Advanced Deep Spatial Models

AI602: Recent Advances in Deep Learning

Lecture 2

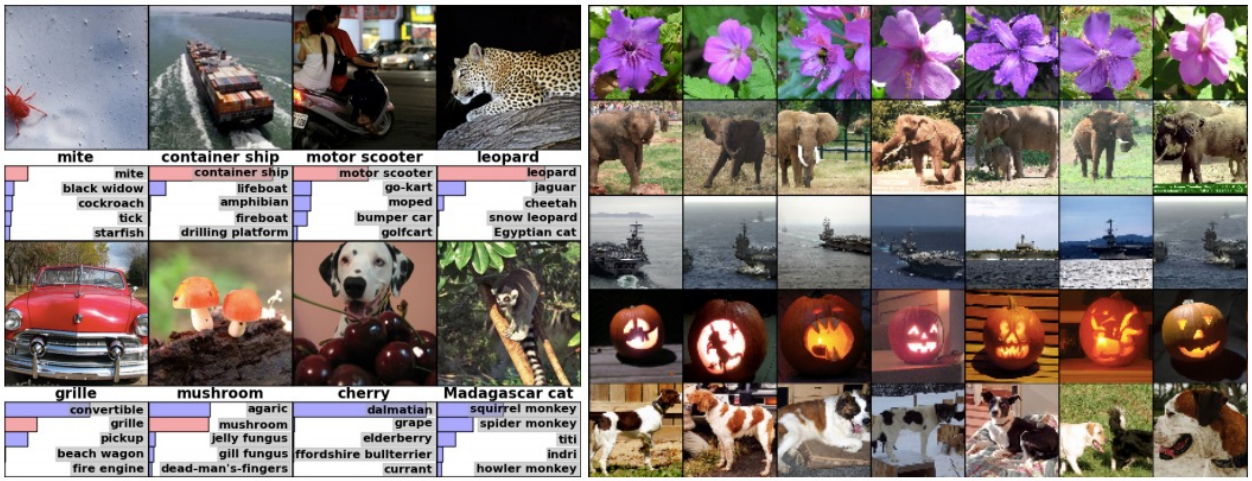
Slide made by

Sukmin Yun and Jongheon Jeong

KAIST EE & AI

Convolutional neural networks have been tremendously successful in practical applications;

Classification and retrieval [Krizhevsky et al., 2012]



Detection [Ren et al., 2015]

Segmentation [Farabet et al., 2013]

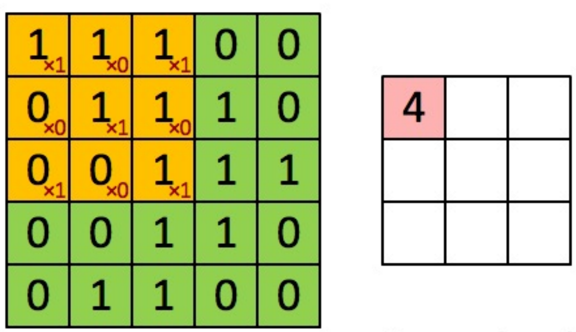
Overview: Convolutional Neural Networks

Neural networks that use convolution in place of general matrix multiplication

- Sharing parameters across **multiple image locations**
- Translation equivariant (invariant with **pooling**) operation

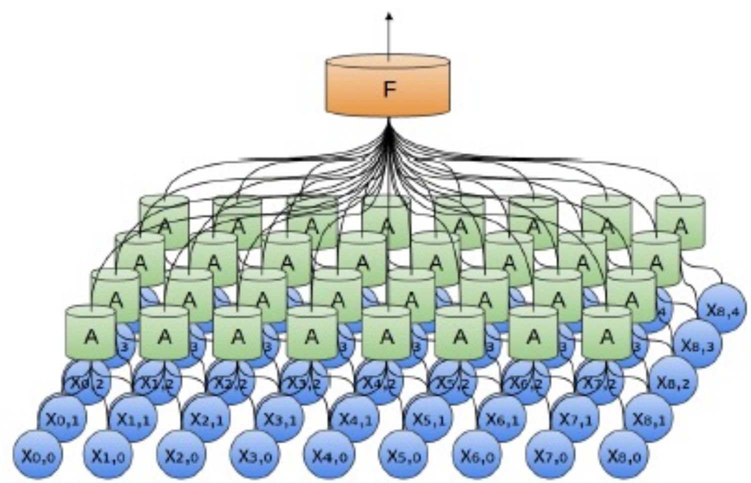
Specialized for processing data that has a known, grid-like topology

- e.g., time-series data (1D grid), image data (2D grid)



Image

Convolved Feature

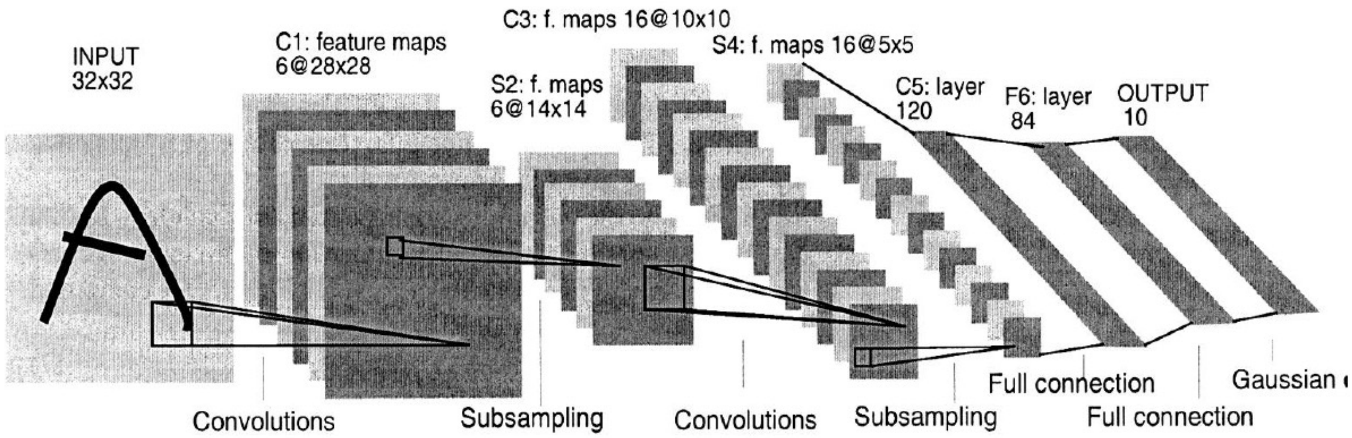


*sources :
- https://www.cc.gatech.edu/~san37/post/dlhc-cnn/
- http://colah.github.io/posts/2014-07-Conv-Nets-Modular/

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]

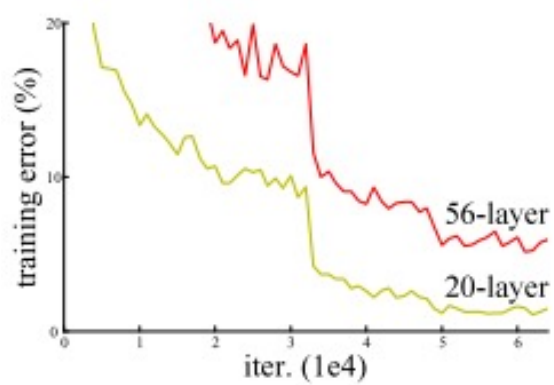


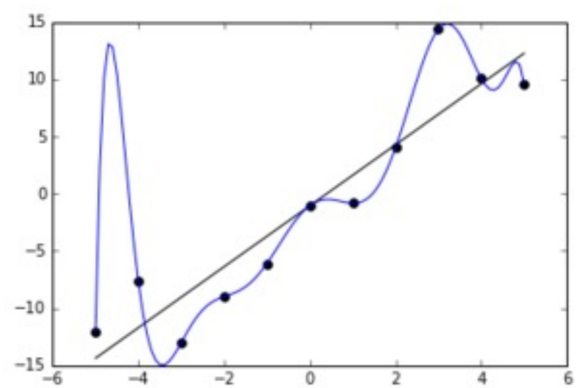
- However, **LeNet** is not enough to solve real-world problems in AI domain
  - CNNs are typically applied to extremely complicated domains, e.g. raw RGB images
  - We need to design **a larger model** to solve them adequately

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

Table of Contents

Table of Contents

Part 1. Basics

- Evolution of CNN architectures
- Batch normalization and ResNet

Part 2. Advanced Topics

- Toward automation of network design
- Dilated and Deformable convolution
- Attention module in CNNs
- Observational Study on CNN Architectures

Part 3. Beyond CNN

- Transformer architecture for Vision
- MLP architecture for Vision

Part 1. Basics

- Evolution of CNN architectures
- Batch normalization and ResNet

Part 2. Advanced Topics

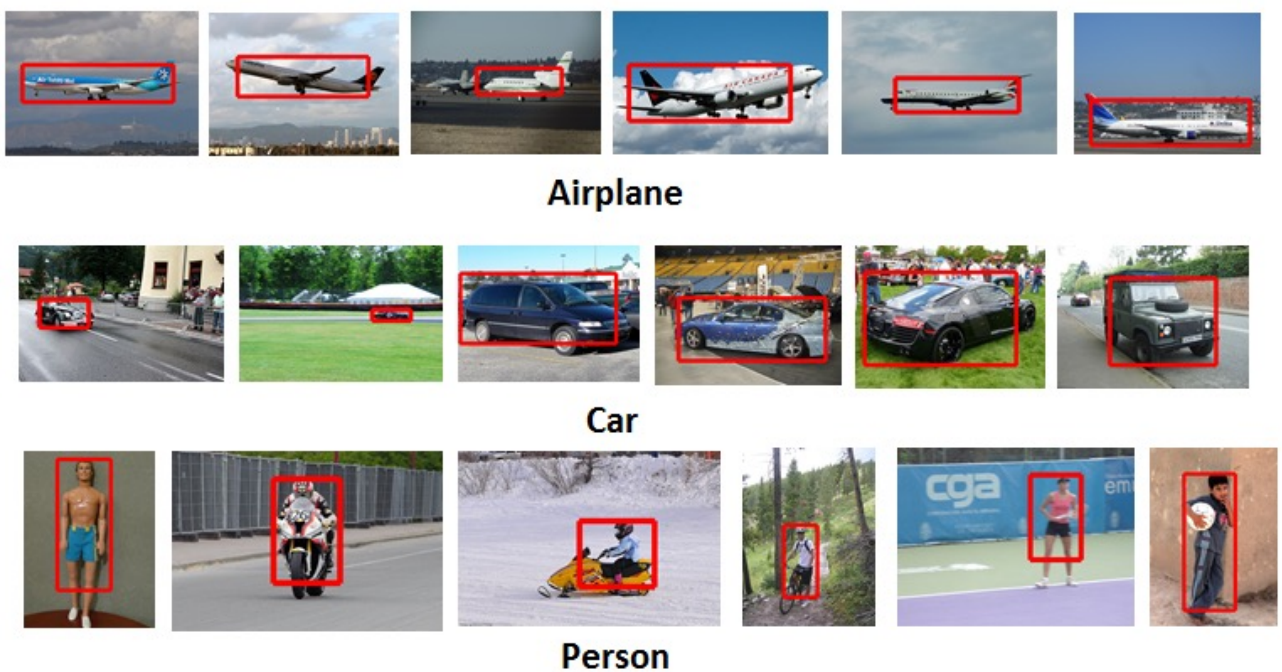
- Toward automation of network design
- Dilated and Deformable convolution
- Attention module in CNNs
- Observational Study on CNN Architectures

Part 3. Beyond CNN

- Transformer architecture for Vision
- MLP architecture for Vision

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

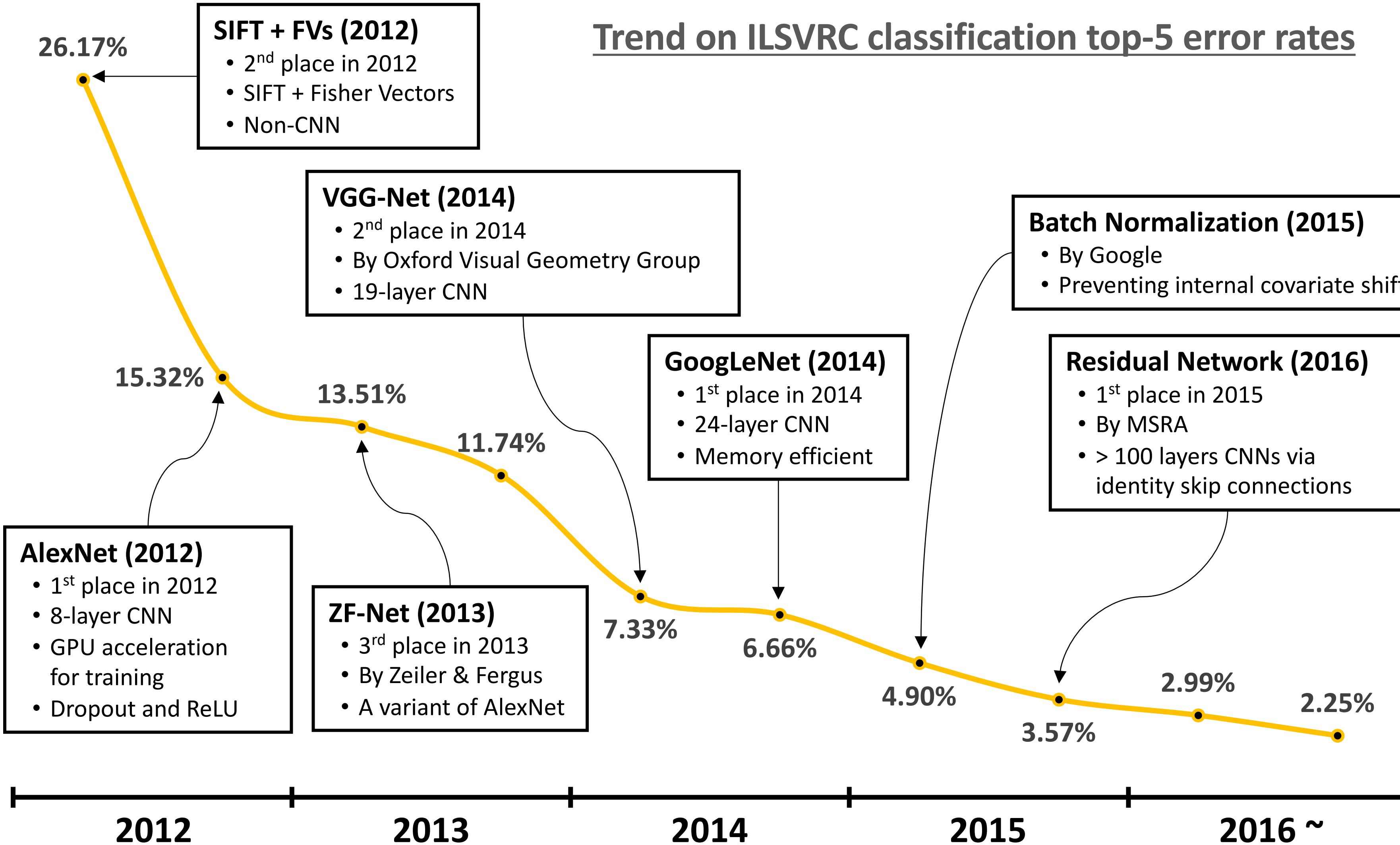
- ImageNet dataset: a large database of visual objects
    - ~14M labeled images, 20K classes
    - Human labels via Amazon MTurk
- Classification: 1,281,167 images for training / 1,000 categories
- Annually ran from 2010 to 2017, and now hosted by Kaggle
- For details, see [Russakovsky et al., 2015]



