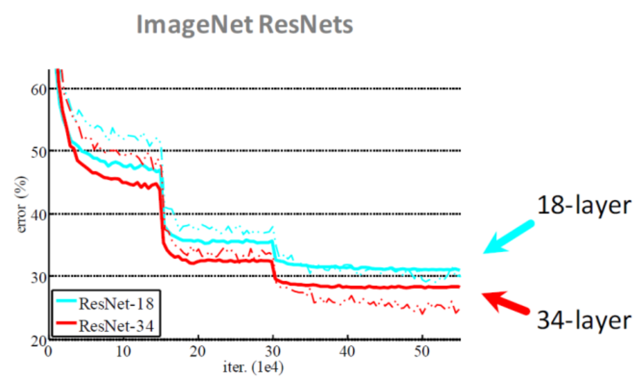


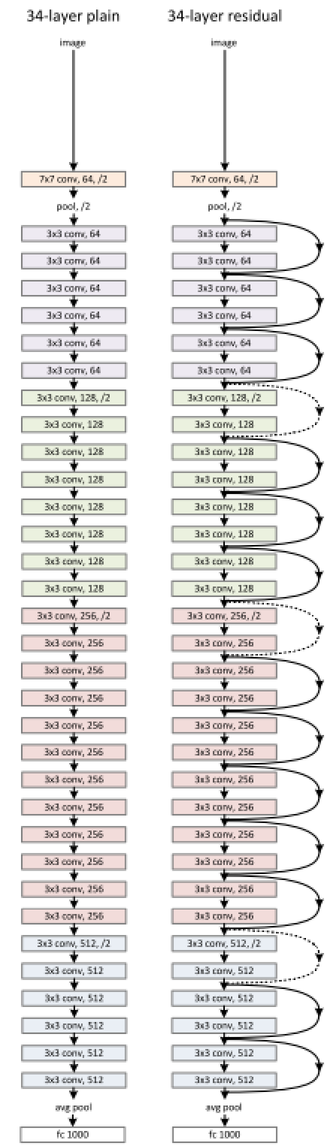
*source : He et al., "Deep residual learning for image recognition". CVPR 2016

Evolution of CNN architectures: ResNet [He et al., 2016a]

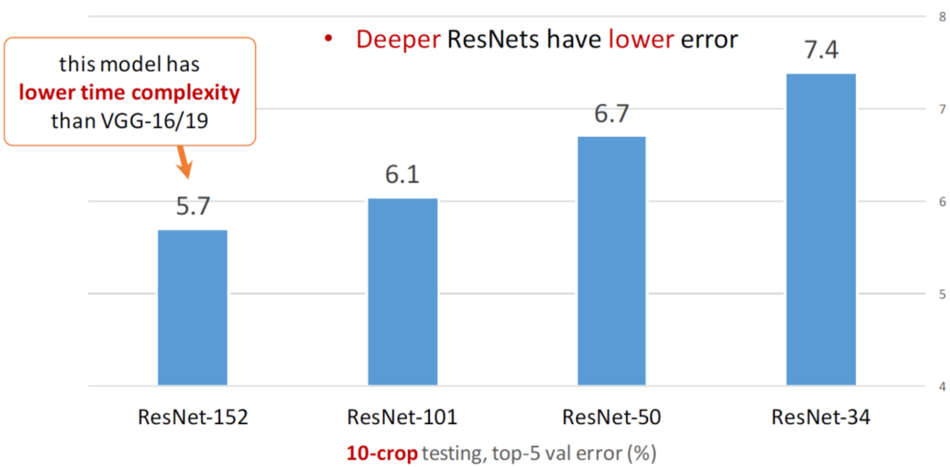
- Plain nets v.s. ResNets







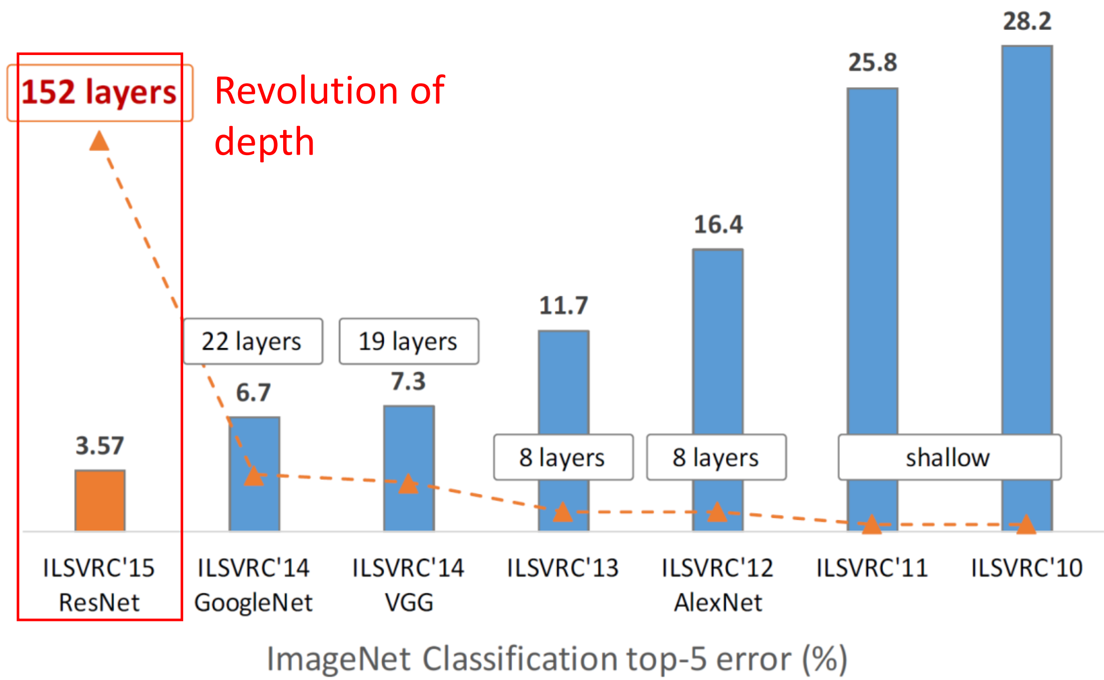
- Deeper ResNets can be trained without any difficulty



*sources :
- He et al., “Deep residual learning for image recognition”. CVPR 2016
- He, Kaiming, "Deep Residual Networks: Deep Learning Gets Way Deeper.” 2016.

Evolution of CNN architectures: ResNet [He et al., 2016a]

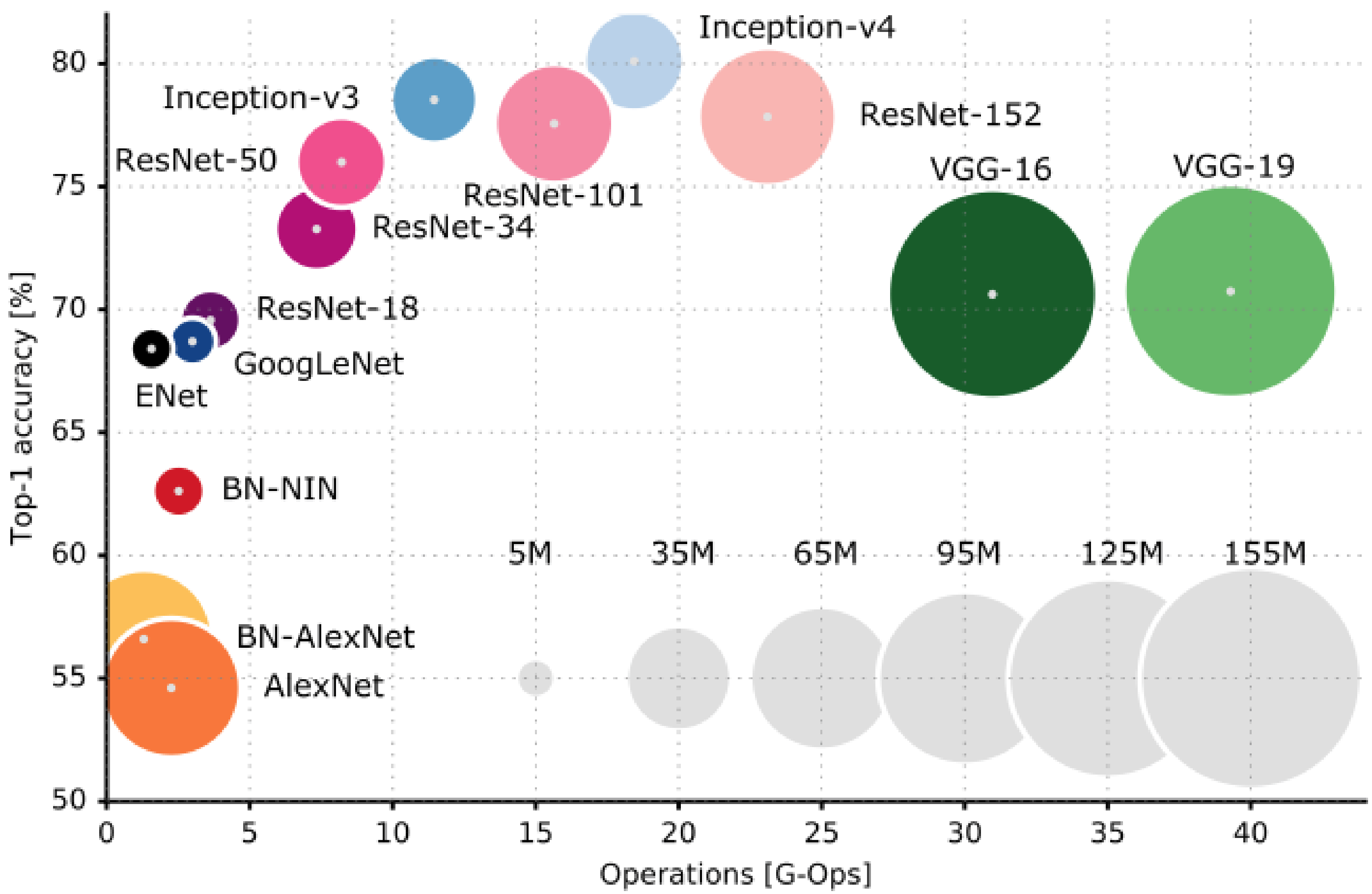
- Identity connection resolved a major difficulty on optimizing large networks
- **Revolution of depth**: Training >100-layer network without difficulty
  - Later, ResNet is revised to allow to train up to >1000 layers [He et al., 2016b]
- ResNet also shows good generalization ability as well