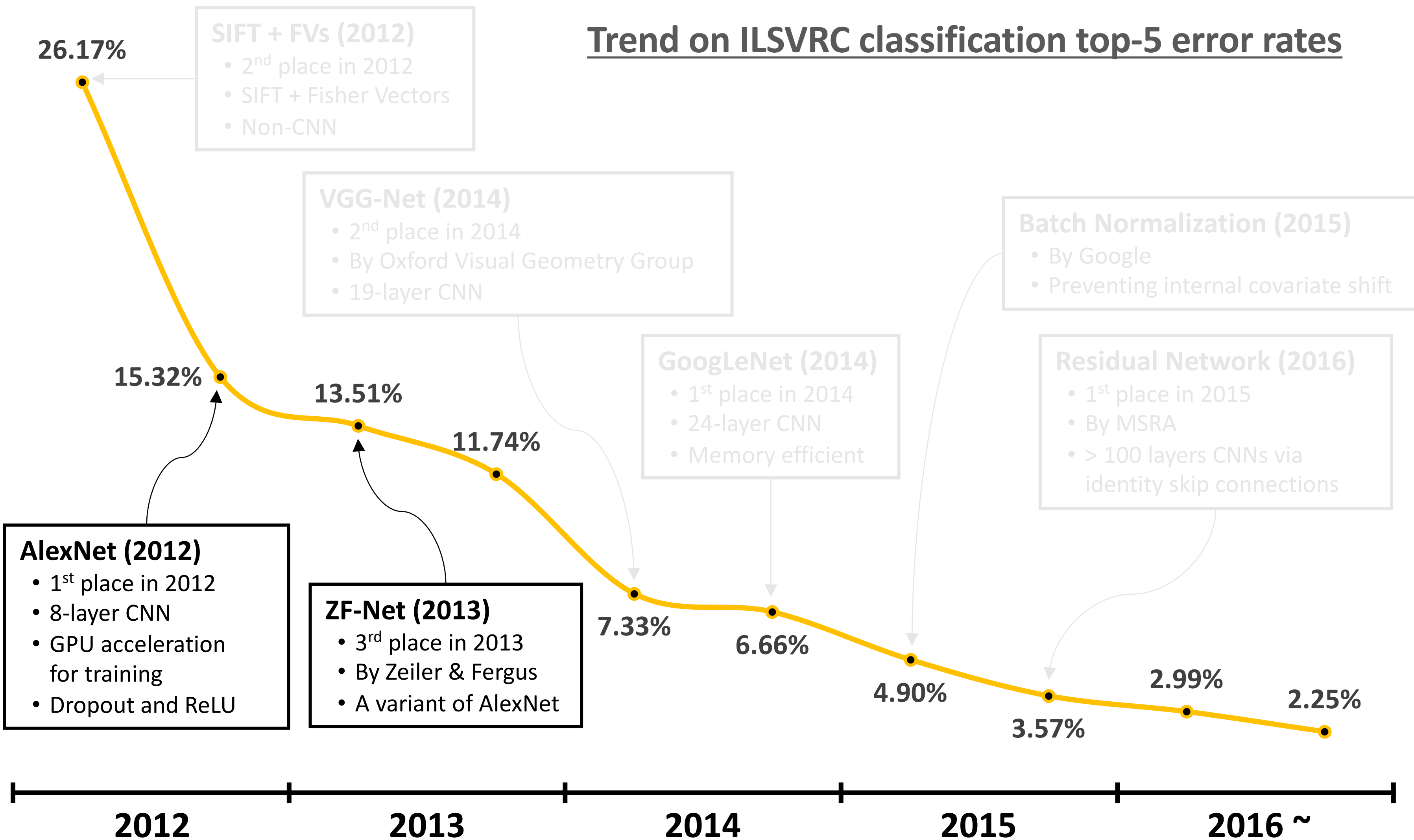
*source : http://visgraph.cse.ust.hk/ilsvrc/files/tesor.png

Evolution of CNN Architectures

ILSVRC contributed greatly to development of CNN architectures



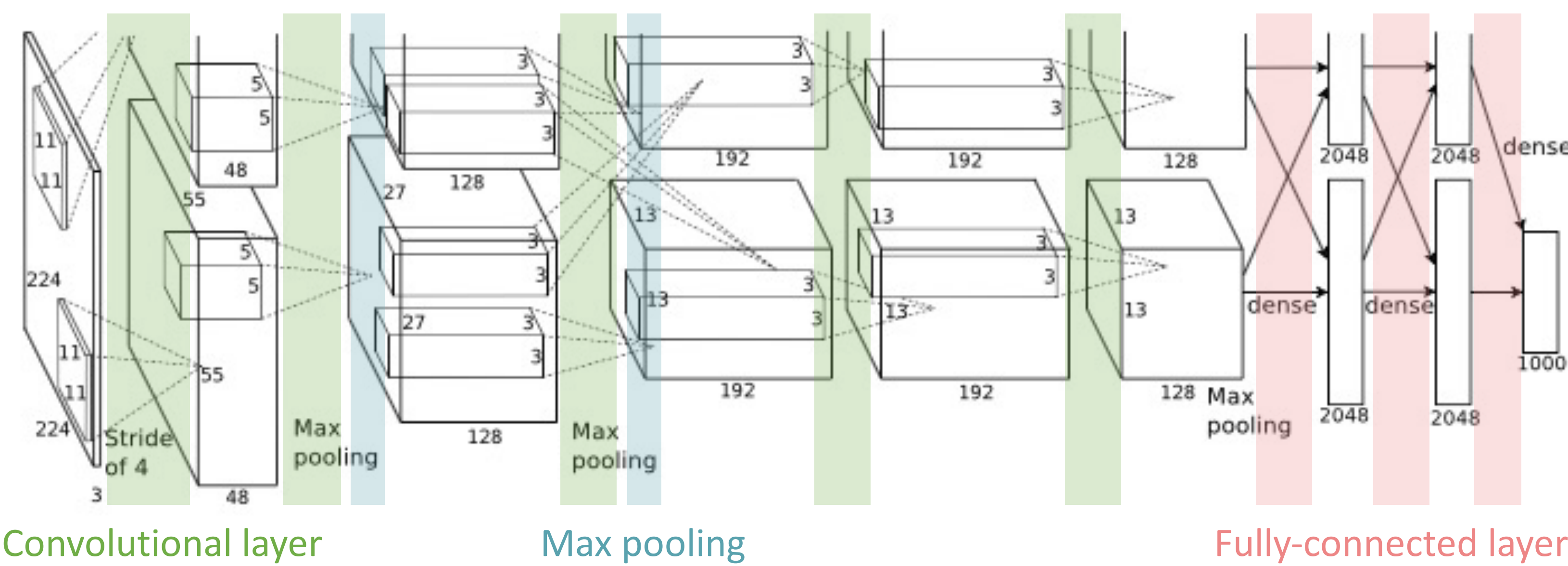
ILSVRC contributed greatly to development of CNN architectures



The first winner to use CNN in ILSVRC, with an astounding improvement

- Top-5 error is largely improved: 25.8% → **15.3%**
- The 2nd 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



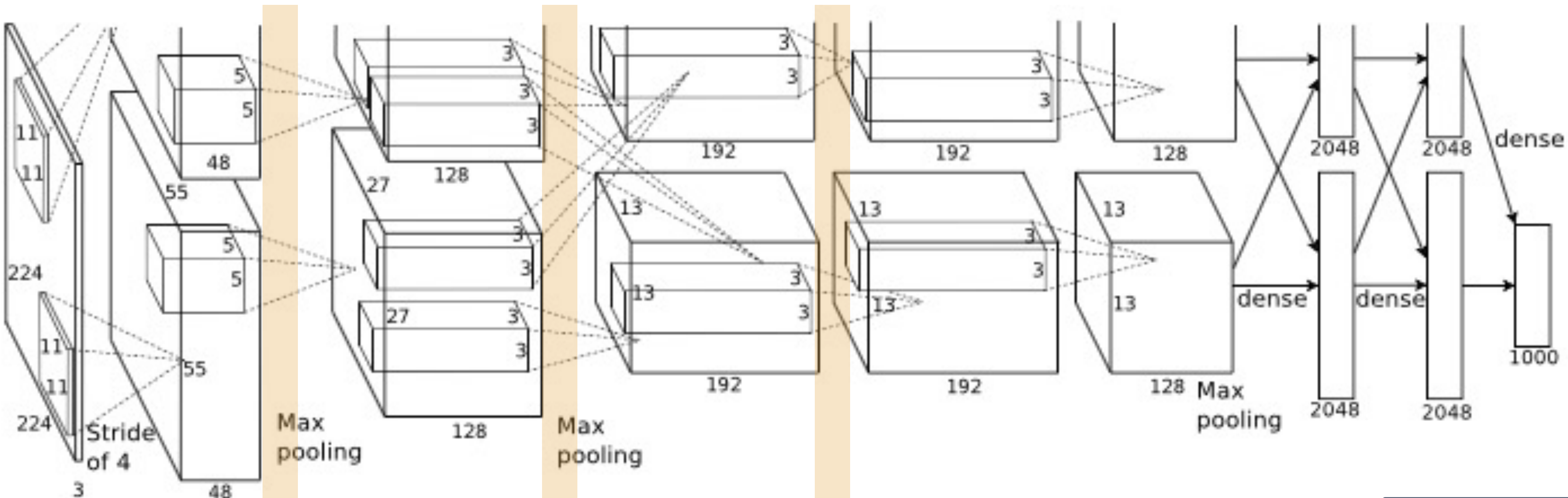
*source : Krizhevsky et al. "Imagenet classification with deep convolutional neural networks". NIPS 2012

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

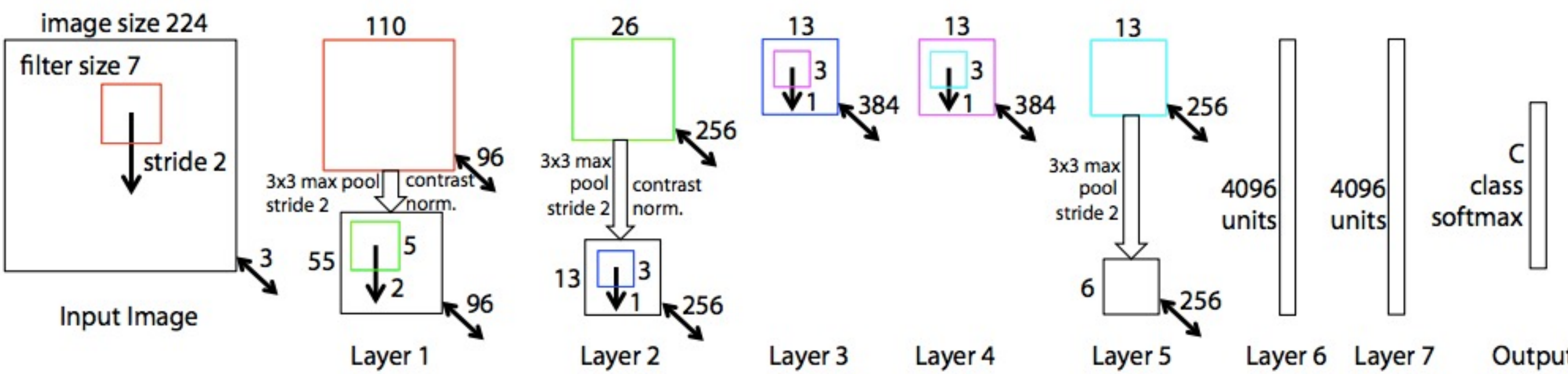
*source : Krizhevsky et al. “Imagenet classification with deep convolutional neural networks”. NIPS 2012

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

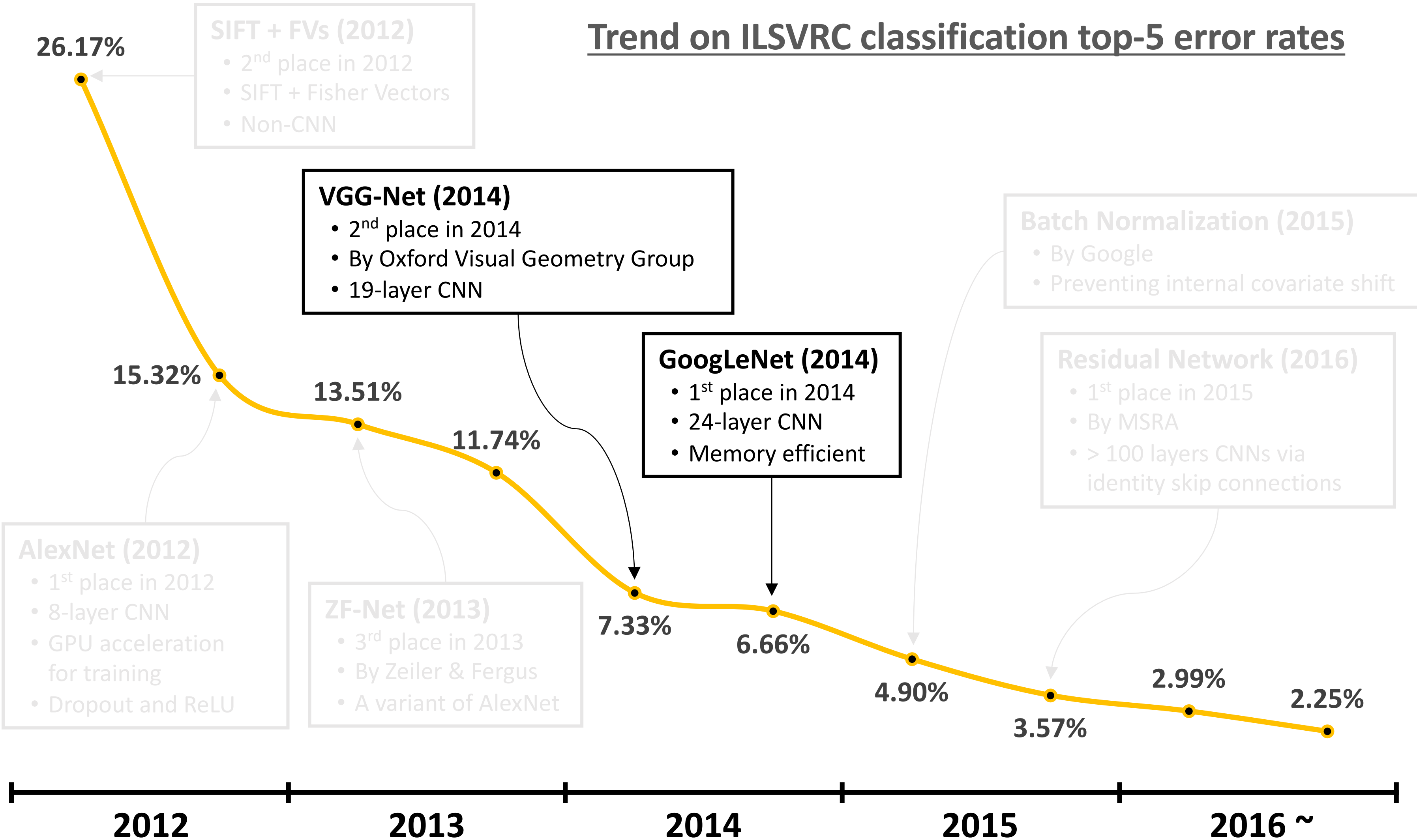
Lessons

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



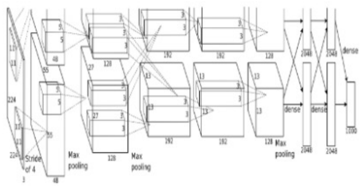
Networks were getting deeper

- AlexNet: 8 layers
- VGGNet: **19** layers
- GoogleNet: **24** layers

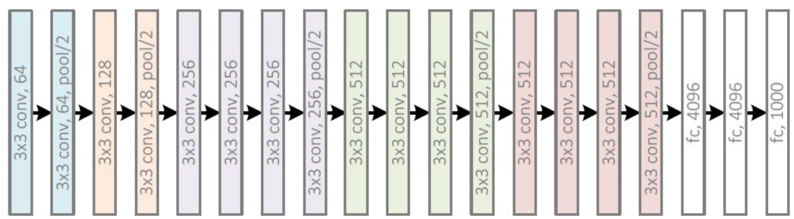
Both focused on parameter efficiency of each block

- Mainly to allow larger networks computable at that time

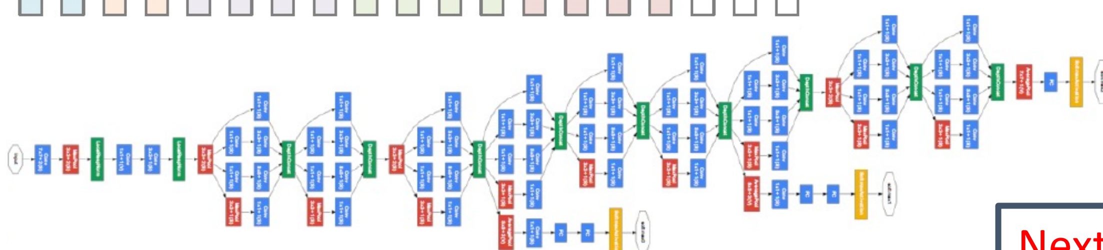
AlexNet



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

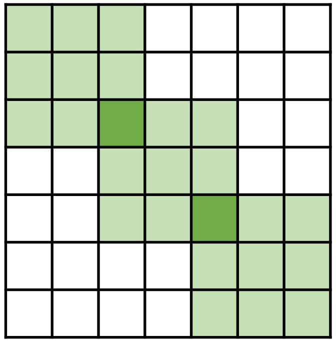
The 2nd place in ILSVRC'14 (11.7% → 7.33%)

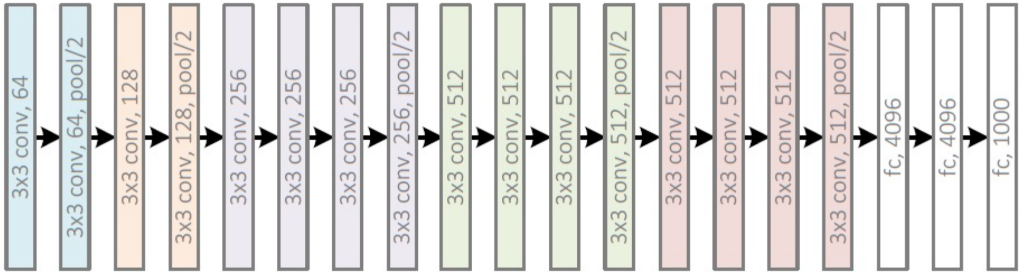
- Designed using only 3×3 kernels for convolutions

Lesson: **Stacking multiple 3×3** is advantageous than using other kernels

Example: $((3\times3)\times3)$ v.s. (7×7)

- Essentially, they get the same receptive field
- $((3\times3)\times3)$ have less # parameters
  - $3\times\left(C\times\left((3\times3)\times C\right)\right) = \mathbf{27C^2}$
  - $C\times\left((7\times7)\times C\right) = \mathbf{49C^2}$
- $((3\times3)\times3)$ gives more non-linearities



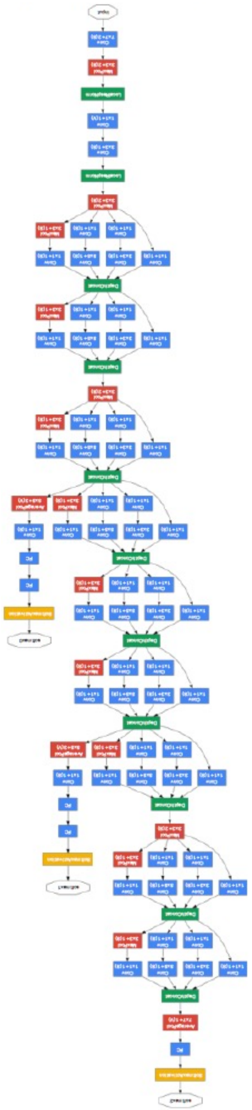


Next, GoogLeNet

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

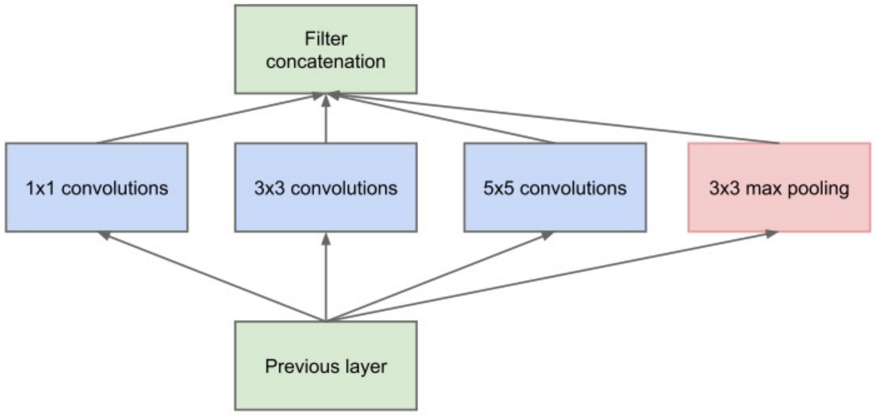
The winner of ILSVRC’14 (11.7% → 6.66%)

- 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

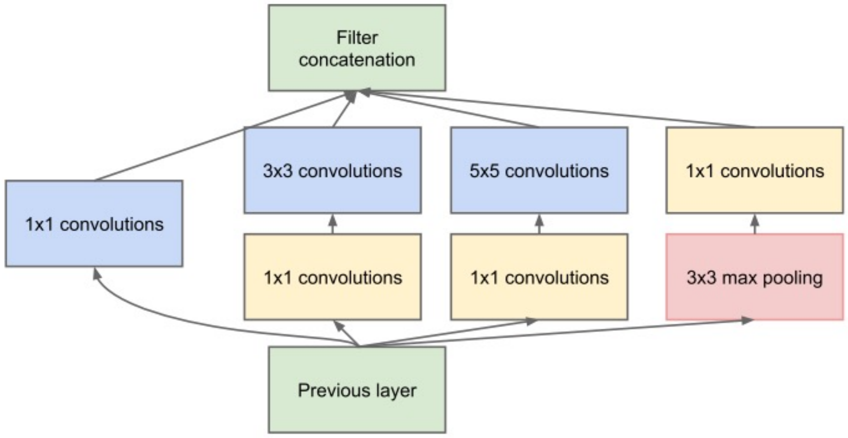
*source : Szegedy et al., “Going deeper with convolutions”. CVPR 2015

The winner of ILSVRC'14 (11.7% → 6.66%)

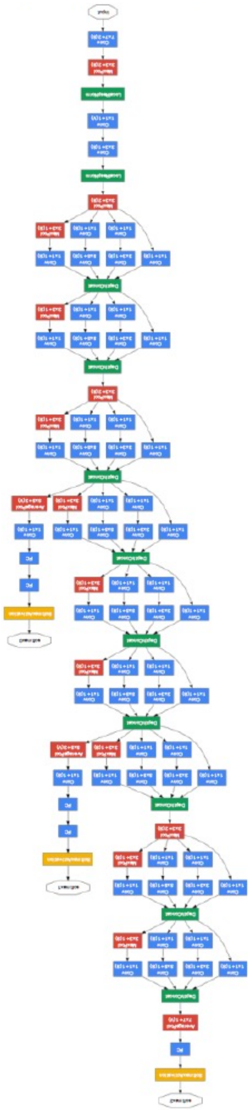
- Achieved 12× fewer parameters than AlexNet

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



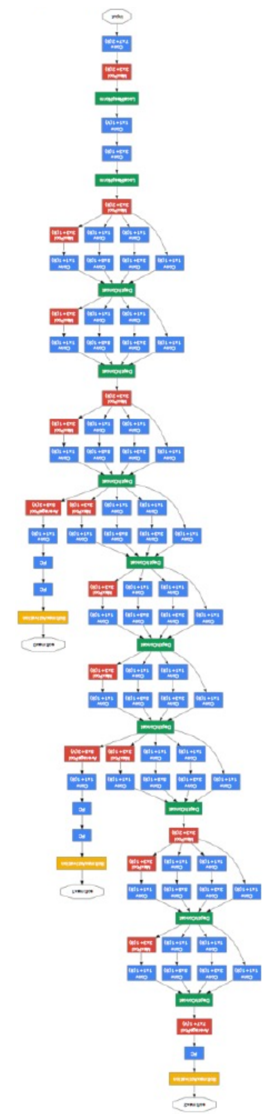
(b) Inception module with dimensionality reduction



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

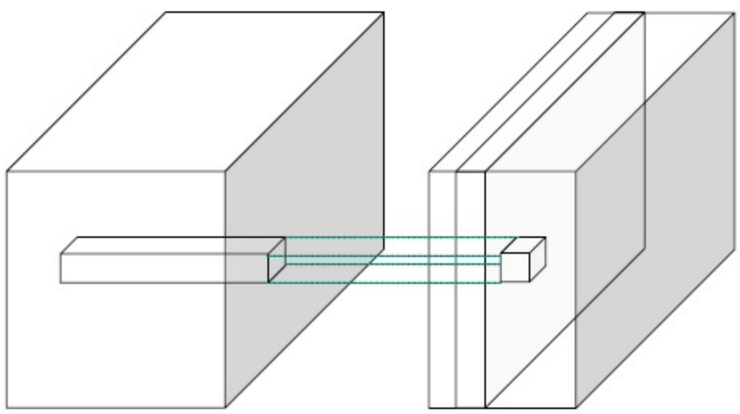
The winner of ILSVRC'14 (11.7% → 6.66%)

- Achieved 12× fewer parameters than AlexNet



cf. 1 × 1 convolutions

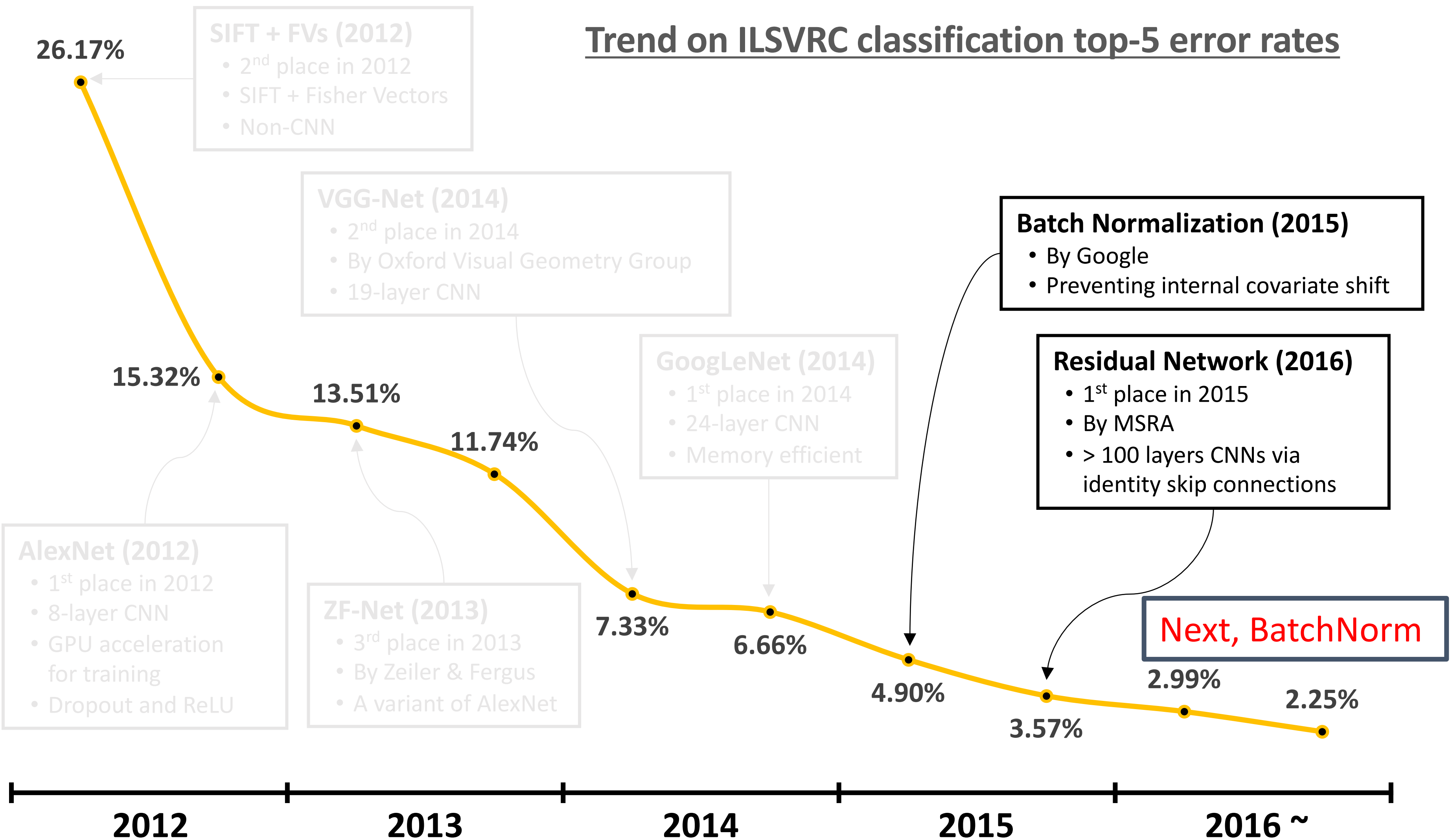
- Linear transformation done in pixel-wise
- Can be represented by a matrix
- Useful for changing # channels efficiently



*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



Part 1. Basics

- Evolution of CNN architectures
- Batch normalization and ResNet

Part 2. Advanced Topics

- Toward automation of network design
- Dilated and Deformable convolution
- Attention module in CNNs
- Observational Study on CNN Architectures

Part 3. Beyond CNN

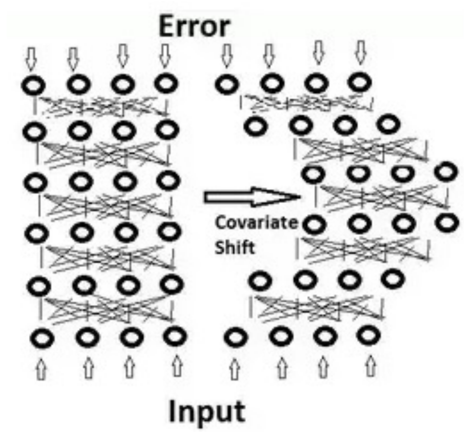
- Transformer architecture for Vision
- MLP architecture for Vision

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"

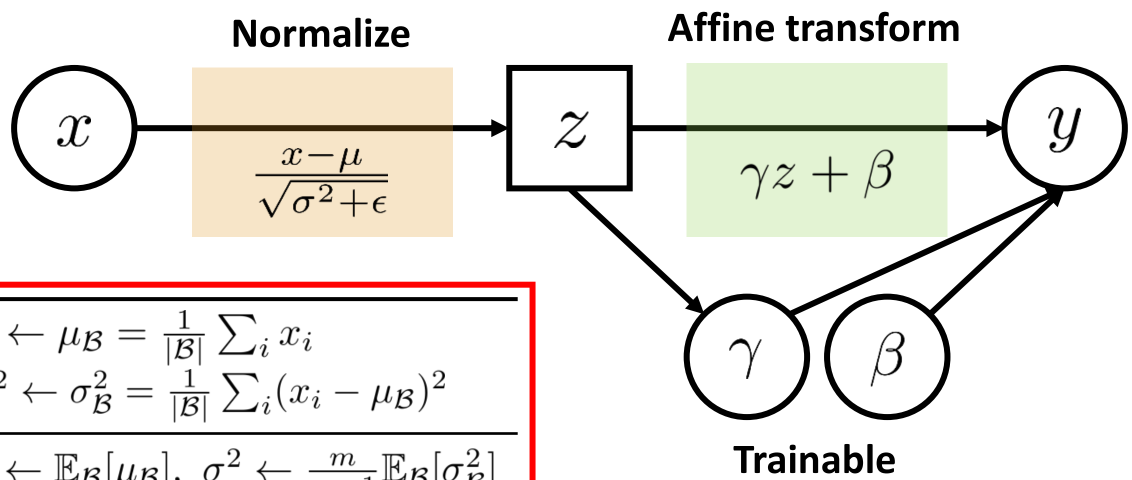


- Similar problem happens during training of deep neural networks
- Updates in early layers may shift the inputs of later layers too much

*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

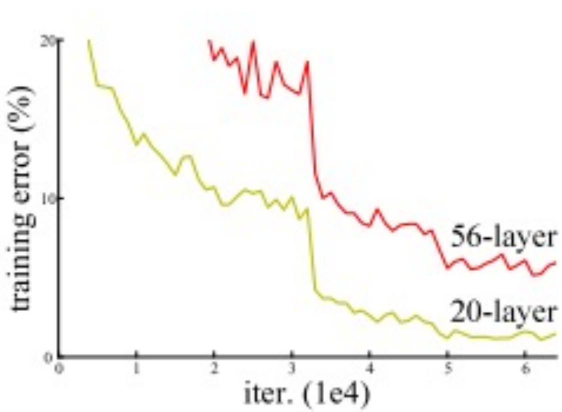
*source : Ioffe et al., “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. ICML 2015

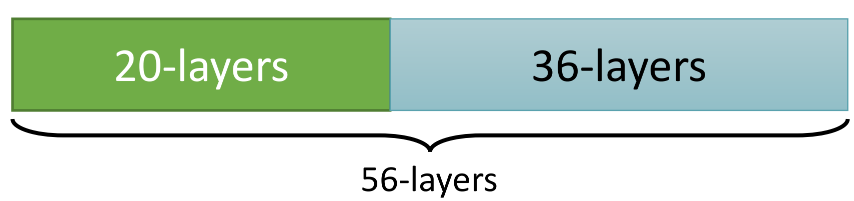
The winner of ILSVRC'15 (6.66% → 3.57%)

- **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?