*sources :
- He et al., “Deep residual learning for image recognition”. CVPR 2016
- Kaiming He, "Deep Residual Networks: Deep Learning Gets Way Deeper.” 2016.
- He et al. "Identity mappings in deep residual networks.", ECCV 2016

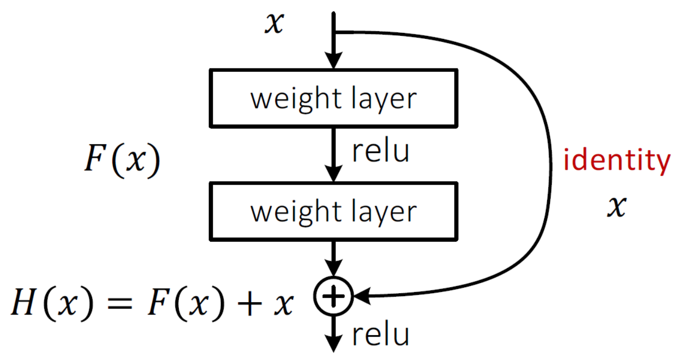
Evolution of CNN architectures

- Comparisons on ImageNet for a single model of popular CNNs



*source : https://towardsdatascience.com/neural-network-architectures-156e5bad51ba

- **Various architectures now are based on ResNet**
  - ResNet with stochastic depth [Huang et al., 2016]
  - Wide ResNet [Zagoruyko et al., 2016]
  - ResNet in ResNet [Targ et al., 2016]
  - ResNeXt [Xie et al., 2016]
  - PyramidNet [Han et al., 2016]
  - Inception-v4 [Szegedy et al., 2017]
  - DenseNet [Huang et al., 2017]
  - Dual Path Network [Chen et al., 2017]

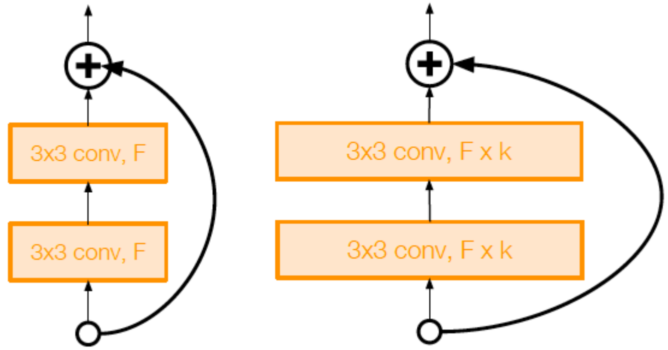


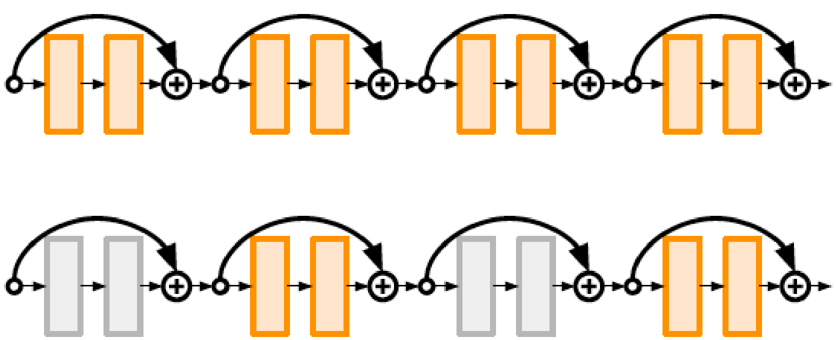
- **Transition of design paradigm:** Optimization ⇒ Generalization
  - People are now less concerned about optimization problems in a model
  - Instead, they now focus more on its generalization ability
  - "How well does an architecture generalize as its scale grows?"

*source : He et al., "Deep residual learning for image recognition". CVPR 2016

- **Wide Residual Networks** [Zagoruyko et al., 2016]
  - Residuals can also work to enlarge the width, not only its depth
  - Residual blocks with $\times$ k wider filters
  - Increasing width instead of depth can be more computationally efficient
    - GPUs are much better on handling "**wide-but-shallow**" than "thin-but-deep“
  - WRN-50 outperforms ResNet-152

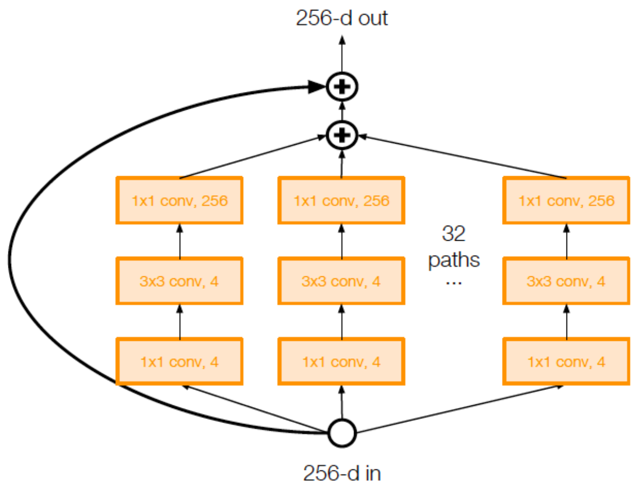
- **Deep Networks with Stochastic Depth** [Huang et al., 2016]
  - Randomly drop a subset of layers during training
  - Bypassing via identity connections
  - Reduces gradient vanishing, and training time as well



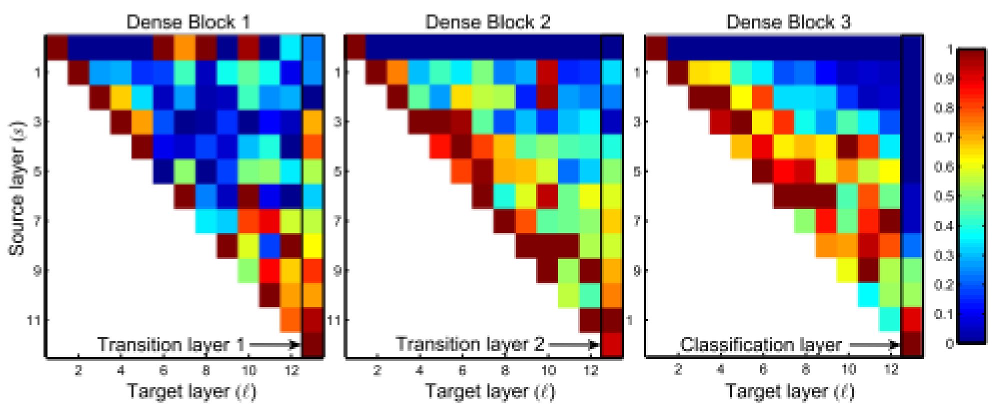


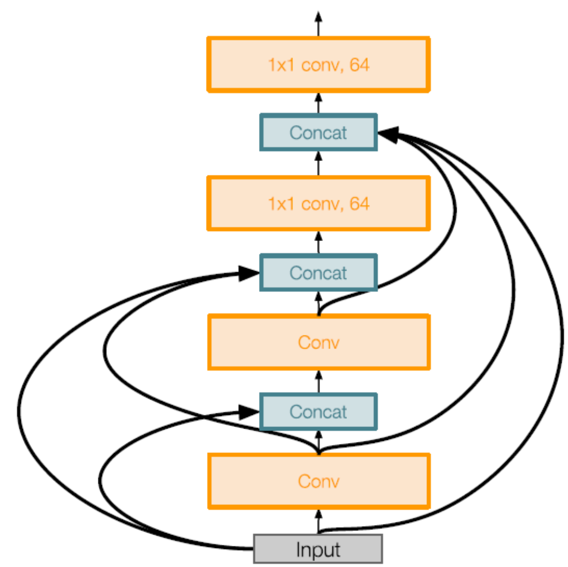
*source : Fei-Fei Li et al. (2018), CS231n Lecture 9, Stanford University

- **ResNeXt** [Xie et al., 2016]
    - Aggregating multiple parallel paths inside a residual block ("**cardinality**")
    - Increasing cardinality is more effective than going deeper or wider



- **DenseNet** [Huang et al. 2017]
    - Passing all the previous representation directly via concatenation of features
    - Strengthens **feature propagation** and **feature reuse**

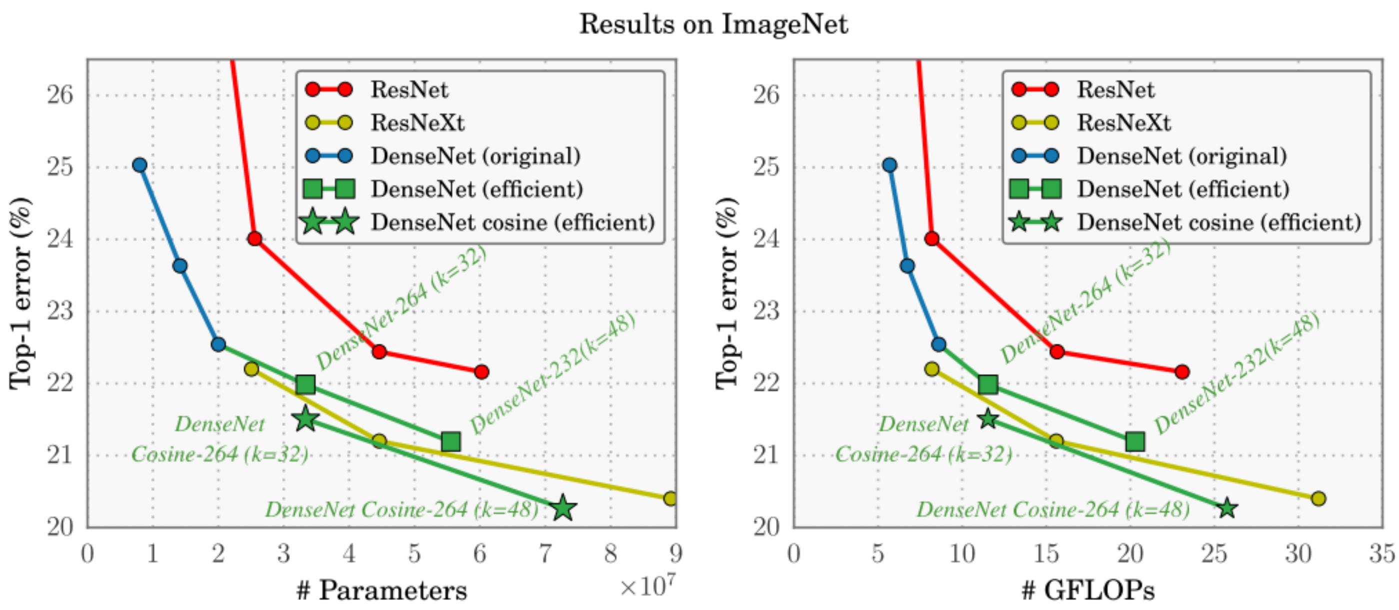




Dense Block

*source : Fei-Fei Li et al. (2018), CS231n Lecture 9, Stanford University

- **ResNeXt** [Xie et al., 2016]
  - Aggregating multiple parallel paths inside a residual block ("**cardinality**")
  - Increasing cardinality is more effective than going deeper or wider
- **DenseNet** [Huang et al. 2017]
  - Passing all the previous representation directly via concatenation of features
  - Strengthens **feature propagation** and **feature reuse**