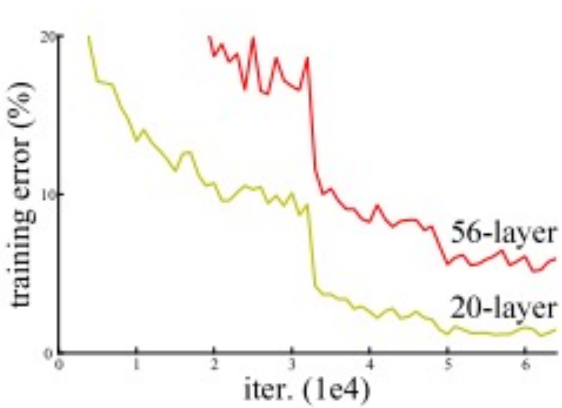
*source : He et al., "Deep residual learning for image recognition". CVPR 2016

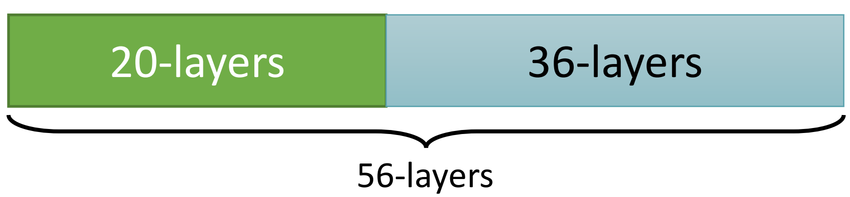
The winner of ILSVRC'15 (6.66% → 3.57%)

- **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





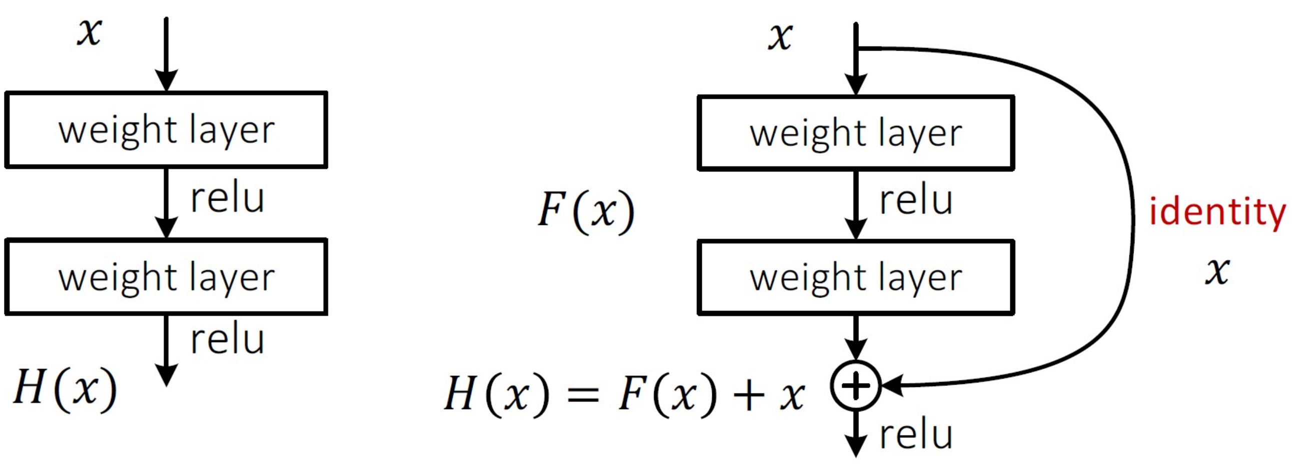
*source : He et al., "Deep residual learning for image recognition". CVPR 2016

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*

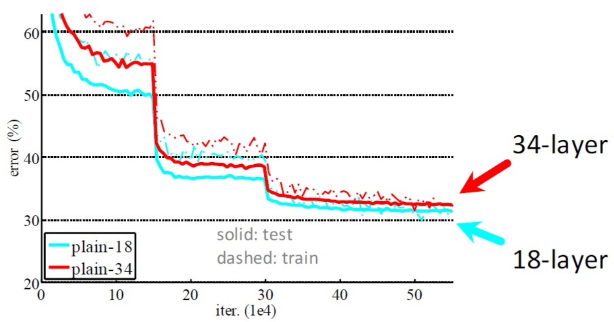
- When all the weights are set to zero, the layer represents an identity



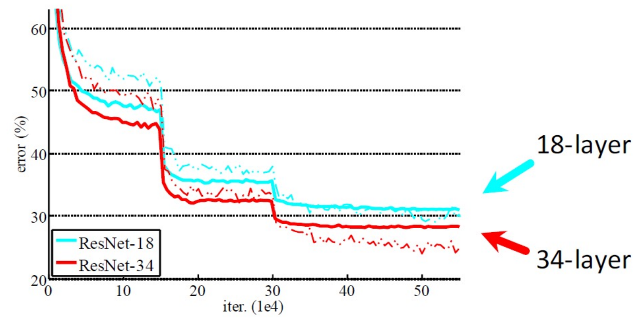
*source : He et al., "Deep residual learning for image recognition". CVPR 2016

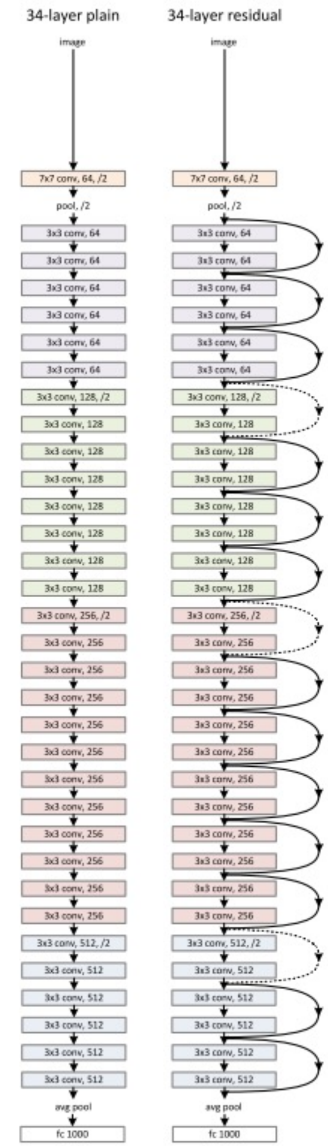
Plain nets v.s. ResNets

ImageNet plain nets

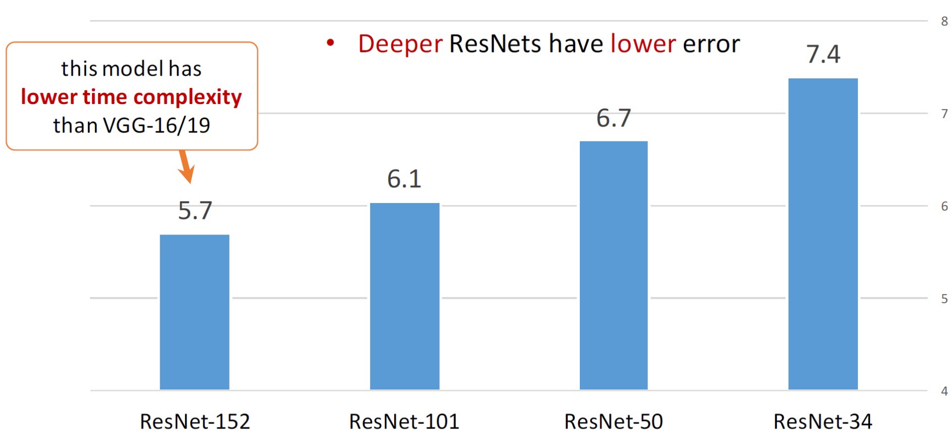


ImageNet ResNets





- Deeper ResNets can be trained without any difficulty



10-crop testing, top-5 val error (%)

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

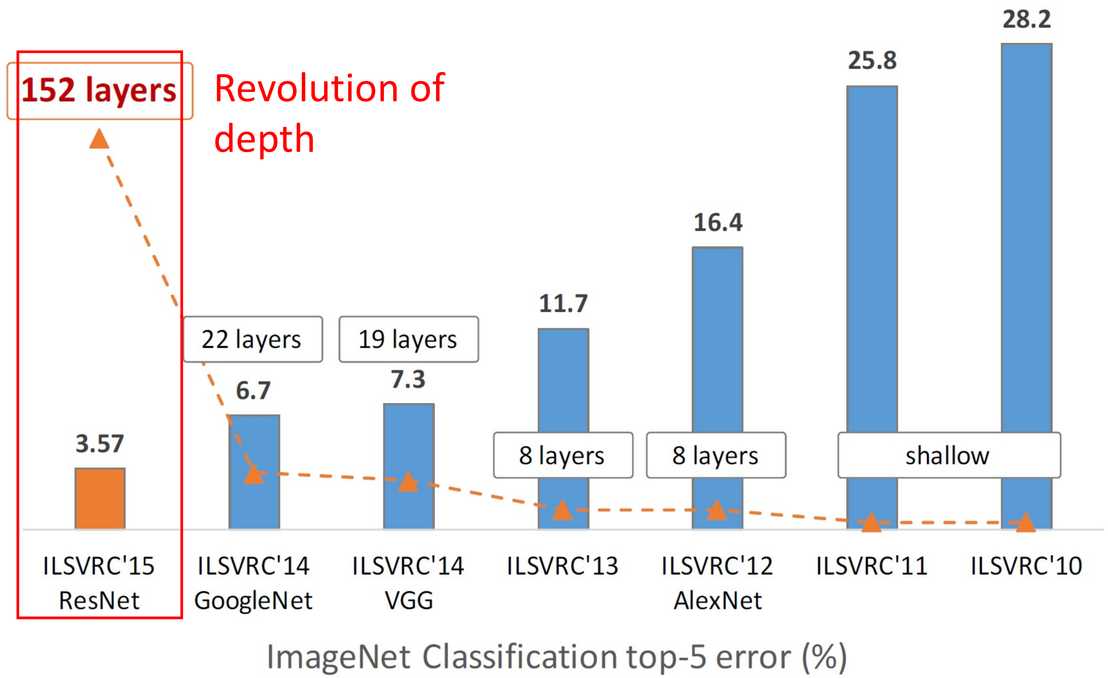
ResNet [He et al., 2016a]

- Identity connection resolved a major difficulty on optimizing large networks

Revolution of depth: Training >100-layer network without difficulty

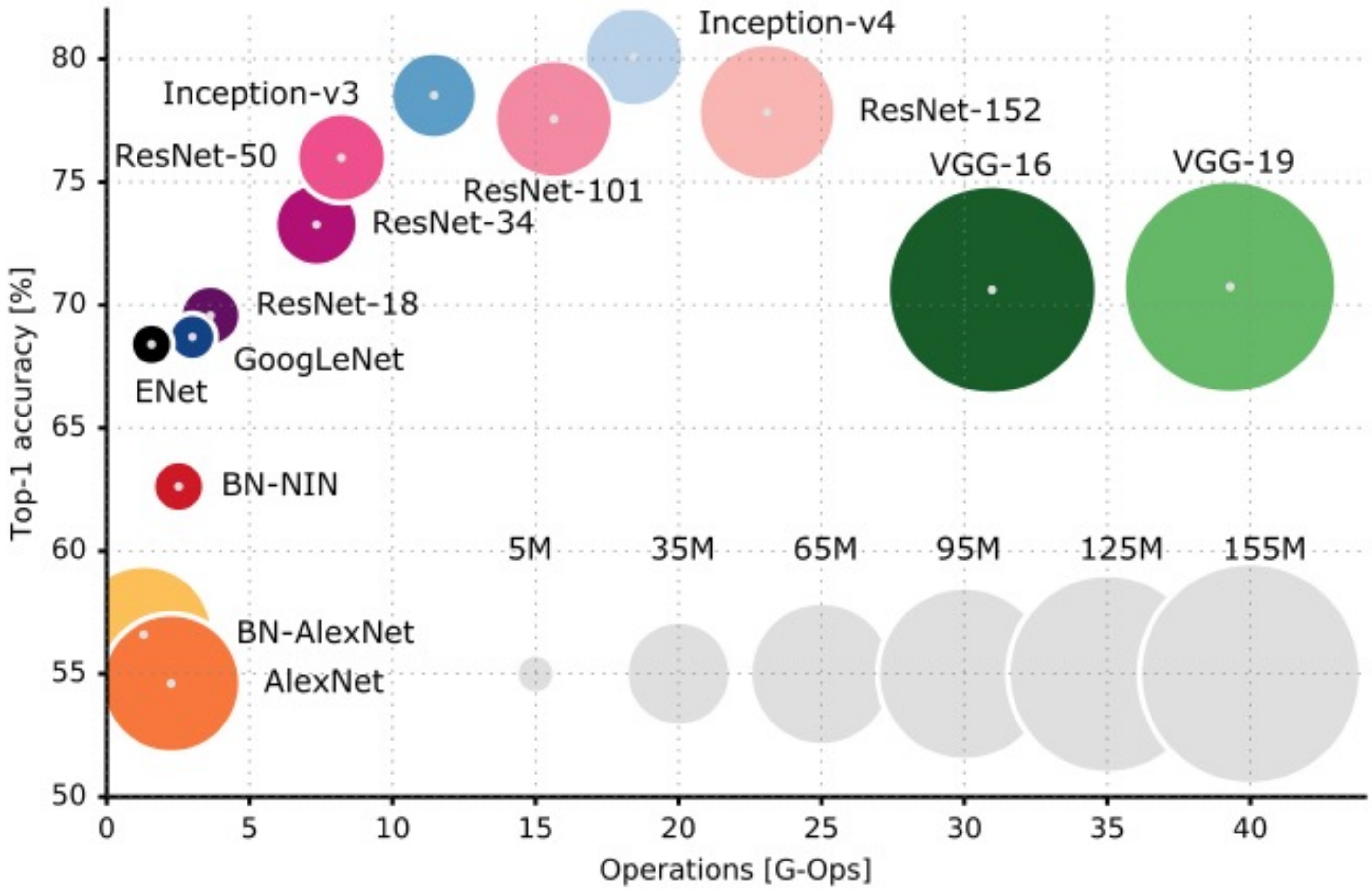
  - Later, ResNet is revised to allow to train up to >1000 layers [He et al., 2016b]

- 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

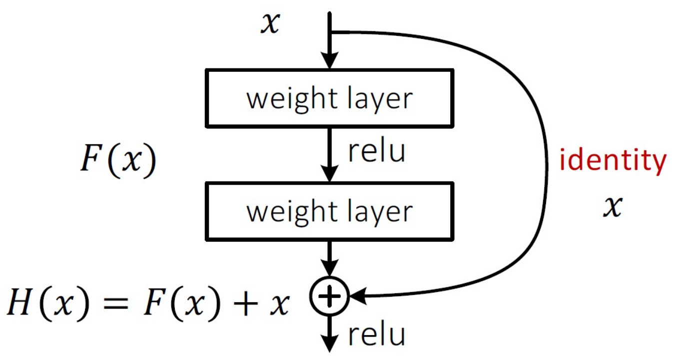
Comparisons on ImageNet for a single model of popular CNNs



*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?”

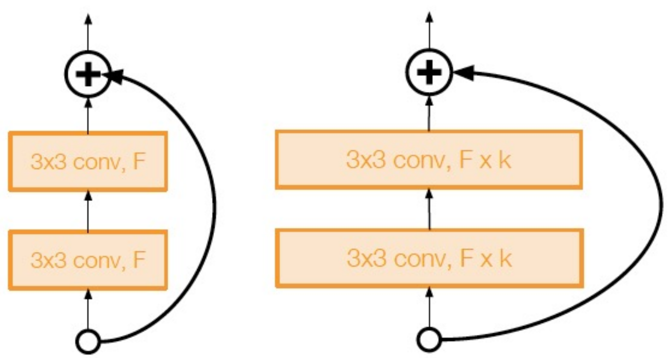
*source : He et al., “Deep residual learning for image recognition”. CVPR 2016

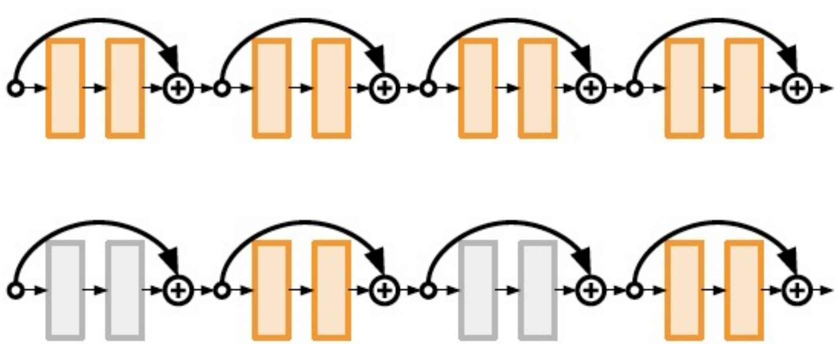
Wide Residual Networks [Zagoruyko et al., 2016]

- 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

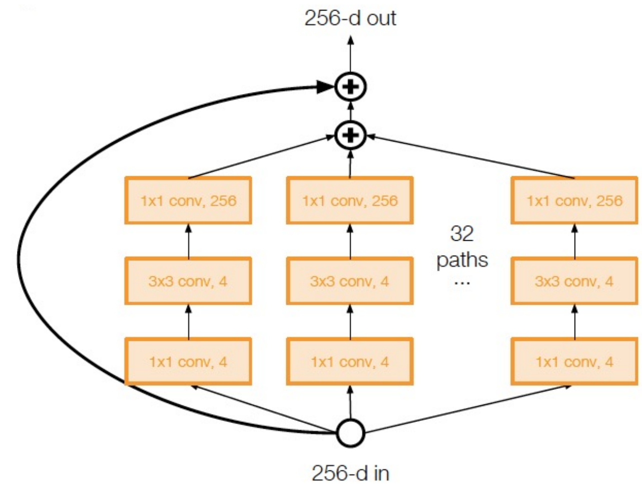




*source : Fei-Fei Li et al. (2018), CS231n Lecture 9, Stanford University

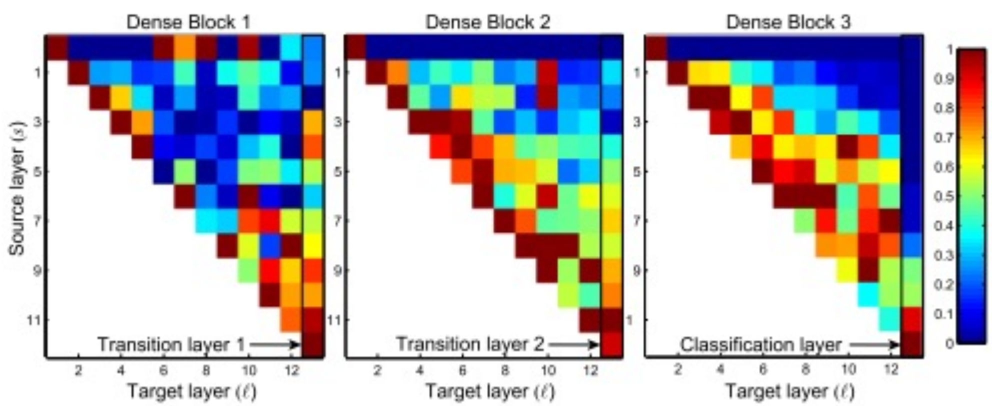
ResNeXt [Xie et al., 2016]

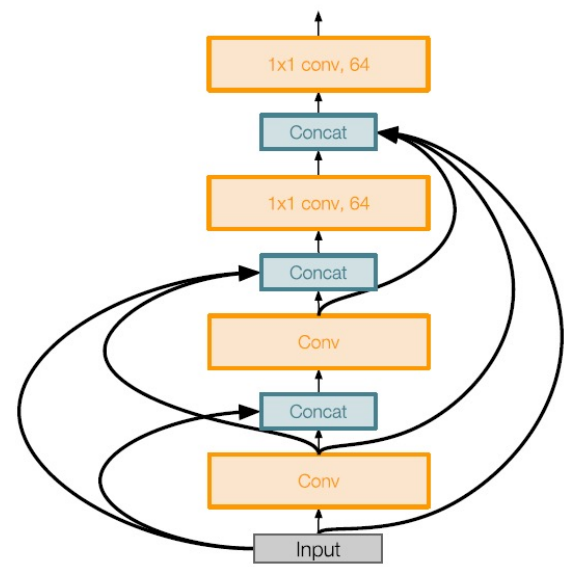
- Aggregating multiple parallel paths inside a residual block (“**cardinality**”)
- Increasing cardinality is more effective than going deeper or wider



DenseNet [Huang et al. 2017]

- Passing all the previous representation directly via concatenation of features
- Strengthens **feature propagation** and **feature reuse**





Dense Block

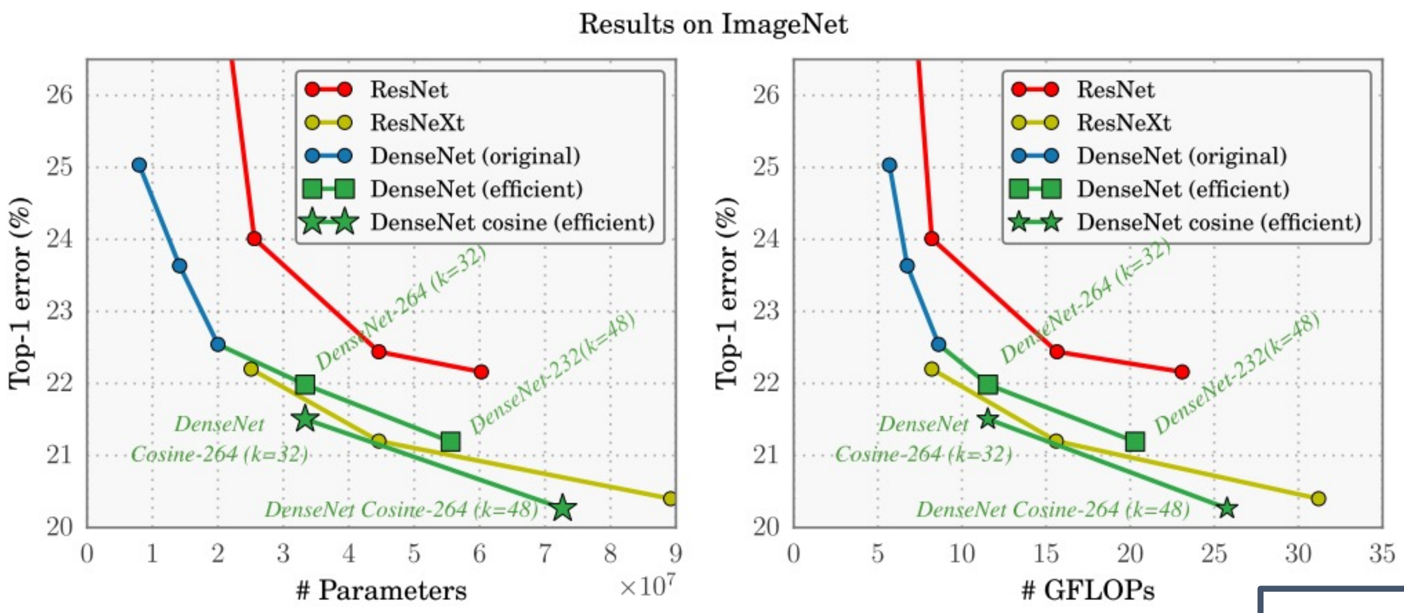
*source : Fei-Fei Li et al. (2018), CS231n Lecture 9, Stanford University

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, NASNet

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

Part 1. Basics

- Evolution of CNN architectures
- Batch normalization and ResNet

Part 2. Advanced Topics

- Toward automation of network design
- Dilated and Deformable convolution
- Attention module in CNNs
- Observational Study on CNN Architectures

Part 3. Beyond CNN

- Transformer architecture for Vision
- MLP architecture for Vision

Table of Contents

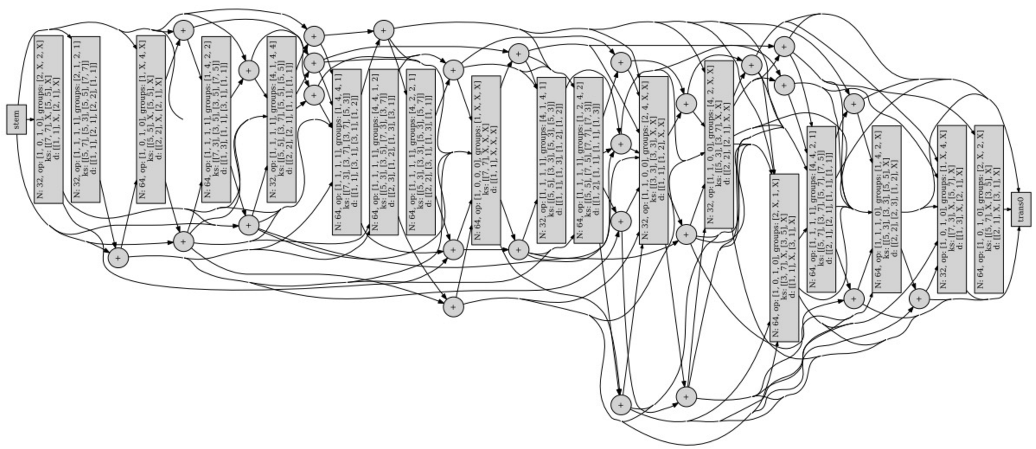
Toward Automation of Network Design

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]

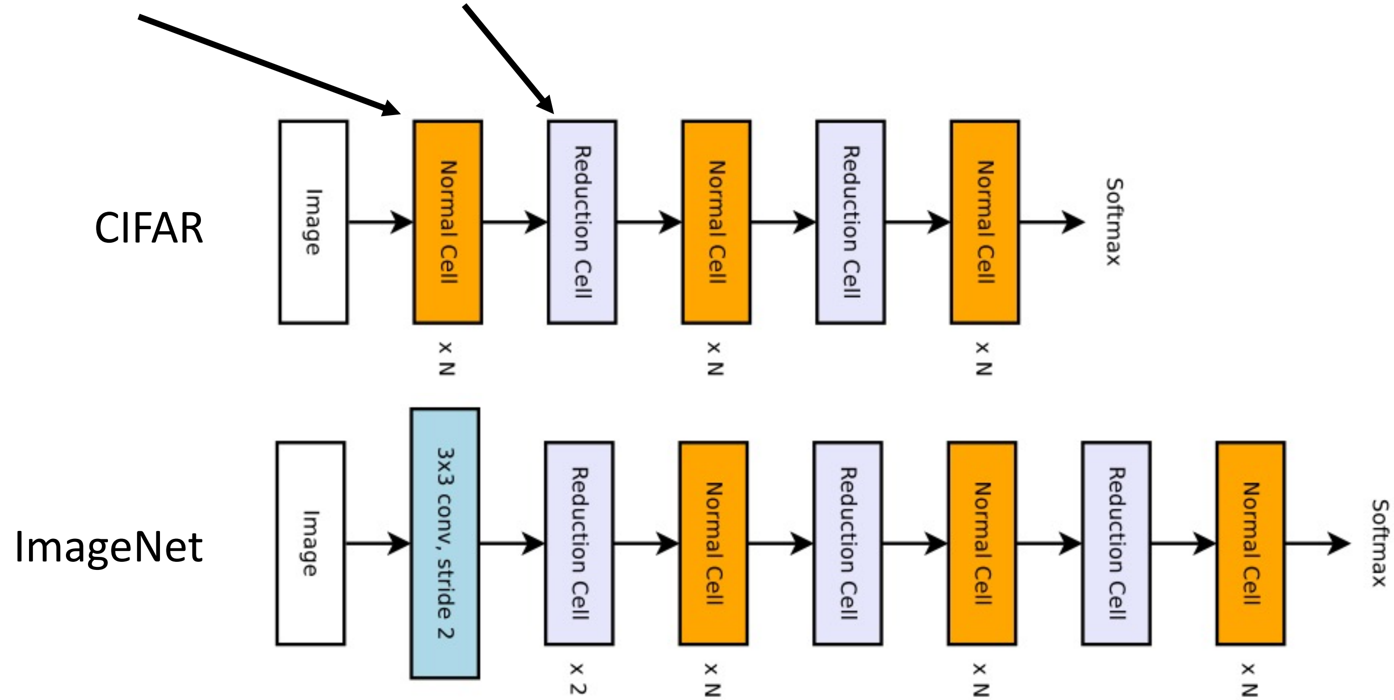
*source : Brock et al., “SMASH: One-Shot Model Architecture Search through HyperNetworks”, ICLR 2018

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)

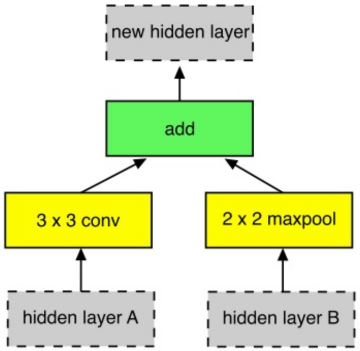


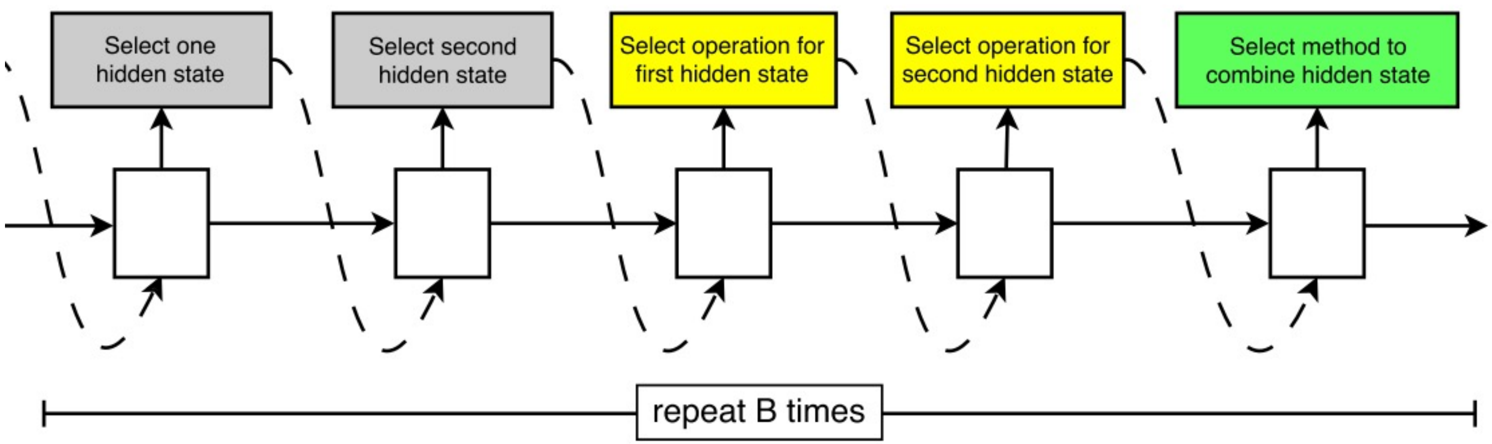
*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

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**



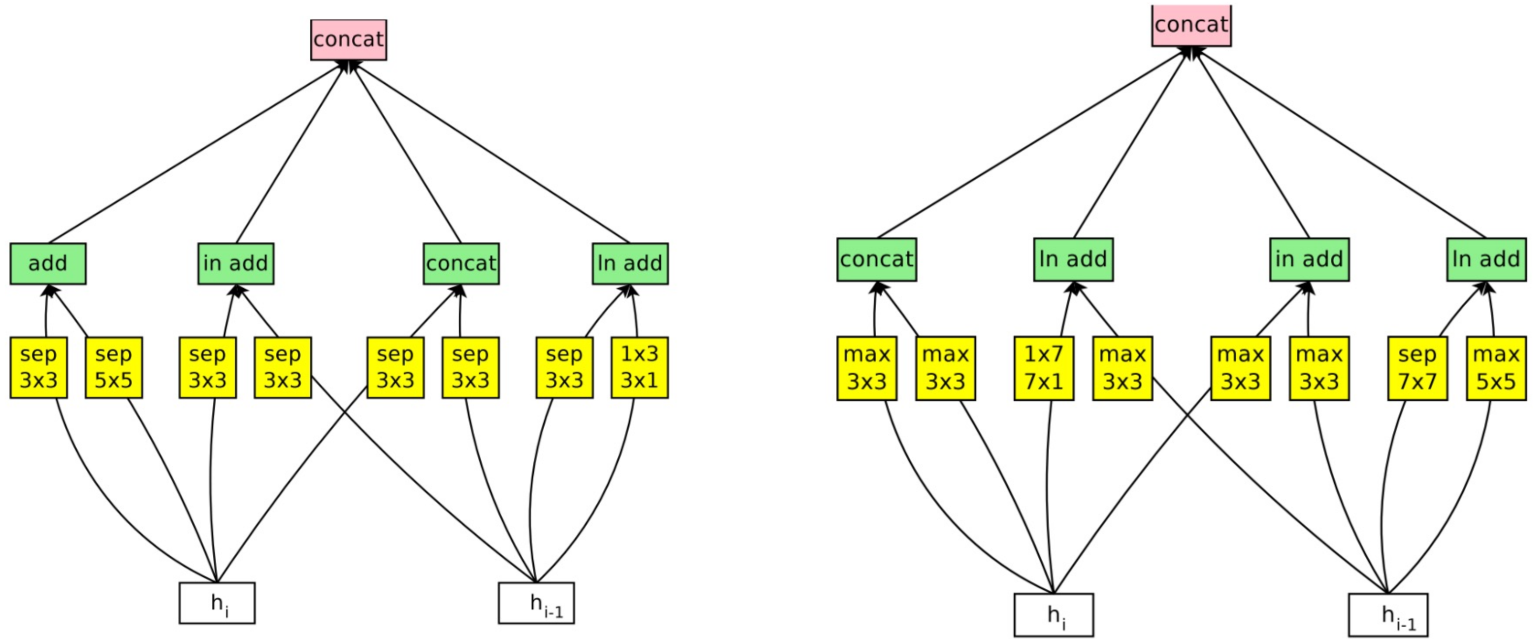


*source : Zoph et al., "Learning Transferable Architectures for Scalable Image Recognition", CVPR 2018

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

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

- 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

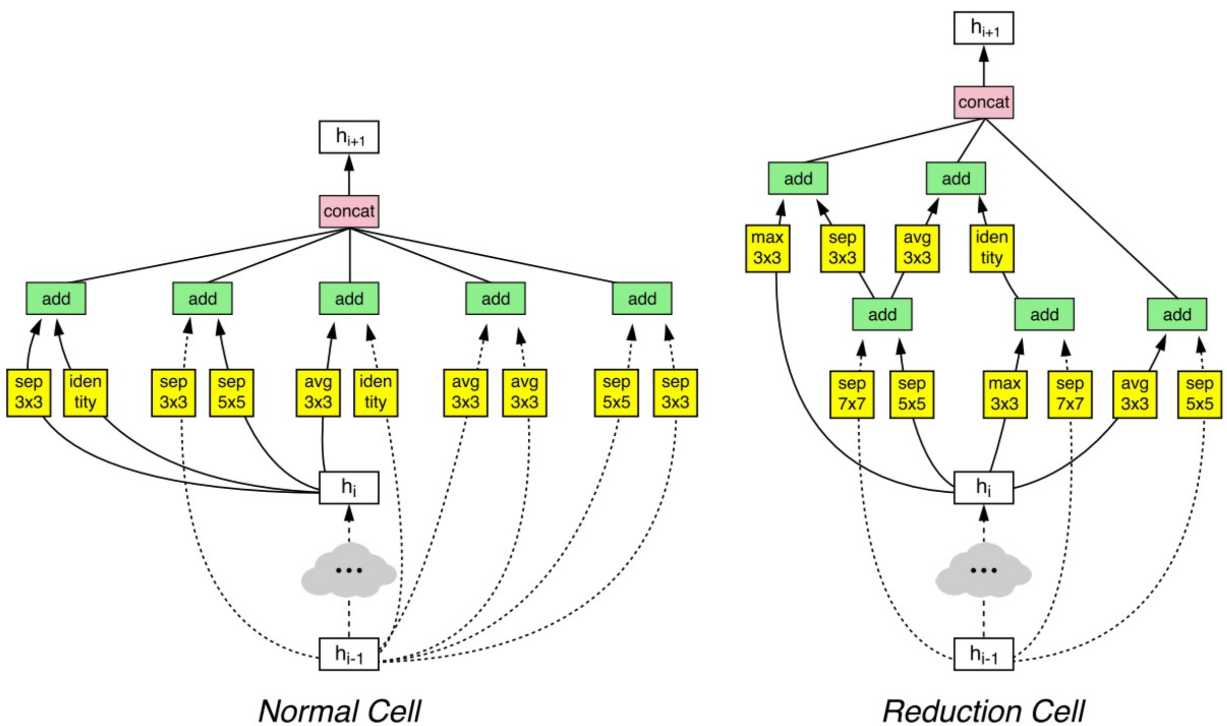
*source : Zoph et al., “Learning Transferable Architectures for Scalable Image Recognition”, CVPR 2018