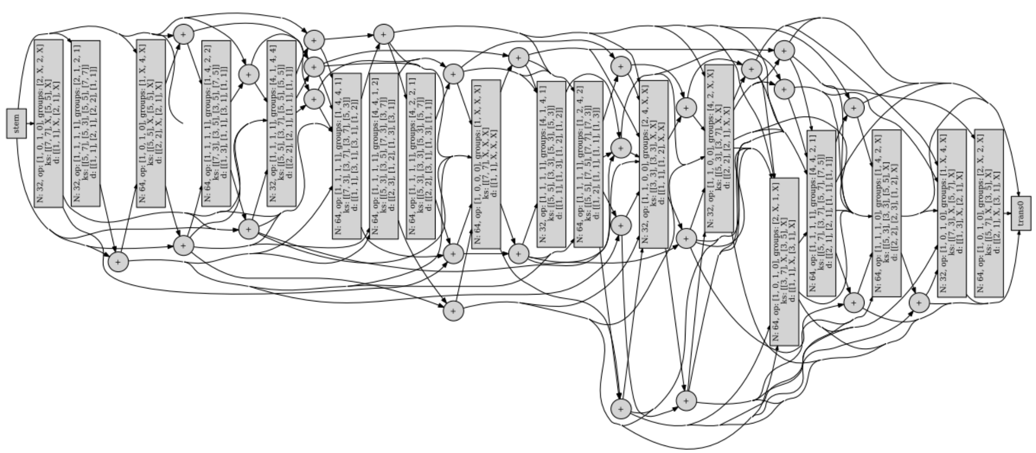
Next, automation of design

*source : Fei-Fei Li et al. (2018), CS231n Lecture 9, Stanford University

Toward automation of network design

- Although the CNN architecture has evolved greatly, our **design principles are still relying on heuristics**
  - Smaller kernel and smaller stride, increase cardinality instead of width ...

- Recently, there have been works on automatically finding a structure which can outperform existing human-crafted architectures
  1. **Search space**: Naïvely searching every model is nearly impossible
  2. **Searching algorithm**: Evaluating each model is very costly, and black-boxed



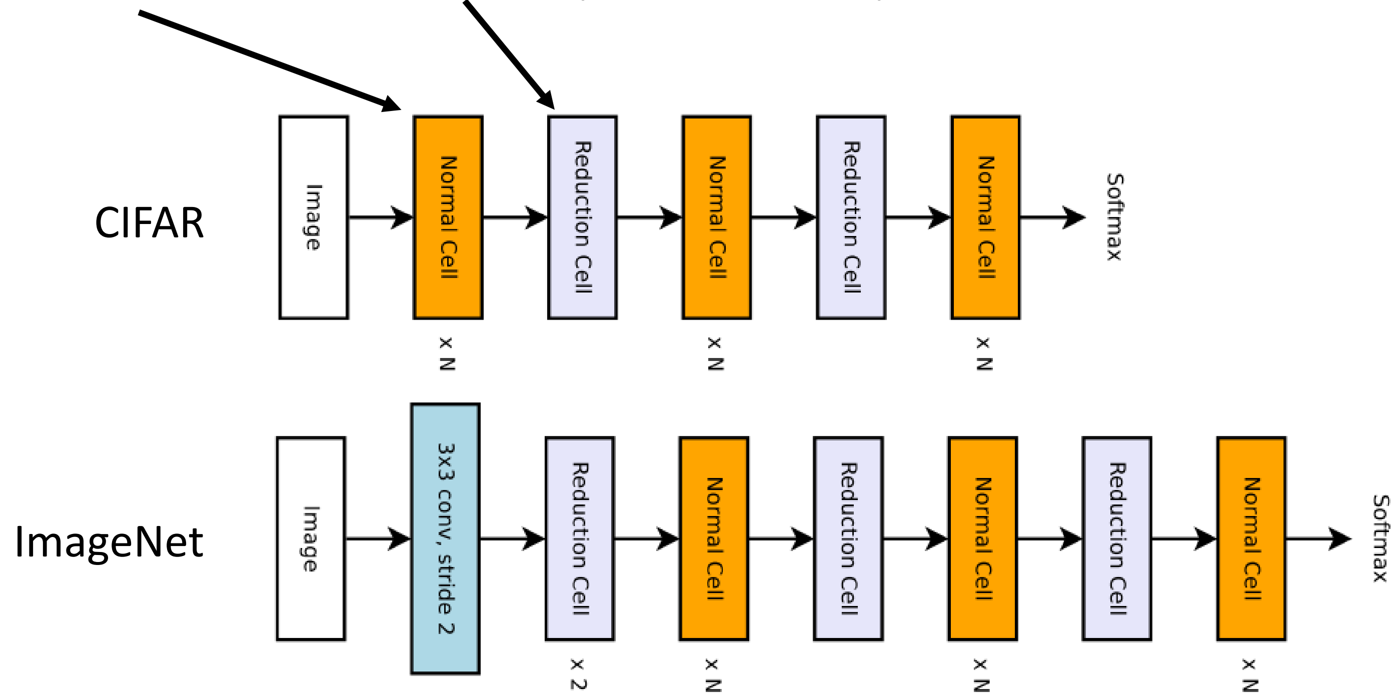
A sample architecture found in [Brock et al., 2018]

Next, NASNet

*source : Brock et al., “SMASH: One-Shot Model Architecture Search through HyperNetworks”, ICLR 2018

- **Designing a good search space** is important in architecture searching
- **NASNet** reduces the search space by incorporating our design principles

- **Motivation**: modern architectures are built simply: **a repeated modules**
  - Try not to search the whole model, but only cells modules
  - **Normal cell** and **Reduction cell** (cell w/ stride 2)

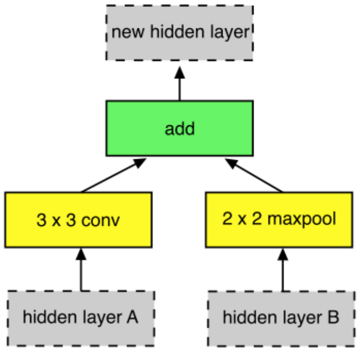


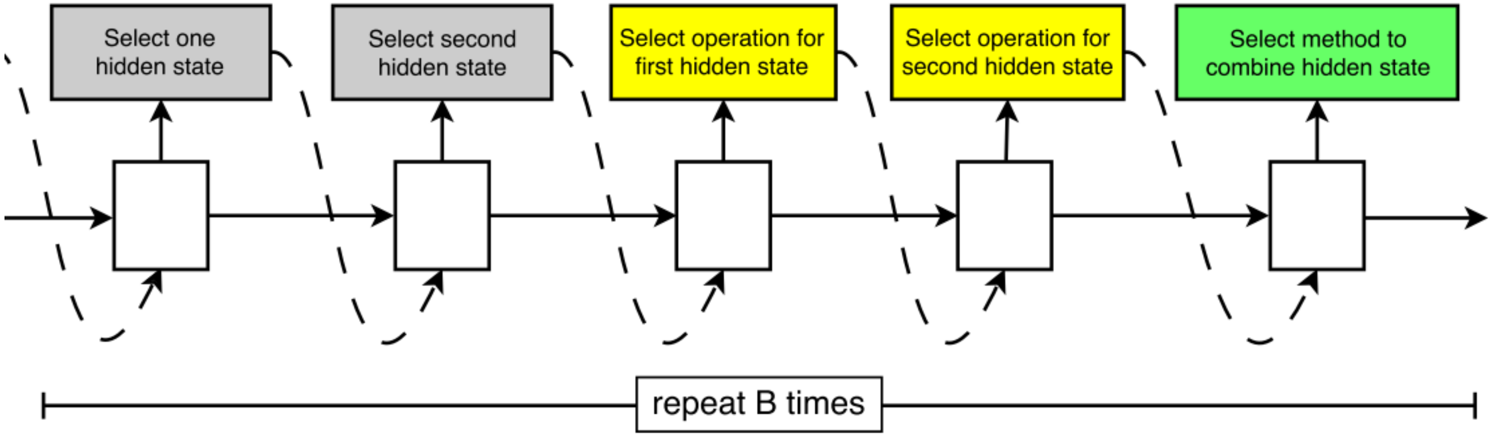
*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

Toward automation of network design: NASNet [Zoph et al., 2018]

- **Designing a good search space** is important in architecture searching
- **NASNet** reduces the search space by incorporating our design principles

- Each cell consists of B **blocks**
- Each block is determined by **selecting methods**
  1. Select two hidden states from h_i, h_{i-1} or of existing block
  2. Select methods to process for each of the selected states
  3. Select a method to combine the two states
     - (1) **element-wise addition** or (2) **concatenation**