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

- 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

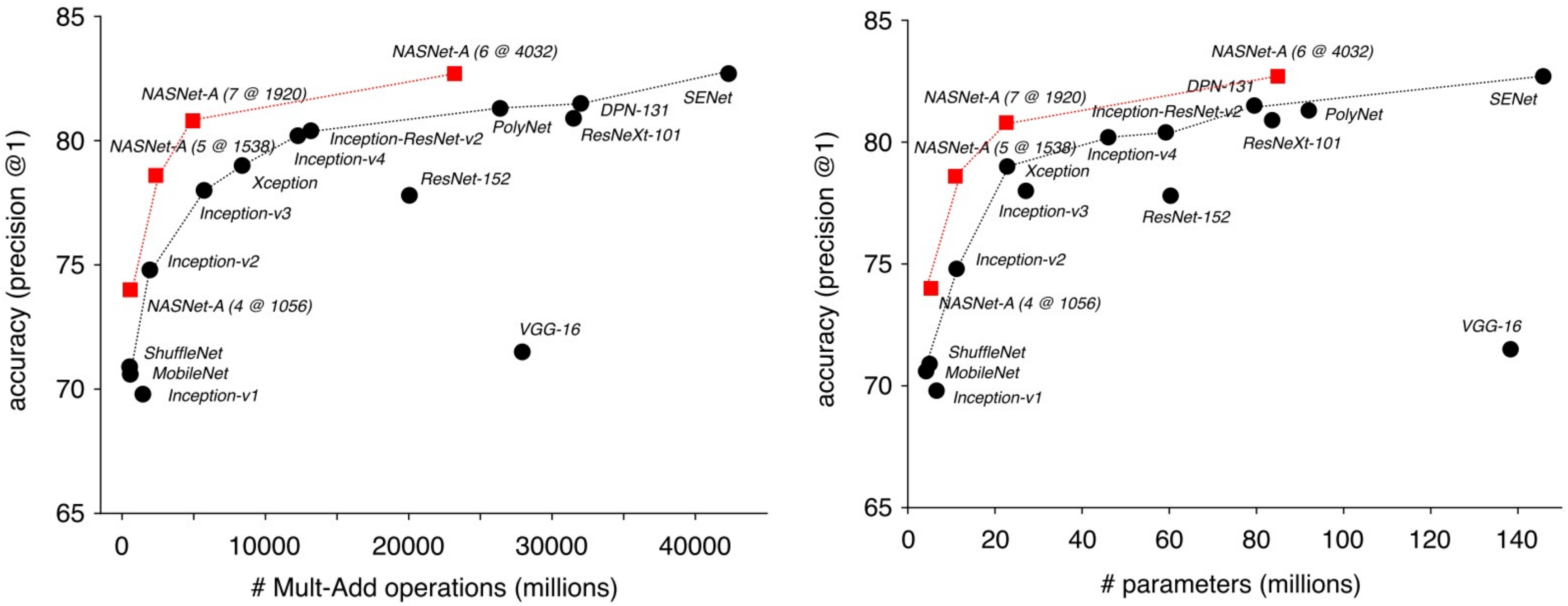
Model	image size	# parameters	Mult-Adds	Top 1 Acc. (%)	Top 5 Acc. (%)
Inception V2 [29]	224×224	11.2 M	1.94 B	74.8	92.2
NASNet-A (5 @ 1538)	**299×299**	**10.9 M**	**2.35 B**	**78.6**	**94.2**
Inception V3 [59]	299×299	23.8 M	5.72 B	78.0	93.9
Xception [9]	299×299	22.8 M	8.38 B	79.0	94.5
Inception ResNet 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

- 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

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

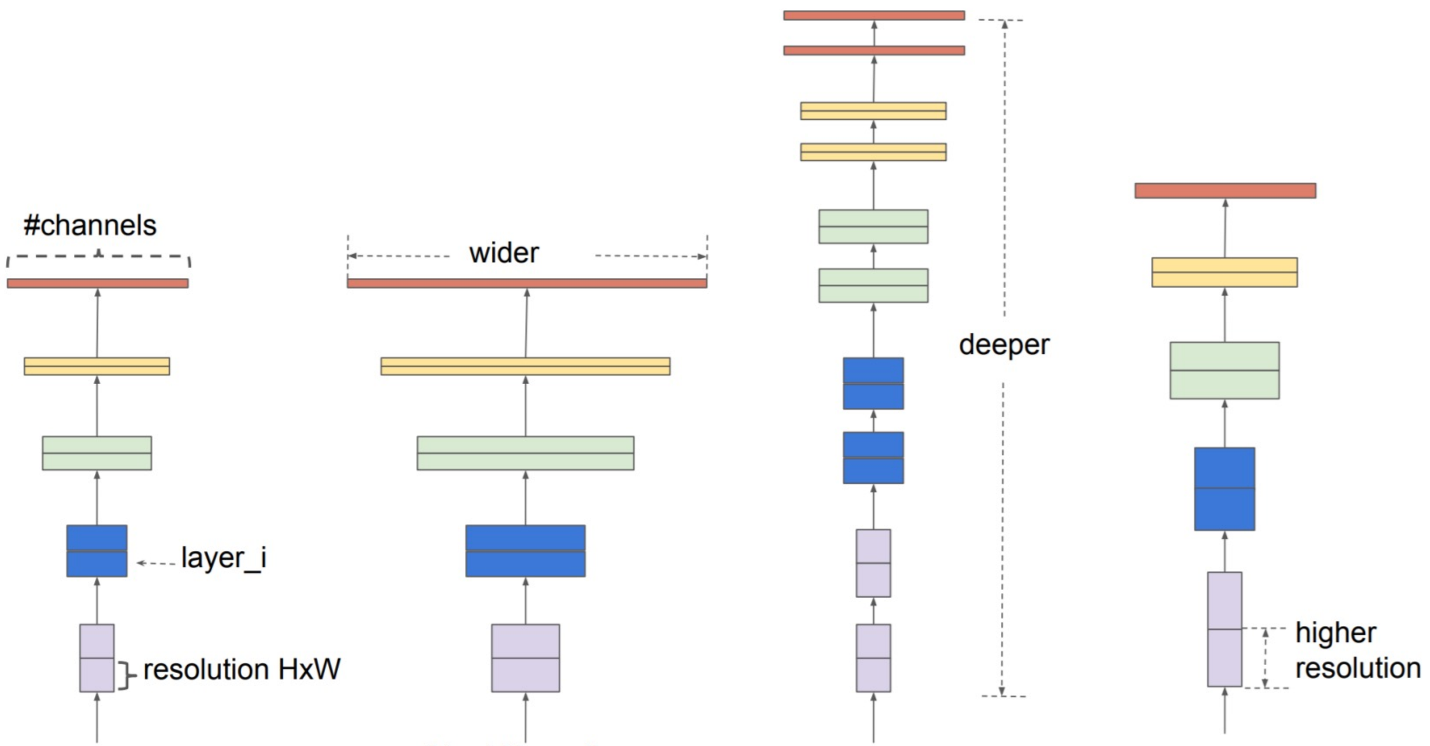
Next, EfficientNet

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

Although **Scaling up** CNNs is widely used to achieve better generalization, the process of scaling has never been understood

- The common way is scaling model depth, width, and image resolution

Question: Is there a principled scaling method for better accuracy and efficiency?



(a) baseline (b) width scaling (c) depth scaling (d) resolution scaling

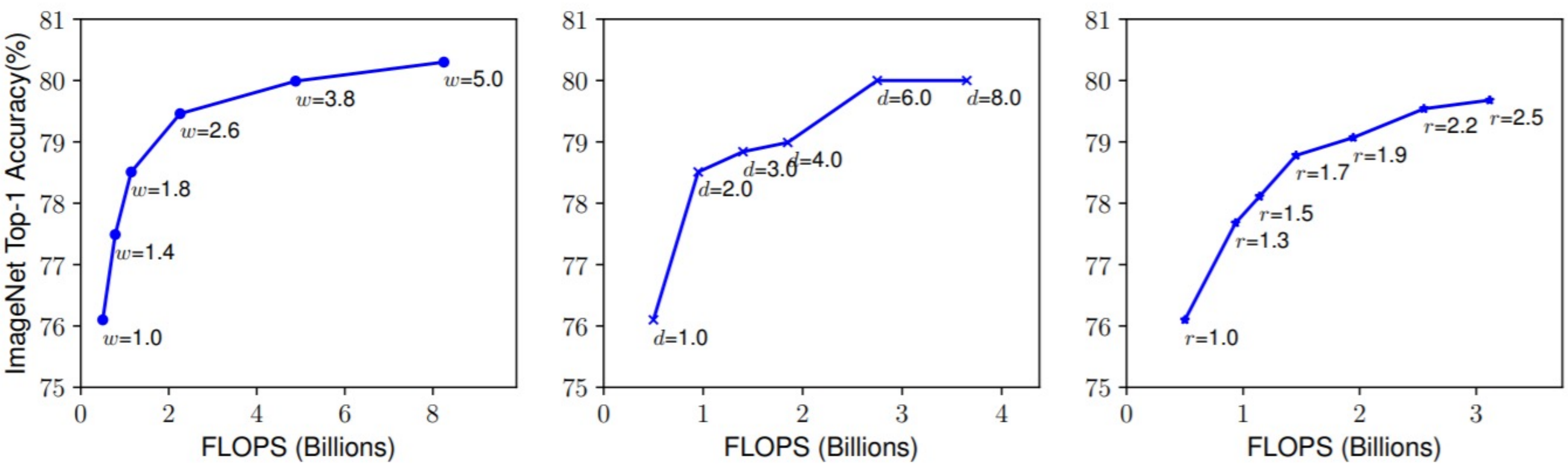
*source : Tan et al., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks", ICML 2019

The state-of-the-art ILSVRC classification in 2019 (top-5 error rate 2.9%)

- **EfficientNet** uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients (called ***"compound scaling"***)

Motivation: There exists certain relationship between network width, depth and image resolution

- Scaling single dimension has a limitation
    - Gain diminishes for bigger models.

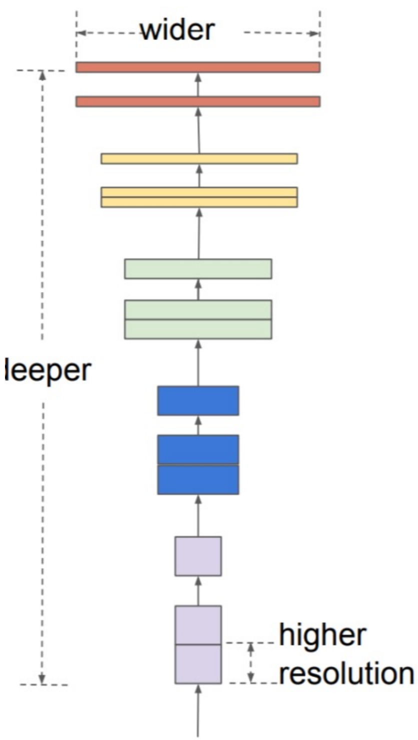


Depth w **Width d** **Resolution r**

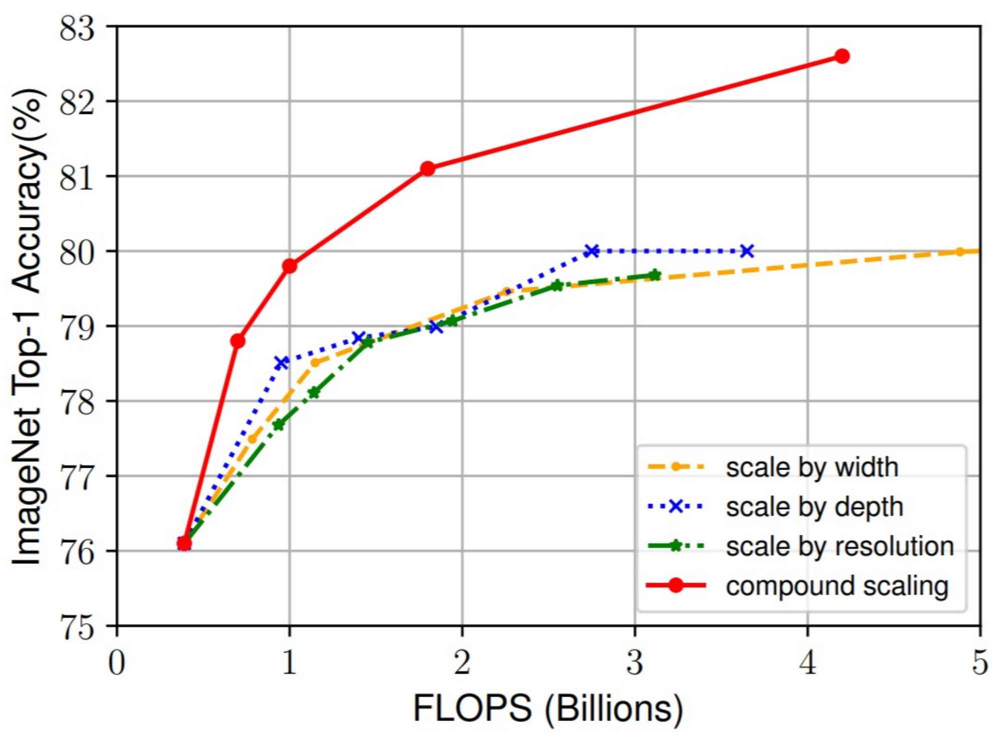
- Scaling all together with a fixed ratio

*source : Tan et al., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks", ICML 2019

- **Compound scaling:** Scaling all together with a fixed ratio ϕ in a principled way
  - Depth $d = \alpha^{\phi}, \alpha \geq 1$
  - Width $w = \beta^{\phi}, \beta \geq 1$
  - Resolution $r = \gamma^{\phi}, \gamma \geq 1$
  - Finding α, β, γ under compound constraint $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$
    - **Why?** Such scaling approximately increases total FLOPS by $(\alpha \cdot \beta^2 \cdot \gamma^2)^{\phi} \approx 2^{\phi}$



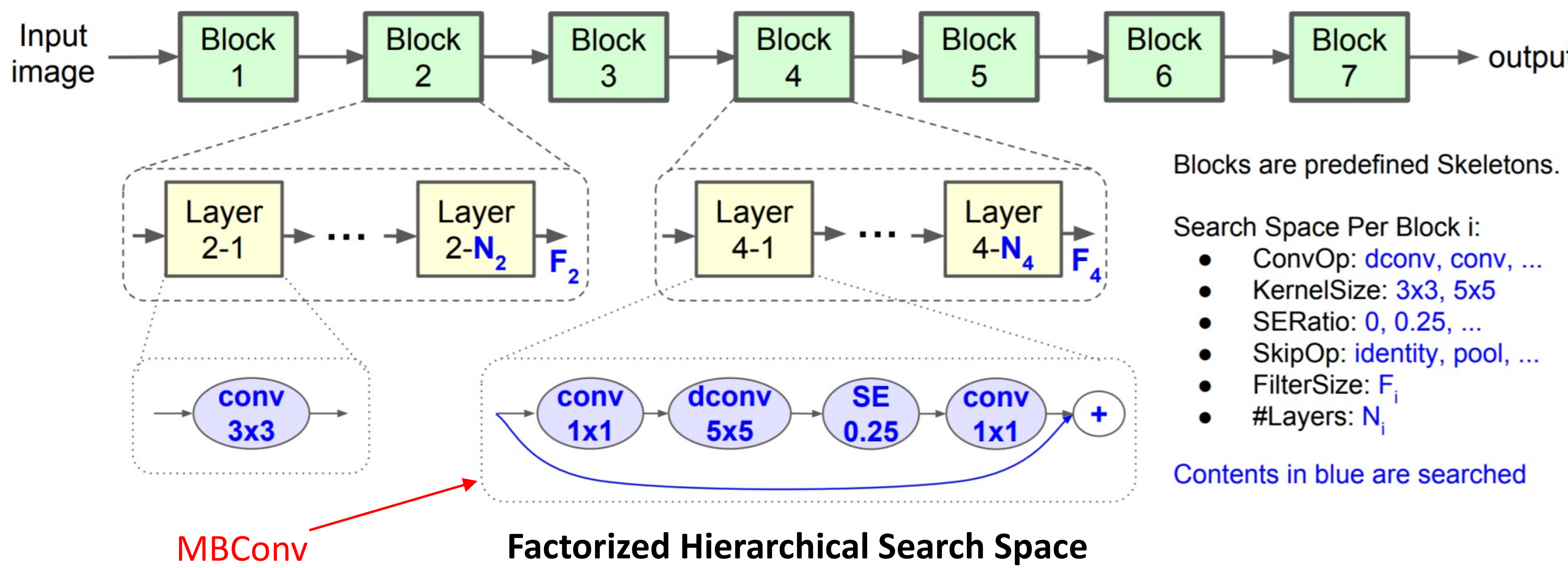
(e) compound scaling



*source : Tan et al., “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”, ICML 2019

Having a good baseline network is also critical!

- Multi-objective neural architecture search
  - Optimizing both **accuracy** and **FLOPS**
  - Search space is the same as MnasNet [Tan et al., 2019]
- Mobile-size baseline, called **EfficientNet-B0**
  - Main building block is mobile inverted bottleneck, **MBConv**
  - Adding squeeze-and-excitation (SE) optimization [Hu et al., 2018]



Factorized Hierarchical Search Space

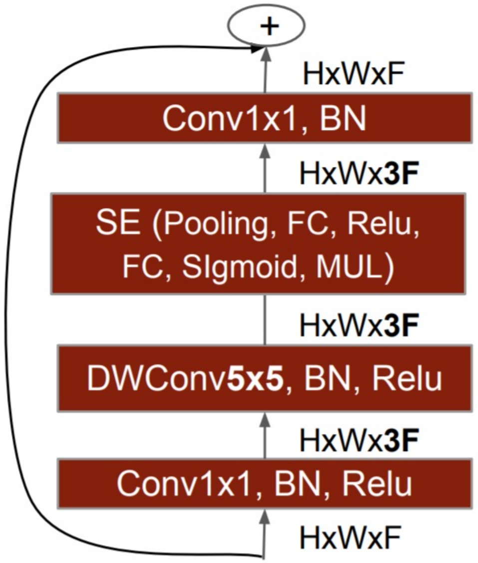
*source : Tan et al., "Mnasnet: Platform-aware neural architecture search for mobile", CVPR 2019

Having a good baseline network is also critical!

- Multi-objective neural architecture search
  - Optimizing both **accuracy** and **FLOPS**
  - Search space is the same as MnasNet [Tan et al., 2019]
- Mobile-size baseline, called **EfficientNet-B0**
  - Main building block is mobile inverted bottleneck, **MBConv**
  - Adding squeeze-and-excitation (SE) optimization [Hu et al., 2018]
  - DWConv denotes depthwise convolution [Howard et al ., 2017]

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Architecture of EfficientNet-B0



MBConv

*source : Tan et al., “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”, ICML 2019
Tan et al., “Mnasnet: Platform-aware neural architecture search for mobile”, CVPR 2019

From EfficientNet-B0 to B7

- **EfficientNet-B0**: Baseline model with $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$
- **EfficientNet-B1 to B7**: Scaling up EfficientNet-B0 with different ϕ

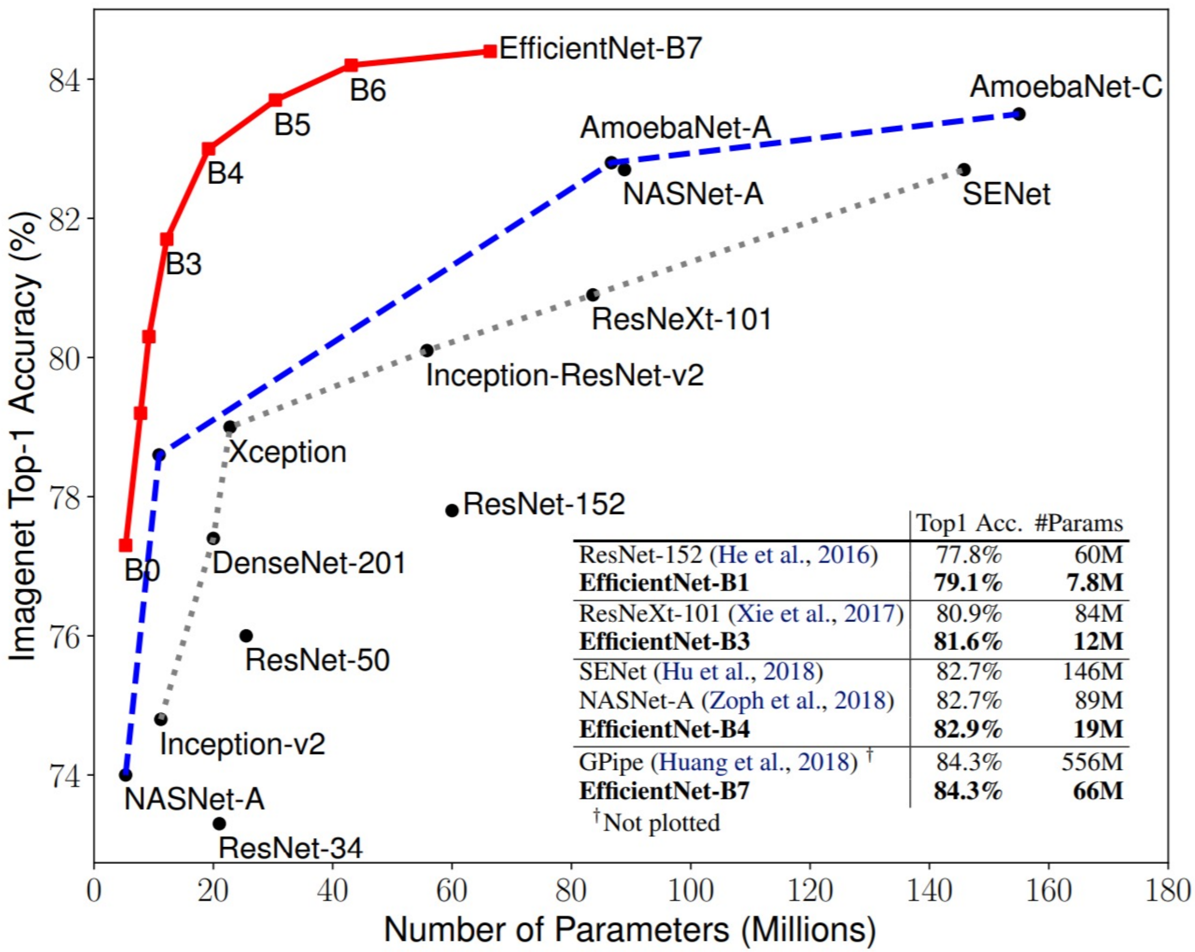
Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPs	Ratio-to-EfficientNet
EfficientNet-B0	**77.1%**	**93.3%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**79.1%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**80.1%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.6%**	**95.7%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.9%**	**96.4%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.6%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.8%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.3%**	**97.0%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

*source : Tan et al., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks", ICML 2019

Principle of Network Scaling: EfficientNet [Tan et al., 2019]

From EfficientNet-B0 to B7

- **EfficientNet-B0**: Baseline model with $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$
- **EfficientNet-B1 to B7**: Scaling up EfficientNet-B0 with different ϕ



EfficientNet-B7 achieves new state-of-the-art 84.3% top-1 accuracy but being 1.3x smaller than NASNet-A.

EfficientNet-B1 is 7.6x smaller and 5.7x faster than ResNet-152

Next, Dilated Conv

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

Table of Contents

Objects in real-world often contain sophisticated spatial information

- Multiple scales
- Irregular shapes

Drawbacks: geometric transformations are assumed fixed and known

- Different size and shape of kernels may be required
- But, regular kernels have fixed-size and shape

Scale:

Deformation:

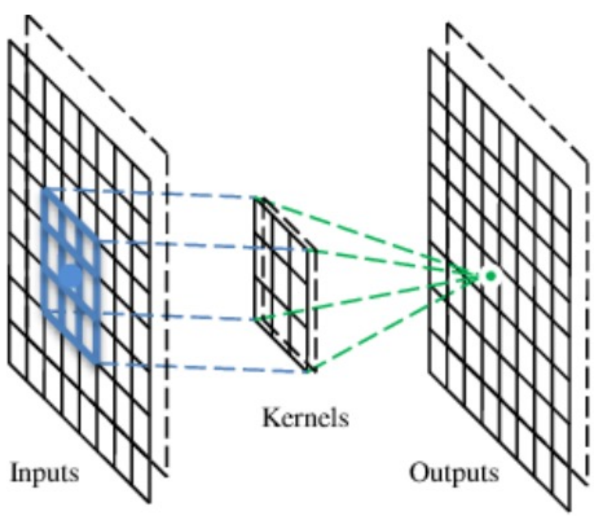
*source : https://jifengdai.org/slides/Deformable_Convolutional_Networks_Oral.pdf

Objects in real-world often contain sophisticated spatial information

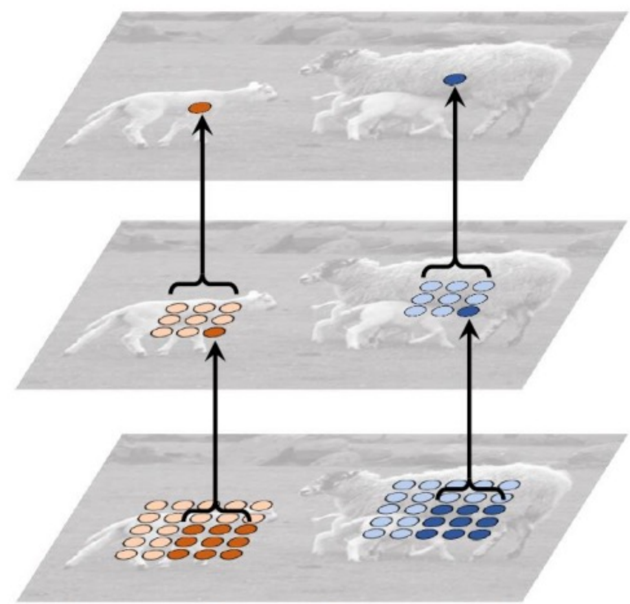
- Multiple scales
- Irregular shapes

Drawbacks: geometric transformations are assumed fixed and known

- Different size and shape of kernels may be required
- But, regular kernels have fixed-size and shape



regular convolution



2 layers of regular convolution

*source : https://jifengdai.org/slides/Deformable_Convolutional_Networks_Oral.pdf

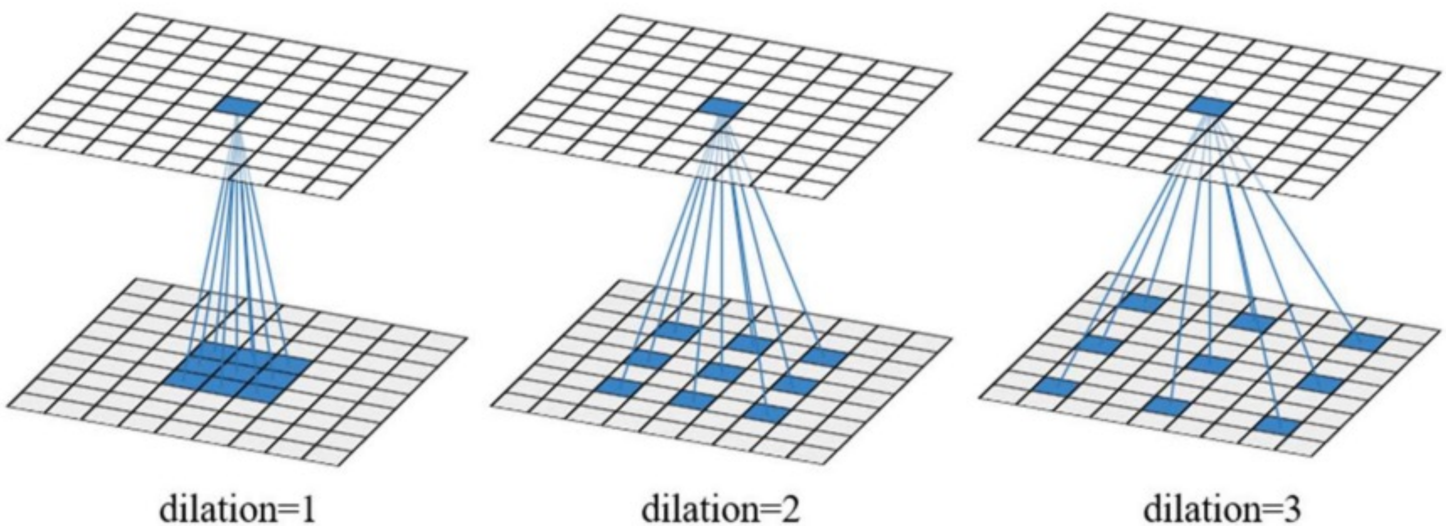
Dilated Convolution [Chen et al., 2017]

Motivation: Images in real-world usually contain **multi-scale objects**

- Regular convolution has a fixed-size of field of view
- Different size of kernels are required for multi-scale objects
- But, large-size of kernels may increase computational costs

Dilated convolution: Filling with **zero values** inside of large-size of kernels for efficient computation

- It can enlarge field-of-view to incorporate multi-scale context

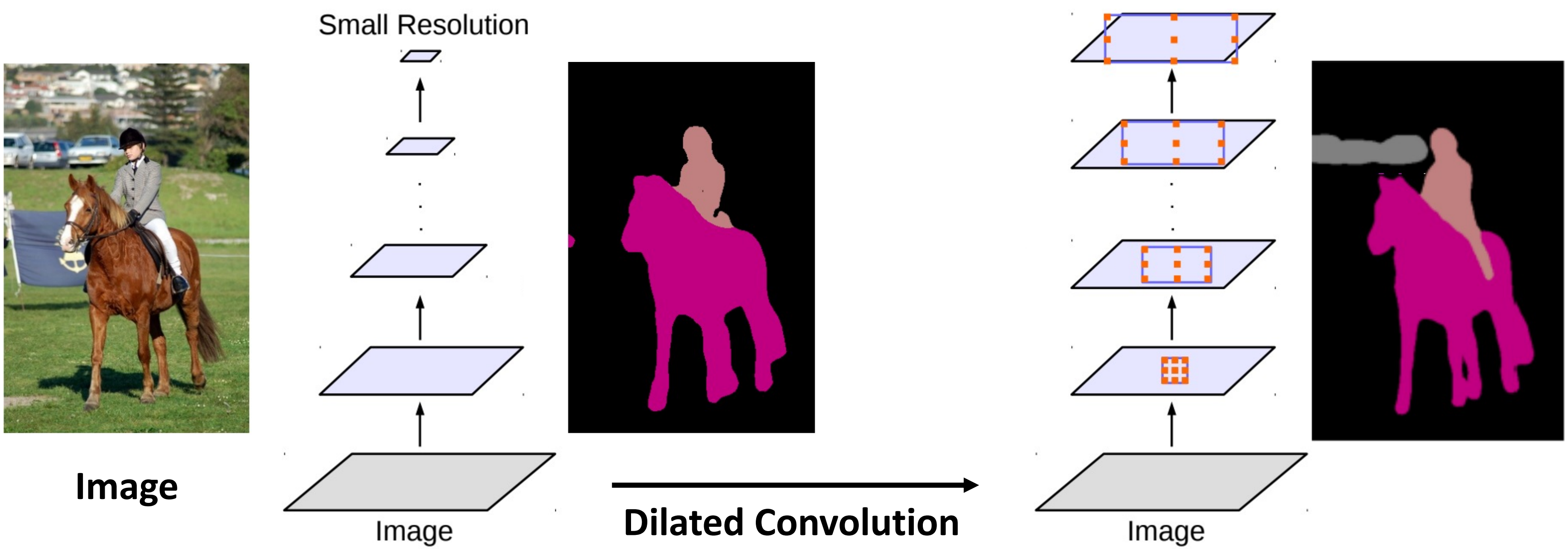


*source: Chen et al., "Rethinking atrous convolution for semantic image segmentation", ArXiv, 2019

Motivation: Images in real-world usually contain **multi-scale objects**

- Regular convolution has a fixed-size of field of view
- Different size of kernels are required for multi-scale objects
- But, large-size of kernels may increase computational costs