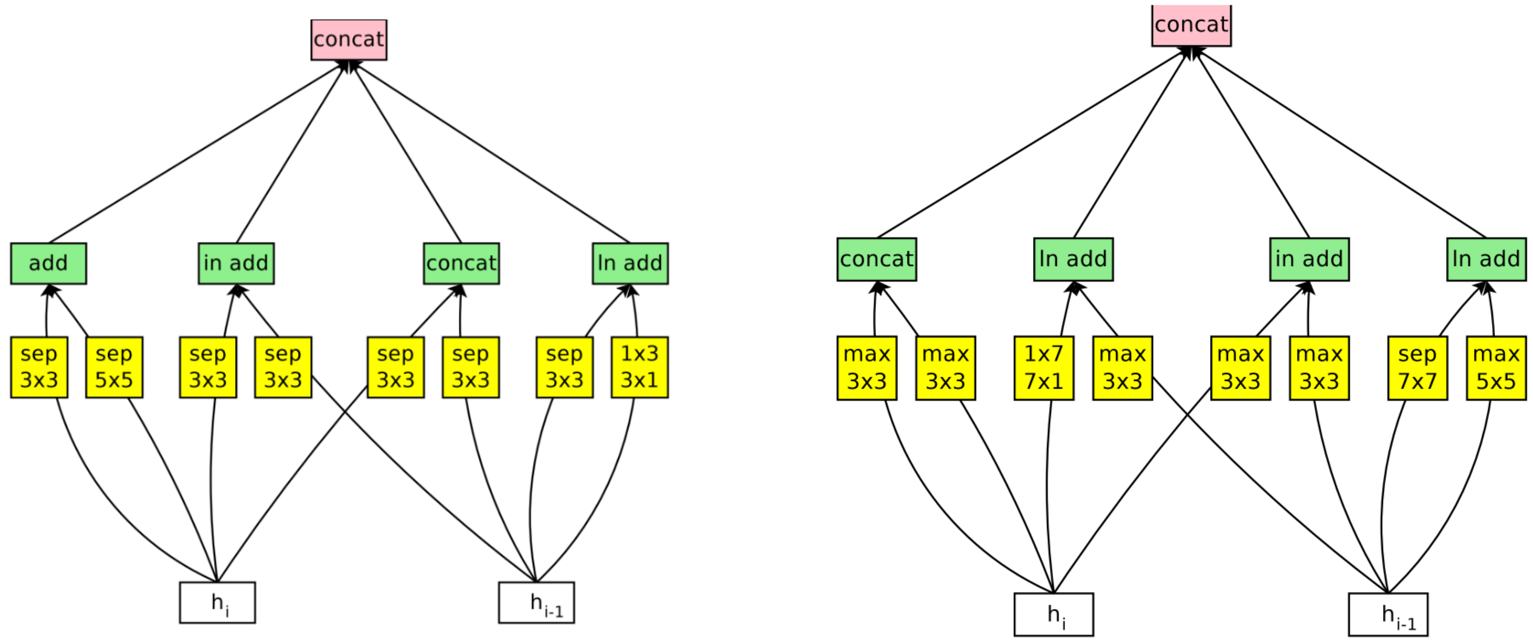




*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

- **Designing a good search space** is important in architecture searching
- **NASNet** reduces the search space by incorporating our design principles

- Each cell consists of B **blocks**
  - **Example**: $B = 4$



Normal Cell   Reduction Cell

*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

Toward automation of network design: NASNet [Zoph et al., 2018]

- **Designing a good search space** is important in architecture searching
- **NASNet** reduces the search space by incorporating our design principles

- Set of methods to be selected based on their prevalence in the CNN literature

- identity
- 1x7 then 7x1 convolution
- 3x3 average pooling
- 5x5 max pooling
- 1x1 convolution
- 3x3 depthwise-separable conv
- 7x7 depthwise-separable conv
- 1x3 then 3x1 convolution
- 3x3 dilated convolution
- 3x3 max pooling
- 7x7 max pooling
- 3x3 convolution
- 5x5 depthwise-seperable conv

- Any searching methods can be used
  - **Random search** [Bergstra et al., 2012] could also work
  - **RL-based search** [Zoph et al., 2016] is mainly used in this paper

*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

Toward automation of network design: NASNet [Zoph et al., 2018]

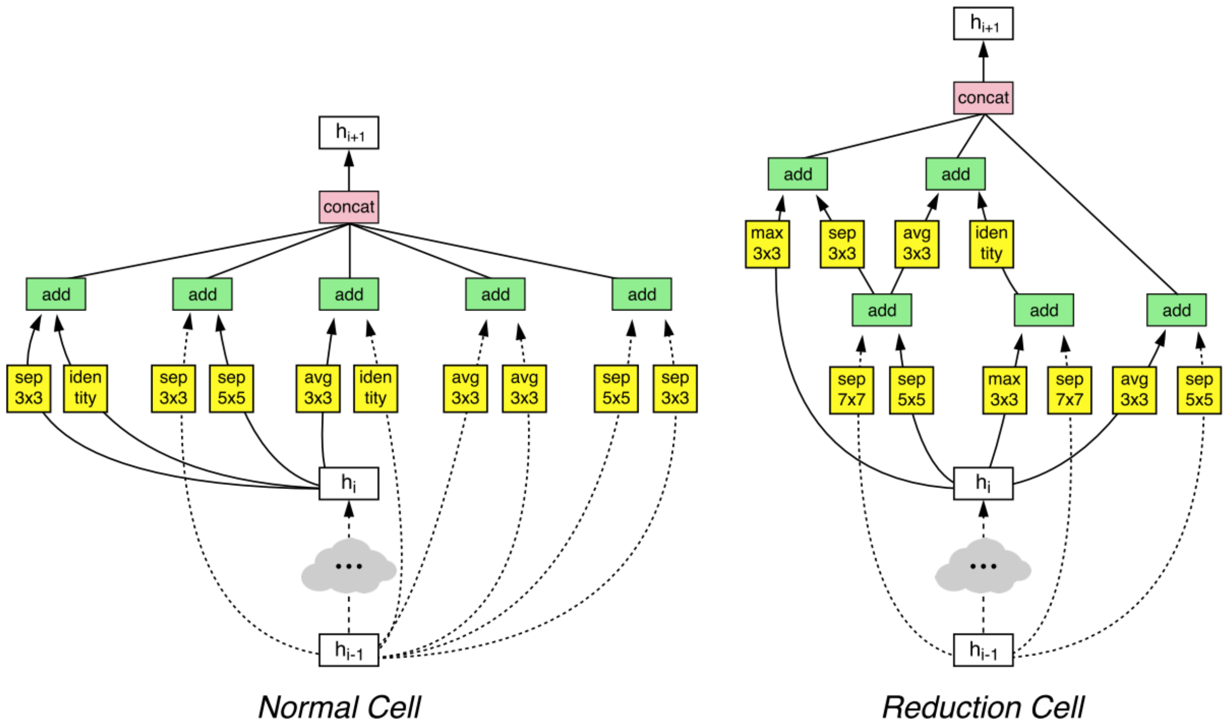
- The pool of workers consisted of **500 GPUs**, processing **over 4 days**
- All architecture searches are performed on **CIFAR-10**
  - NASNet-A: State-of-the-art error rates could be achieved
  - NASNet-B/C: Extremely parameter-efficient models were also found

model	depth	# params	error rate (%)
DenseNet ($L = 40, k = 12$) [26]	40	1.0M	5.24
DenseNet($L = 100, k = 12$) [26]	100	7.0M	4.10
DenseNet ($L = 100, k = 24$) [26]	100	27.2M	3.74
DenseNet-BC ($L = 100, k = 40$) [26]	190	25.6M	3.46
Shake-Shake 26 2x32d [18]	26	2.9M	3.55
Shake-Shake 26 2x96d [18]	26	26.2M	2.86
Shake-Shake 26 2x96d + cutout [12]	26	26.2M	2.56
NAS v3 [70]	39	7.1M	4.47
NAS v3 [70]	39	37.4M	3.65
NASNet-A (6 @ 768)	-	3.3M	3.41
NASNet-A (6 @ 768) + cutout	-	3.3M	2.65
NASNet-A (7 @ 2304)	-	27.6M	2.97
NASNet-A (7 @ 2304) + cutout	-	27.6M	2.40
NASNet-B (4 @ 1152)	-	2.6M	3.73
NASNet-C (4 @ 640)	-	3.1M	3.59

*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

Toward automation of network design: NASNet [Zoph et al., 2018]

- The pool of workers consisted of **500 GPUs**, processing **over 4 days**
- All architecture searches are performed on **CIFAR-10**
  - NASNet-A: State-of-the-art error rates could be achieved
  - NASNet-B/C: Extremely parameter-efficient models were also found



NASNet-A

*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

Toward automation of network design: NASNet [Zoph et al., 2018]

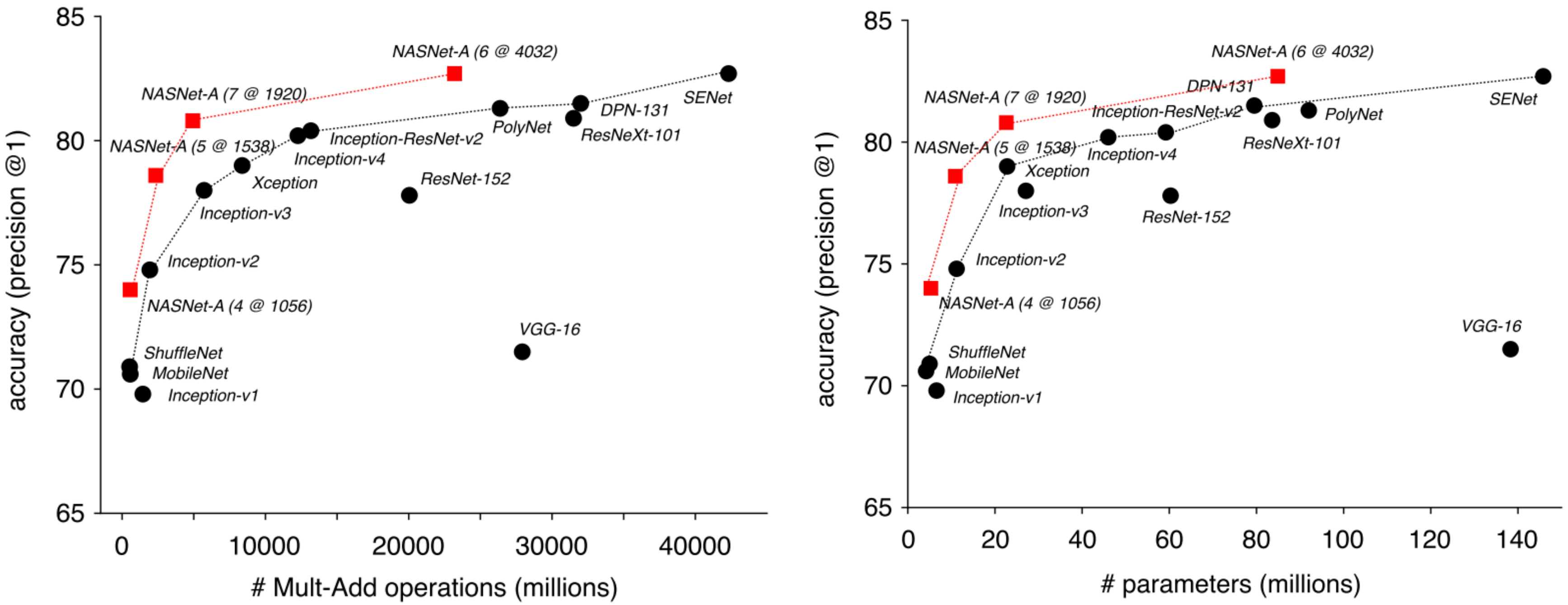
- The pool of workers consisted of 500 GPUs, processing over 4 days
- All architecture searches are performed on CIFAR-10
- **Cells found in CIFAR-10 could also transferred well into ImageNet**

Model	image size	# parameters	Mult-Adds	Top 1 Acc. (%)	Top 5 Acc. (%)
Inception V2 [29]	224×224	11.2 M	1.94 B	74.8	92.2
NASNet-A (5 @ 1538)	**299×299**	**10.9 M**	**2.35 B**	**78.6**	**94.2**
Inception V3 [59]	299×299	23.8 M	5.72 B	78.0	93.9
Xception [9]	299×299	22.8 M	8.38 B	79.0	94.5
Inception ResNet V2 [57]	299×299	55.8 M	13.2 B	80.4	95.3
NASNet-A (7 @ 1920)	**299×299**	**22.6 M**	**4.93 B**	**80.8**	**95.3**
ResNeXt-101 (64 x 4d) [67]	320×320	83.6 M	31.5 B	80.9	95.6
PolyNet [68]	331×331	92 M	34.7 B	81.3	95.8
DPN-131 [8]	320×320	79.5 M	32.0 B	81.5	95.8
SENet [25]	**320×320**	**145.8 M**	**42.3 B**	**82.7**	**96.2**
NASNet-A (6 @ 4032)	**331×331**	**88.9 M**	**23.8 B**	**82.7**	**96.2**

*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

Toward automation of network design: NASNet [Zoph et al., 2018]

- The pool of workers consisted of 500 GPUs, processing over 4 days
- All architecture searches are performed on CIFAR-10
- **Cells found in CIFAR-10 could also transferred well into ImageNet**



*source : Zoph et al., “Learning Transferable Architectures for Scalable Image Recognition”, CVPR 2018

Toward automation of network design

- Architecture searching is still an active research area
  - AmoebaNet [Real et al., 2018]
  - Efficient-NAS (ENAS) [Pham et al., 2018]
  - NAONet [Luo et al., 2018]

Model	Error(%)	#params	GPU Days
DenseNet-BC [19]	3.46	25.6M	/
ResNeXt-29 [43]	3.58	68.1M	/
NASNet-A [48]	3.41	3.3M	2000
NASNet-B [48]	3.73	2.6M	2000
NASNet-C [48]	3.59	3.1M	2000
Hier-EA [28]	3.75	15.7M	300
AmoebaNet-A [38]	3.34	3.2M	3150
AmoebaNet-B [38]	3.37	2.8M	3150
AmoebaNet-B [38]	3.04	13.7M	3150
AmoebaNet-B [38]	2.98	34.9M	3150
AmoebaNet-B + Cutout [38]	2.13	34.9M	3150
ENAS [37]	3.54	4.6M	0.45
PNAS [27]	3.41	3.2M	225
DARTS + Cutout [29]	2.83	4.6M	4
NAONet	3.18	10.6M	200
NAONet	2.98	28.6M	200
NAONet + Cutout	2.07	128M	200
NAONet-WS	3.53	3.7M	0.4

*source : Luo et al., "Neural Architecture Optimization", Arxiv 2018

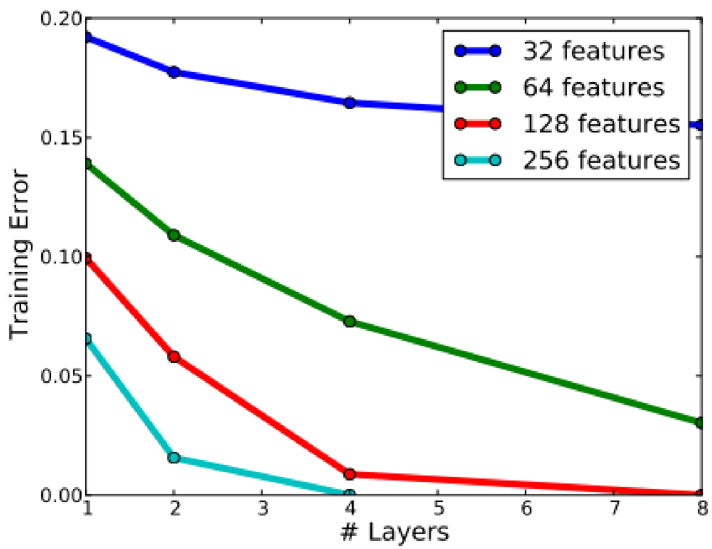
Table of Contents

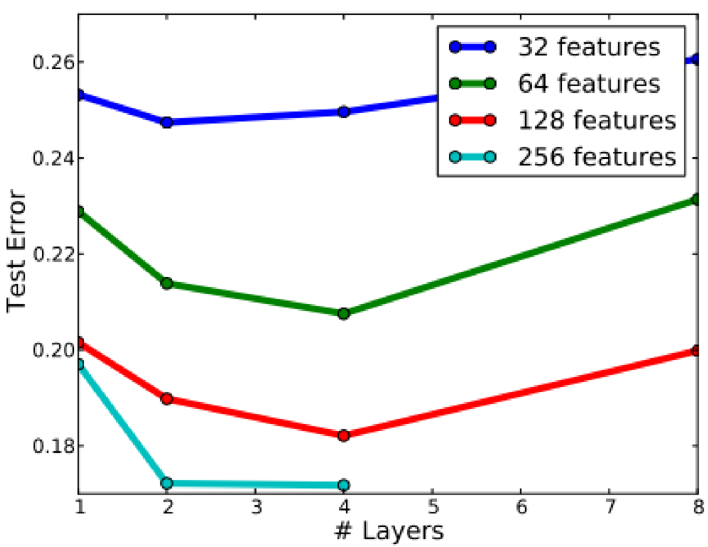
ResNets behave like ensembles of relatively shallow nets [Veit et al., 2016]

- ResNet improved generalization by **revolution of depth**

  Quiz: But, does it fully explain why deep ResNets generalize well?

- Increasing depth **does not always mean** better generalization
  - Naïve CNNs are very easy to overfit on deeper networks [Eigen et al., 2014]

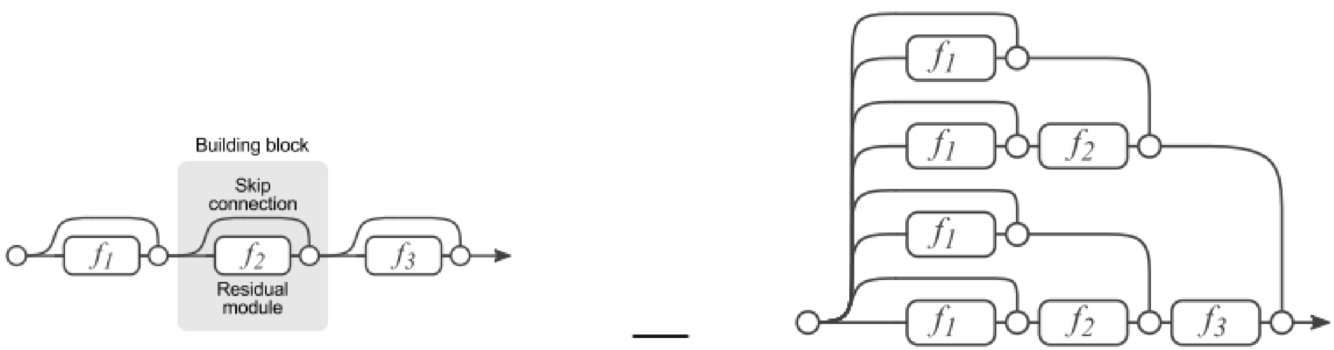




*source : Eigen et al., “Understanding Deep Architectures using a Recursive Convolutional Network”, Arxiv 2014

- **Veit et al.** (2016): ResNet can be viewed as a collection of many paths, instead of a single ultra-deep network
  - Each module in a ResNet receives a **mixture of 2^{n-1} different distributions**

$$
\begin{aligned}
y_3 &= y_2 + f_3(y_2) \\
&= [y_1 + f_2(y_1)] + f_3(y_1 + f_2(y_1)) \\
&= [y_0 + f_1(y_0) + f_2(y_0 + f_1(y_0))] + f_3(y_0 + f_1(y_0) + f_2(y_0 + f_1(y_0)))
\end{aligned}
$$



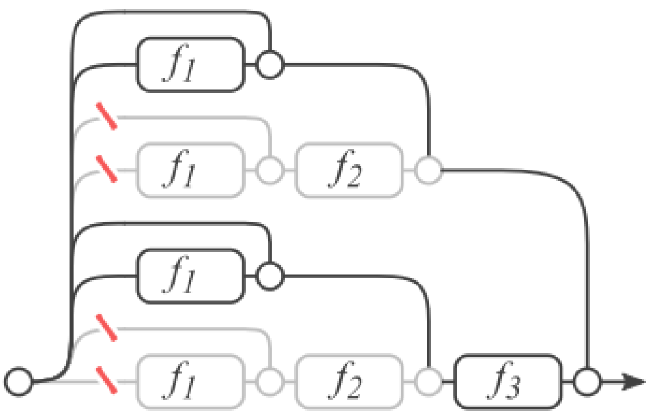
(a) Conventional 3-block residual network

(b) Unraveled view of (a)

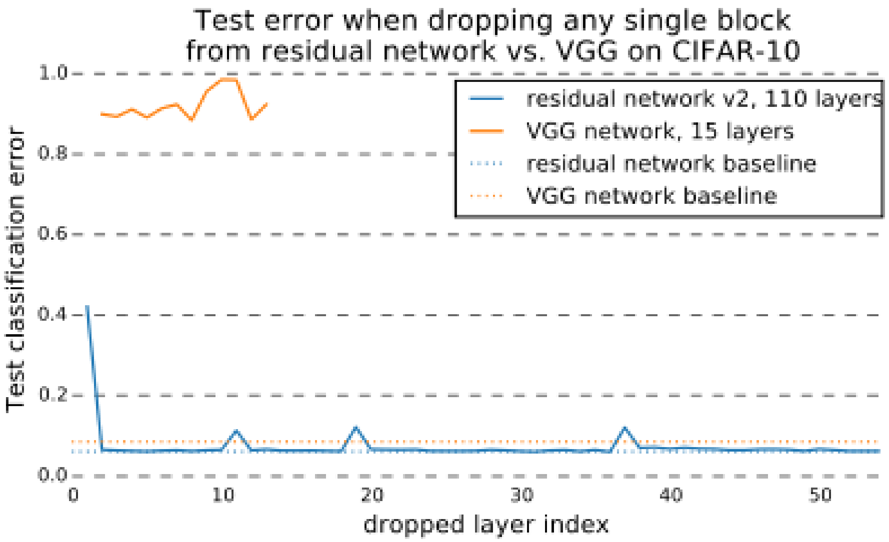
*source : Veit et al., “ResNets behave like ensembles of relatively shallow nets”, NIPS 2016

ResNets behave like ensembles of relatively shallow nets [Veit et al., 2016]

- **Veit et al.** (2016): ResNet can be viewed as a collection of many paths, instead of a single ultra-deep network
  - Deleting a module in ResNet has a minimal effect on performance
  - Similar effect as removing 2^{n-1} paths out of 2^n: still 2^{n-1} paths alive!



(a) Deleting f_2 from unraveled view



Next, visualizing loss functions in CNN

*source : Veit et al., "ResNets behave like ensembles of relatively shallow nets", NIPS 2016

Visualizing the loss landscape of neural nets [Li et al., 2018]

- **Trainability of neural nets** is highly dependent on network architecture
- However, the effect of each choice on the underlying loss surface is unclear
  - Why are we able to minimize highly non-convex neural loss?
  - Why do the resulting minima generalize?

- **Li et al.** (2018) analyzes **random-direction 2D plot of loss** around local minima

$$f(\alpha, \beta) = L(\theta^* + \alpha\delta + \beta\eta)$$

Local minima (θ^*)   **Random directions** (δ, η)

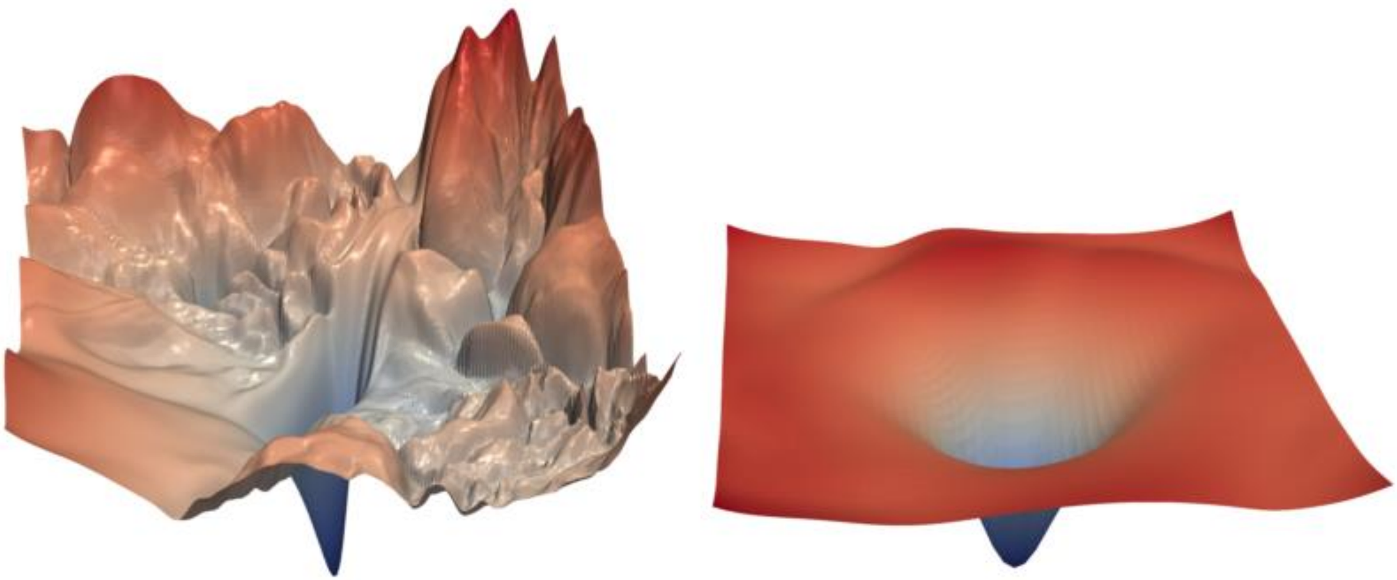
  - δ and η are sampled from a random Gaussian distribution
  - To remove some scaling effect, δ and η are normalized filter-wise

$$\delta_{i,j} \leftarrow \frac{\delta_{i,j}}{||\delta_{i,j}||}||\theta_{i,j}||$$

i^{th} **layer,** j^{th} **filter**

*source : Li et al., "Visualizing the loss landscape of neural nets", ICLR Workshop 2018

- **Li et al.** (2018) analyzes **random-direction 2D plot of loss** around local minima
- **Modern architectures** prevent the loss to be chaotic as depth increases



(a) without skip connections

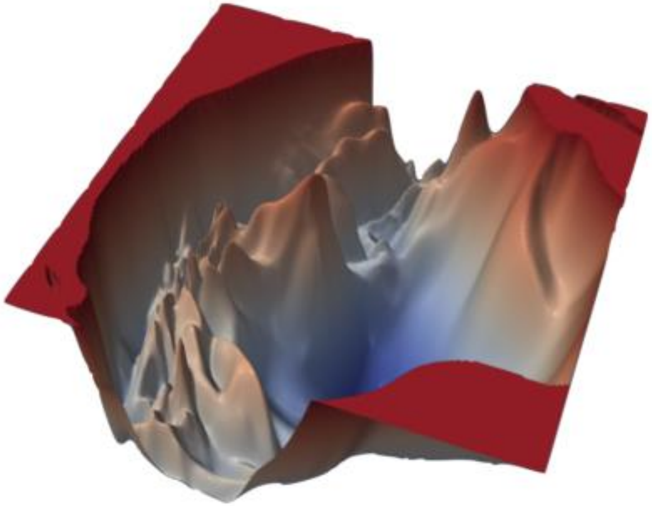
(b) with skip connections

ResNet-56

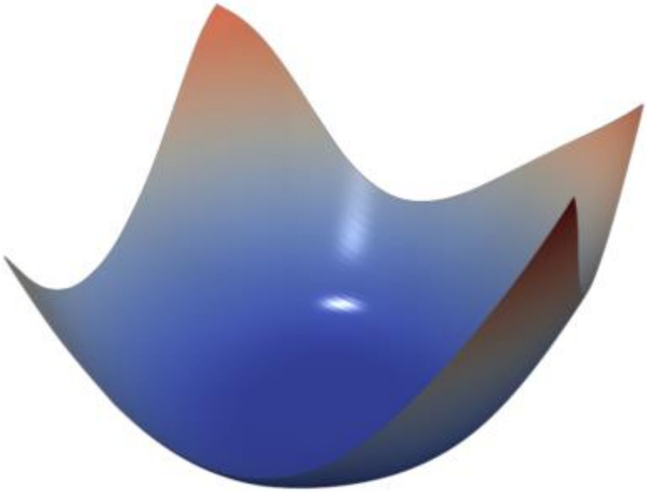
*source : Li et al., "Visualizing the loss landscape of neural nets", ICLR Workshop 2018

Visualizing the loss landscape of neural nets [Li et al., 2018]

- **Li et al.** (2018) analyzes **random-direction 2D plot of loss** around local minima
- **Modern architectures** prevent the loss to be chaotic as depth increases



(a) 110 layers, no skip connections



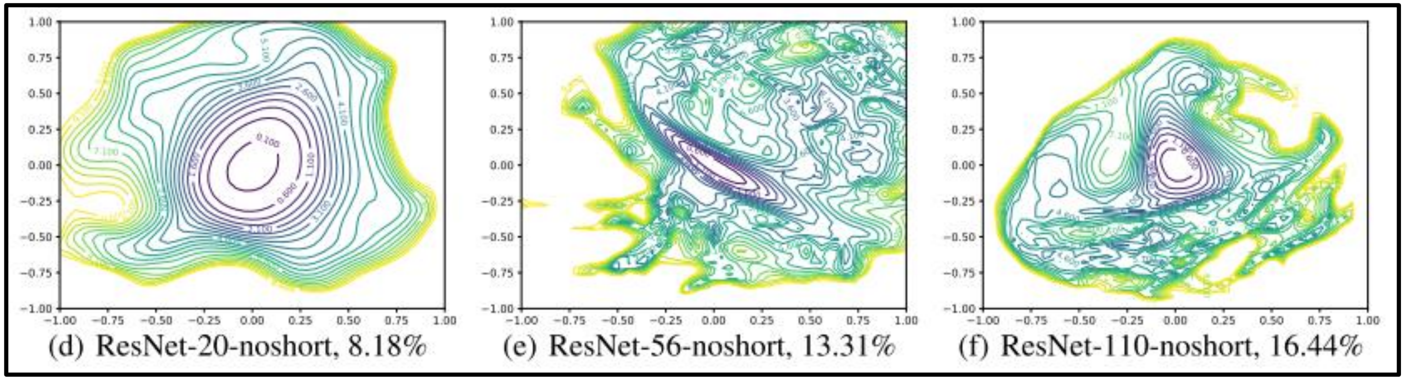
(b) DenseNet, 121 layers

*source : Li et al., "Visualizing the loss landscape of neural nets", ICLR Workshop 2018

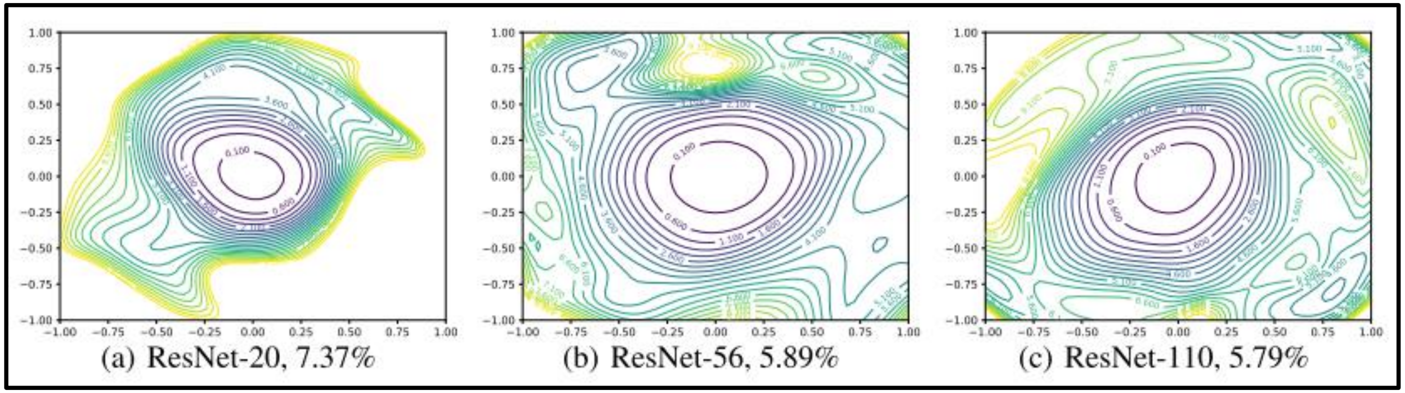
- **Li et al.** (2018) analyzes **random-direction 2D plot of loss** around local minima

- **Modern architectures** prevent the loss to be chaotic as depth increases

ResNet, **no shortcuts** ⇒ sharp minima



ResNet ⇒ flat minima

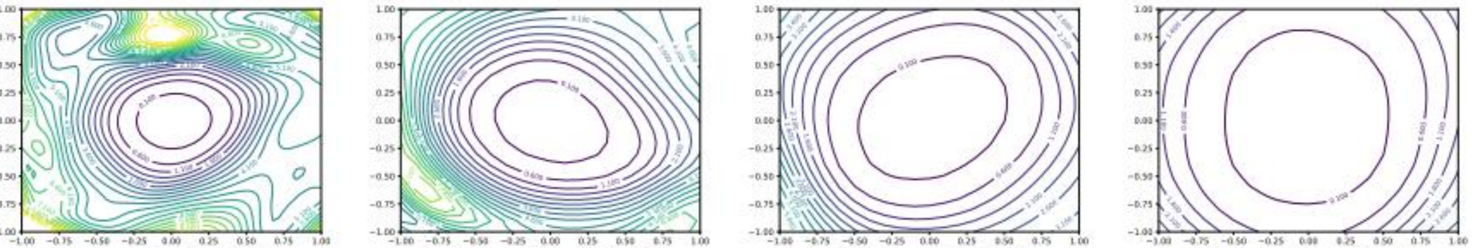


*source : Li et al., "Visualizing the loss landscape of neural nets", ICLR Workshop 2018

- **Li et al.** (2018) analyzes **random-direction 2D plot of loss** around local minima

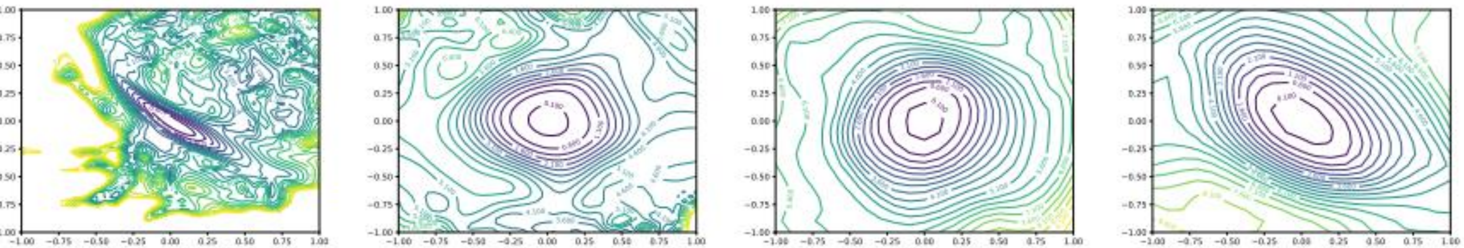
- **Wide-ResNet** lead the network toward more flat minimizer
  - WideResNet-56 with **width-multiplier** $k = 1, 2, 4, 8$
  - Increased width flatten the minimizer in ResNet

WRN-56



(a) $k = 1$, 5.89% (b) $k = 2$, 5.07% (c) $k = 4$, 4.34% (d) $k = 8$, 3.93%

WRN-56, **no shortcuts**



(e) $k = 1$, 13.31% (f) $k = 2$, 10.26% (g) $k = 4$, 9.69% (h) $k = 8$, 8.70%

Next, minimum energy paths in CNNs

*source : Li et al., "Visualizing the loss landscape of neural nets", ICLR Workshop 2018

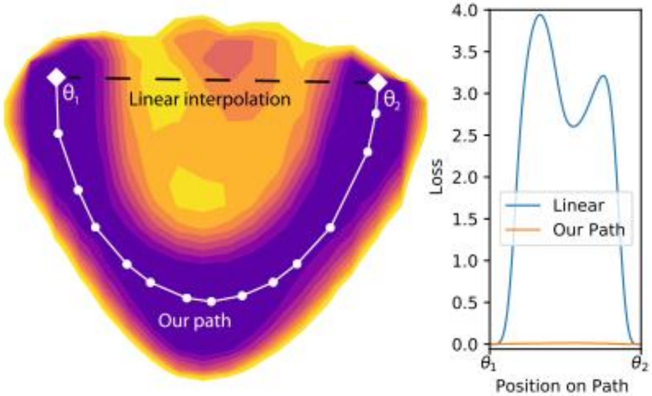
Essentially no barriers in neural network energy landscape [Draxler et al., 2018]

- **Draxler et al.** (2018) analyzes **minimum energy paths** [Jónsson et al., 1998] between two local minima θ_1 and θ_2 of a given model:

$$p(\theta_i, \theta_2)^* = \underset{\text{path } p:\ \theta_1 \to \theta_2}{\operatorname{argmin}} \left(\max_{\theta \in p} L(\theta) \right)$$

  - They found a path $\theta_1 \to \theta_2$ with almost zero barrier
    - A path that **keeps low loss constantly** both in training and test
  - The gap vanishes as the model grows, especially on modern architectures
    - e.g. ResNet, DenseNet

- Minima of a loss of deep neural networks are perhaps on a single connected manifold



DenseNet-40-12

*source : Draxler et al., "Essentially no barriers in neural network energy landscape", ICML 2018

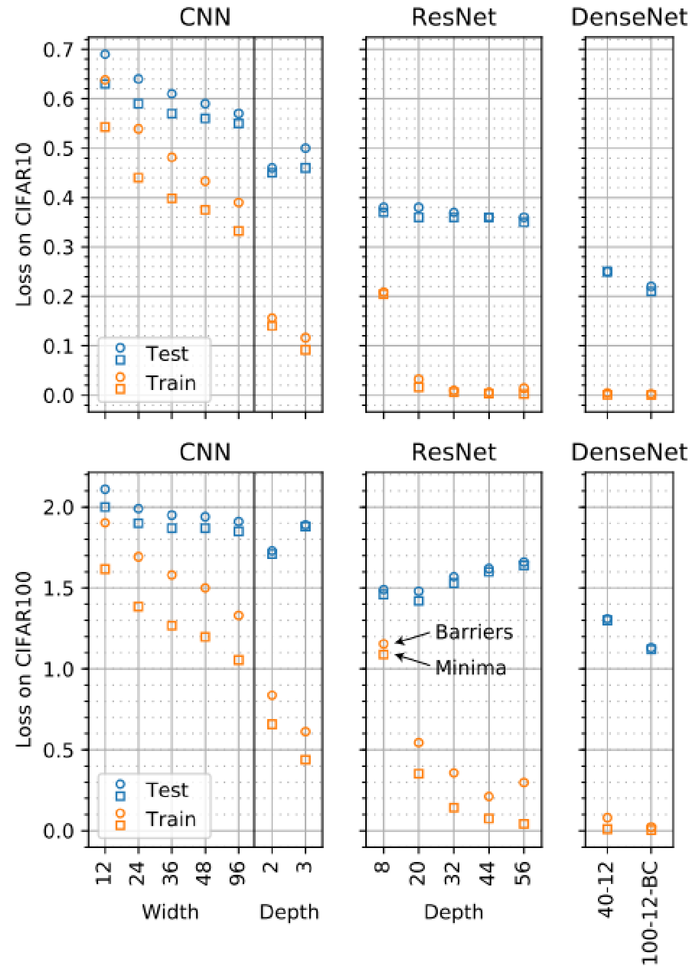
Essentially no barriers in neural network energy landscape [Draxler et al., 2018]

- For a given model with two local minima θ_1 and θ_2, they applied **AutoNEB** [Kolsbjerg et al., 2016] to find a minimum energy path
  - A state-of the-art for connecting minima from molecular statistical mechanics

- The deeper and wider an architecture, the lower are the saddles between minima

- They essentially vanish for current-day deep architectures

- The **test accuracy** is also preserved
  - **CIFAR-10**: < +0.5%
  - **CIFAR-100**: < +2.2%

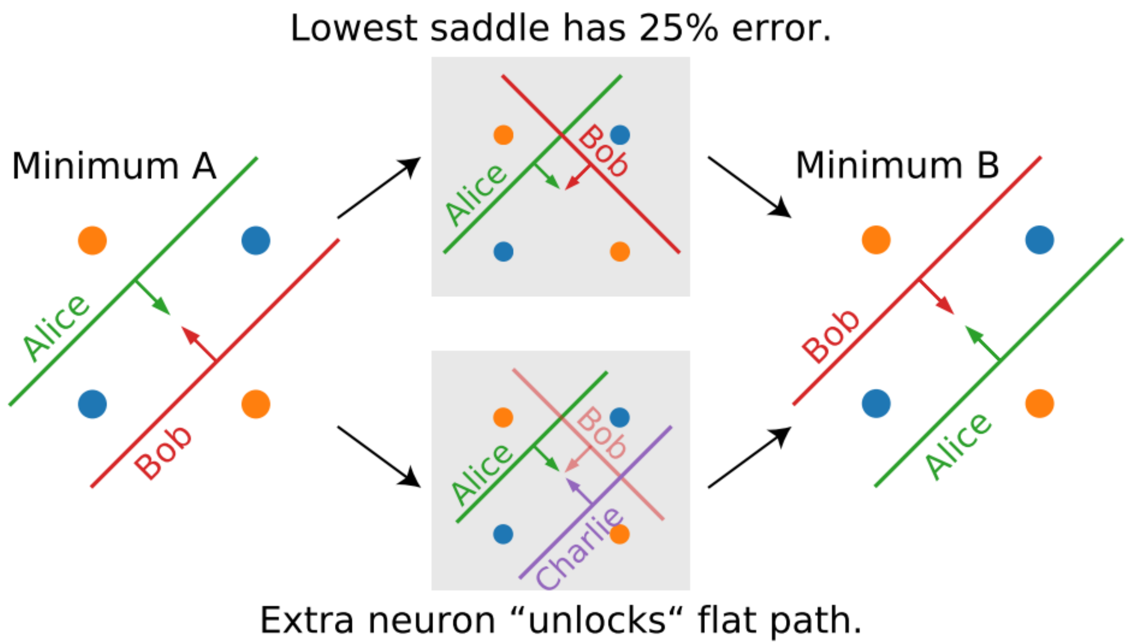


*source : Draxler et al., “Essentially no barriers in neural network energy landscape”, ICML 2018

Essentially no barriers in neural network energy landscape [Draxler et al., 2018]

- The deeper and wider an architecture, the lower are the barriers
- They essentially vanish for current-day deep architectures

- Why do this phenomenon happen?
  - **Parameter redundancy** may help to flatten the neural loss



*source : Draxler et al., “Essentially no barriers in neural network energy landscape”, ICML 2018

Summary

- The **larger** the network, the **more difficult** it is to design
  1. Optimization difficulty
  2. Generalization difficulty

- **ImageNet challenge** contributed greatly to development of CNN architectures
- **ResNet**: Optimization ⇒ Generalization
  - Many variants of ResNet have been emerged

- Very recent trends towards **automation of network design**

- Many **observational study** supports the advantages of modern CNN architectures

References

[Jónsson et al., 1998] Jónsson, H., Mills, G., & Jacobsen, K. W. (1998). Nudged elastic band method for finding minimum energy paths of transitions. In Classical and quantum dynamics in condensed phase simulations (pp. 385-404).
link : https://www.worldscientific.com/doi/abs/10.1142/9789812839664_0016

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.
link : https://ieeexplore.ieee.org/abstract/document/726791/

[Bergstra et al., 2012] Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(Feb), 281-305.
link : http://www.jmlr.org/papers/v13/bergstra12a.html

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).
link : http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

[Farabet et al., 2013] Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning hierarchical features for scene labeling. IEEE transactions on pattern analysis and machine intelligence, 35(8), 1915-1929.
link : https://ieeexplore.ieee.org/abstract/document/6338939/

[Eigen et al., 2014] Eigen, D., Rolfe, J., Fergus, R., & LeCun, Y. (2013). Understanding Deep Architectures using a Recursive Convolutional Network. ArXiv Preprint ArXiv:1312.1847, 1–9.
link : http://arxiv.org/abs/1312.1847

[Simonyan et al., 2014] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
link: https://arxiv.org/abs/1409.1556

References

[Zeiler et al., 2014] Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham.
link : https://link.springer.com/chapter/10.1007/978-3-319-10590-1_53

[Ioffe et al., 2015] Ioffe, S. & Szegedy, C.. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning, in PMLR 37:448-456
link : http://proceedings.mlr.press/v37/ioffe15.html

[Ren et al., 2015] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*(pp. 91-99).
link : http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks

[Russakovsky et al., 2015] Russakovsky, O. et al. (2015). Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3), 211-252.
link : https://link.springer.com/article/10.1007/s11263-015-0816-y

[Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).
link : https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Szegedy_Going_Deeper_With_2015_CVPR_paper.html

[Han et al., 2016] Han, D., Kim, J., & Kim, J. (2017, July). Deep pyramidal residual networks. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on (pp. 6307-6315). IEEE.
link : https://ieeexplore.ieee.org/document/8100151/

References

[He et al., 2016a] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
link : https://ieeexplore.ieee.org/document/7780459/

[He et al., 2016b] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Identity mappings in deep residual networks. In European conference on computer vision (pp. 630-645). Springer, Cham.
link : https://link.springer.com/chapter/10.1007/978-3-319-46493-0_38

[Huang et al., 2016] Huang, G., Sun, Y., Liu, Z., Sedra, D., & Weinberger, K. Q. (2016). Deep networks with stochastic depth. In European Conference on Computer Vision (pp. 646-661).
link : https://link.springer.com/chapter/10.1007/978-3-319-46493-0_39

[Kolsbjerg et al., 2016] Kolsbjerg, E. L., Groves, M. N., & Hammer, B. (2016). An automated nudged elastic band method. The Journal of chemical physics, 145(9), 094107.
link : https://aip.scitation.org/doi/abs/10.1063/1.4961868

[Targ et al., 2016] Targ, S., Almeida, D., & Lyman, K. (2016). Resnet in Resnet: generalizing residual architectures. arXiv preprint arXiv:1603.08029.
link : https://arxiv.org/abs/1603.08029

[Veit et al., 2016] Veit, A., Wilber, M. J., & Belongie, S. (2016). Residual networks behave like ensembles of relatively shallow networks. In Advances in Neural Information Processing Systems (pp. 550-558).
link : http://papers.nips.cc/paper/6556-residual-networks-behave-like-ensembles-of-relatively-shallow-networks

[Xie et al., 2016] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017, July). Aggregated residual transformations for deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on (pp. 5987-5995). IEEE.
link : http://openaccess.thecvf.com/content_cvpr_2017/papers/Xie_Aggregated_Residual_Transformations_CVPR_2017_paper.pdf

References

[Zagoruyko et al., 2016] Zagoruyko, S. and Komodakis, N. (2016). Wide Residual Networks. In Proceedings of the British Machine Vision Conference (pp. 87.1-87.12).
link : http://www.bmva.org/bmvc/2016/papers/paper087/index.html

[Zoph et al., 2016] Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.
link : https://arxiv.org/abs/1611.01578

[Chen et al., 2017] Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S., & Feng, J. (2017). Dual path networks. In Advances in Neural Information Processing Systems (pp. 4467-4475).
link : https://papers.nips.cc/paper/7033-dual-path-networks

[Huang et al., 2017] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017, July). Densely Connected Convolutional Networks. In CVPR (Vol. 1, No. 2, p. 3).
link : http://openaccess.thecvf.com/content_cvpr_2017/papers/Huang_Densely_Connected_Convolutional_CVPR_2017_paper.pdf

[Szegedy et al., 2017] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. In AAAI (Vol. 4, p. 12).
link : https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/download/14806/14311

[Brock et al., 2018] Brock, A., Lim, T., Ritchie, J. M., & Weston, N. (2018). SMASH: one-shot model architecture search through hypernetworks. In International Conference on Learning Representations.
link : https://openreview.net/forum?id=rydeCEhs-

[Draxler et al., 2018] Draxler, F., Veschgini, K., Salmhofer, M. & Hamprecht, F. (2018). Essentially No Barriers in Neural Network Energy Landscape. Proceedings of the 35th International Conference on Machine Learning, in PMLR 80:1309-1318.
link : http://proceedings.mlr.press/v80/draxler18a.html

References

[Luo et al., 2018] Luo, R., Tian, F., Qin, T., Chen, E. & Liu, T. (2018) Neural Architecture Optimization. arXiv preprint arXiv:1808.07233.
link : https://arxiv.org/abs/1808.07233

[Li et al., 2018] Li, H., Xu, Z., Taylor, G., & Goldstein, T. (2017). Visualizing the loss landscape of neural nets. arXiv preprint arXiv:1712.09913.
link : https://arxiv.org/abs/1712.09913

[Pham et al., 2018] Pham, H., Guan, M., Zoph, B., Le, Q. & Dean, J.. (2018). Efficient Neural Architecture Search via Parameters Sharing. Proceedings of the 35th International Conference on Machine Learning, in PMLR 80:4095-4104
link : http://proceedings.mlr.press/v80/pham18a.html

[Real et al., 2018] Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2018). Regularized evolution for image classifier architecture search. arXiv preprint arXiv:1802.01548.
link : https://arxiv.org/abs/1802.01548

[Zoph et al., 2018] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2017). Learning transferable architectures for scalable image recognition. arXiv preprint arXiv:1707.07012, 2(6).
link : http://openaccess.thecvf.com/content_cvpr_2018/papers/Zoph_Learning_Transferable_Architectures_CVPR_2018_paper.pdf