- **Example: Dilated convolution** in semantic segmentation



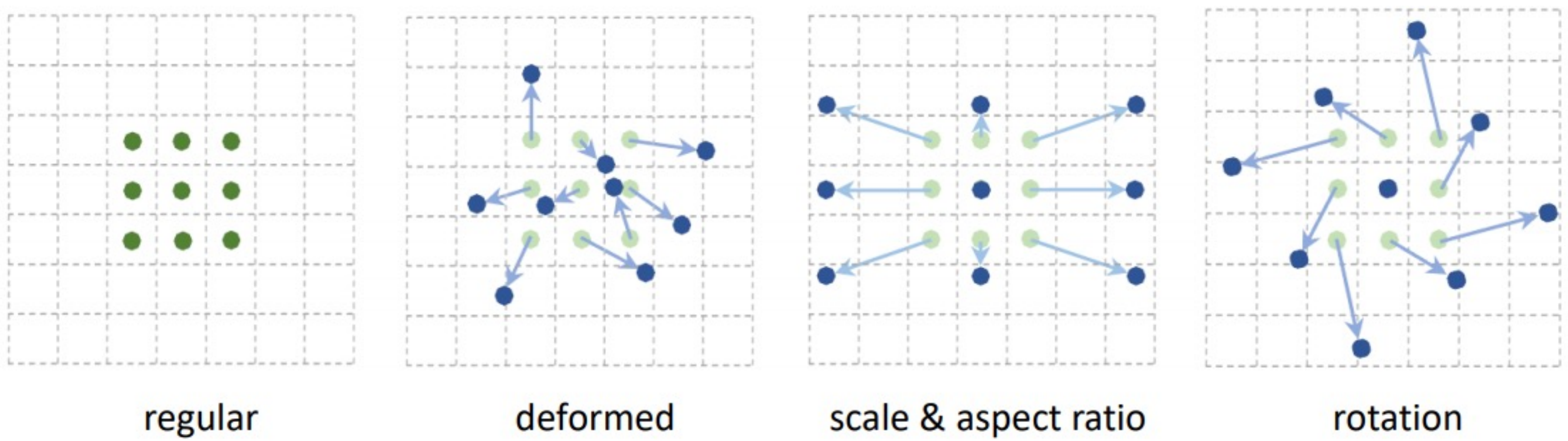
*source: Chen et al., "Rethinking atrous convolution for semantic image segmentation", ArXiv, 2019

Motivation: Shape of objects in the real world are usually irregular

- Different shape of kernels are required for irregular objects
- Regular convolution has a fixed-shape of kernel

Deformable convolution: Learning sampling location of kernels to capture irregular shape of objects

- Adding **offset field** to generate irregular sampling locations



Different types of sampling locations

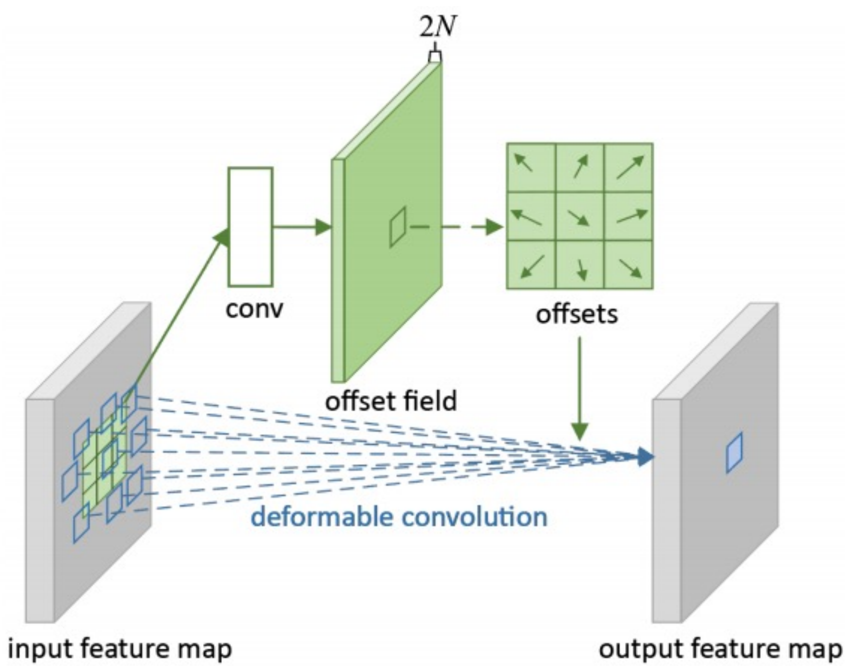
*source : https://jifengdai.org/slides/Deformable_Convolutional_Networks_Oral.pdf

Motivation: Shape of objects in the real world are usually irregular

- Different shape of kernels are required for irregular objects
- Regular convolution has a fixed-shape of kernel

Deformable convolution: Learning sampling location of kernels to capture irregular shape of objects

- Adding **offset field** to generate irregular sampling locations



Regular convolution

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n)$$

Deformable convolution

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n)$$

where $\Delta\mathbf{p}_n$ is generated by a sibling branch of regular convolution (offset field)

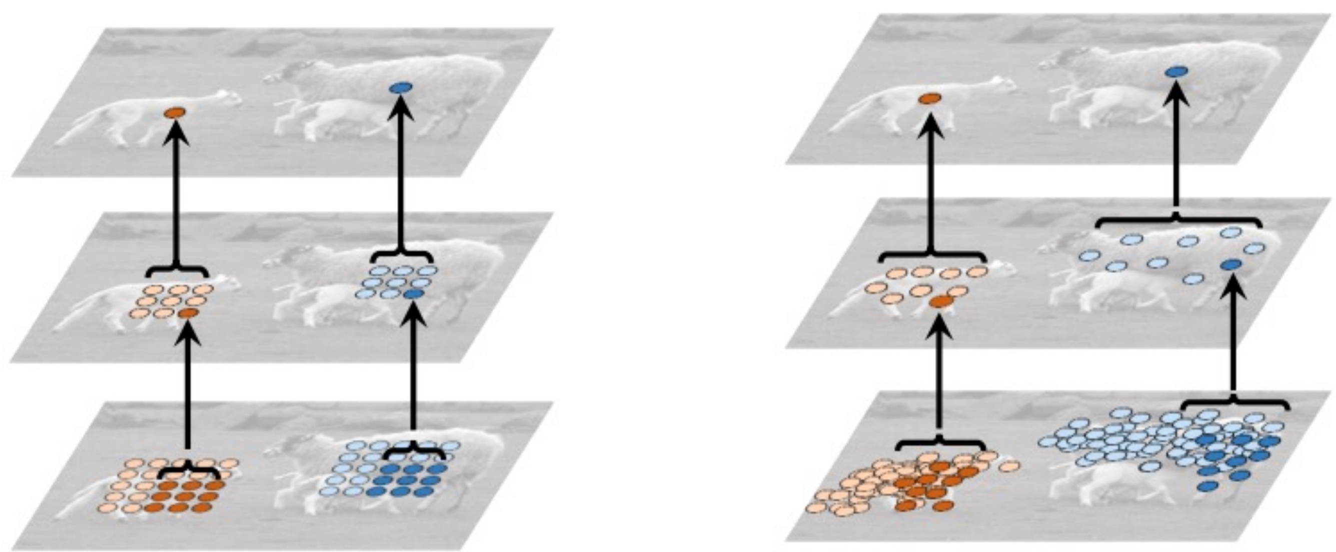
*source : https://jifengdai.org/slides/Deformable_Convolutional_Networks_Oral.pdf

Motivation: Shape of objects in the real world are usually irregular

- Different shape of kernels are required for irregular objects
- Regular convolution has a fixed-shape of kernel

Deformable convolution: Learning sampling location of kernels to capture irregular shape of objects

- Adding **offset field** to generate irregular sampling locations



(a) standard convolution (b) deformable convolution

*source : https://jifengdai.org/slides/Deformable_Convolutional_Networks_Oral.pdf

Motivation: Shape of objects in the real world are usually irregular

- Different shape of kernels are required for irregular objects
- Regular convolution has a fixed-shape of kernel

Learned offsets in the **deformable convolution** layers are highly adaptive to the image content

- Different size and shape of kernels for multiple objects

Visualizations of sampling locations

Next, SENet

*source: Dai et al., "Deformable Convolutional Networks", ICCV, 2017

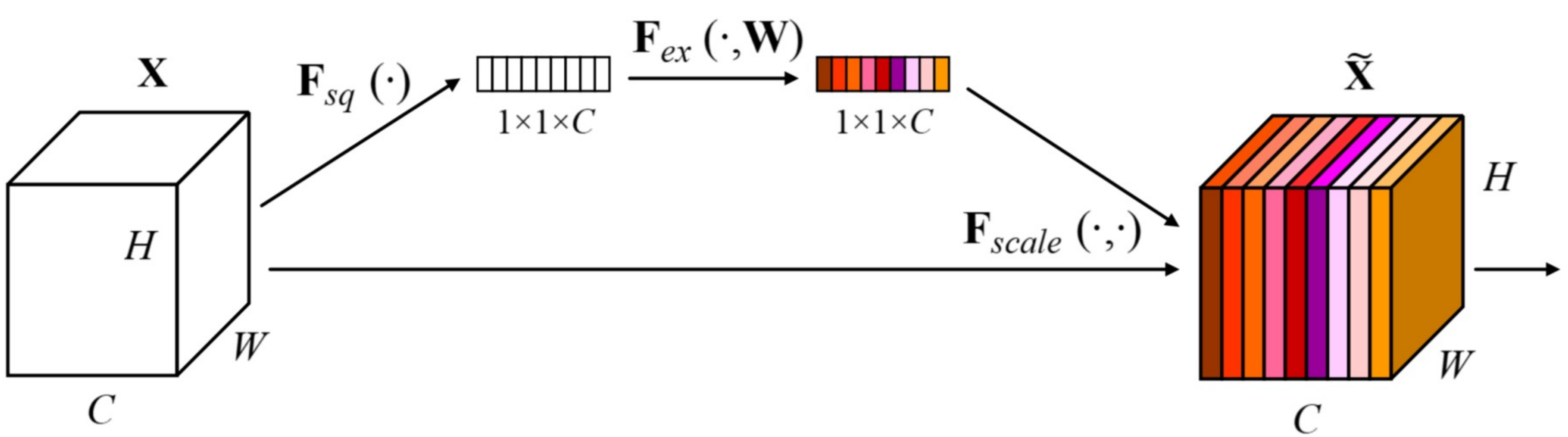
Table of Contents

Motivation: The deeper the model, the more feature maps are generated

- Many of them might be important for classification task
- Others might redundant or less important

Squeeze and Excitation Network [Hu et al., 2018]

- It selectively emphasizes informative feature maps and suppress less useful ones via global information in two steps
- **Squeeze** step: obtaining global information by shrinking feature maps
  - Global average pooling
- **Excitation** step: recalibrating weights of features by learning channel-wise weights
  - MLP of two fully-connected layers



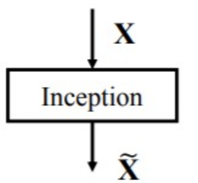
*source: Hu et al., "Squeeze-and-Excitation Networks", CVPR, 2018

Motivation: The deeper the model, the more feature maps are generated

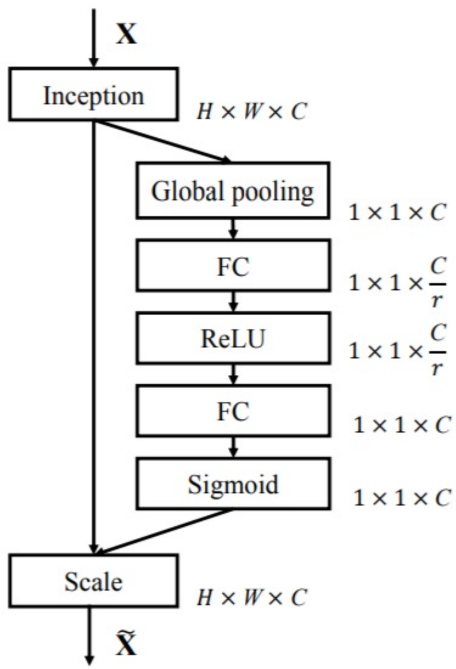
- Many of them might be important for classification task
- Others might redundant or less important

SE block integrates to Inception and ResNet module

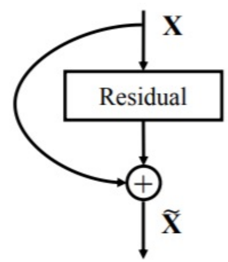
- SENet ranked first in the ILSVRC’17 (2.99% → **2.25%**)



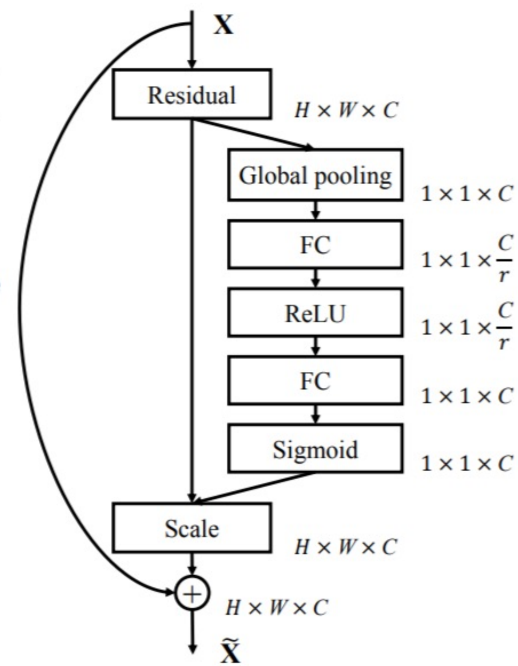
Inception Module



SE-Inception Module



ResNet Module



SE-ResNet Module

*source: Hu et al., “Squeeze-and-Excitation Networks”, CVPR, 2018

Motivation: The deeper the model, the more feature maps are generated

- Many of them might be important for classification task
- Others might redundant or less important

SE block integrates to Inception and ResNet module

- SENet ranked first in the ILSVRC'17 (2.99% → **2.25%**)

	original		re-implementation			SENet		
	top-1 err.	top-5 err.	top-1 err.	top-5 err.	GFLOPs	top-1 err.	top-5 err.	GFLOPs
ResNet-50 [13]	24.7	7.8	24.80	7.48	3.86	$23.29_{(1.51)}$	$6.62_{(0.86)}$	3.87
ResNet-101 [13]	23.6	7.1	23.17	6.52	7.58	$22.38_{(0.79)}$	$6.07_{(0.45)}$	7.60
ResNet-152 [13]	23.0	6.7	22.42	6.34	11.30	$21.57_{(0.85)}$	$5.73_{(0.61)}$	11.32
ResNeXt-50 [19]	22.2	-	22.11	5.90	4.24	$21.10_{(1.01)}$	$5.49_{(0.41)}$	4.25
ResNeXt-101 [19]	21.2	5.6	21.18	5.57	7.99	$20.70_{(0.48)}$	$5.01_{(0.56)}$	8.00
VGG-16 [11]	-	-	27.02	8.81	15.47	$25.22_{(1.80)}$	$7.70_{(1.11)}$	15.48
BN-Inception [6]	25.2	7.82	25.38	7.89	2.03	$24.23_{(1.15)}$	$7.14_{(0.75)}$	2.04
Inception-ResNet-v2 [21]	$19.9^{\dagger}$	$4.9^{\dagger}$	20.37	5.21	11.75	$19.80_{(0.57)}$	$4.79_{(0.42)}$	11.76

Next, Convolutional Block Attention Module

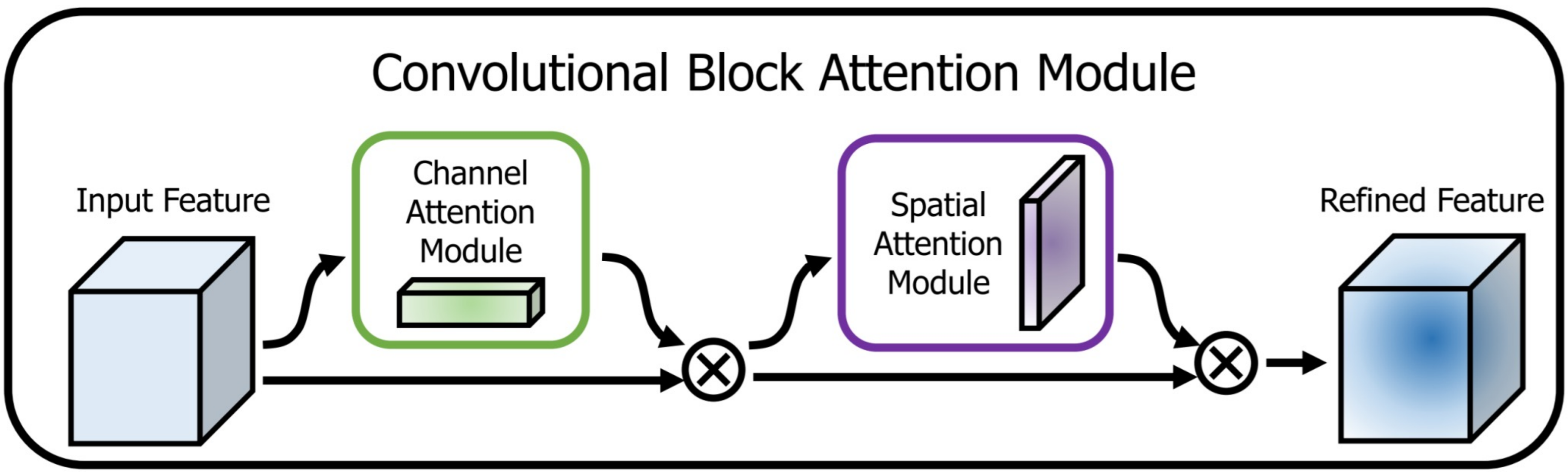
*source: Hu et al., "Squeeze-and-Excitation Networks", CVPR, 2018

Motivation: SENet only considers the contribution of feature maps

- It ignores the spatial locality of the object in image
- The spatial location of the object has a vital role in understanding image

Convolutional Block Attention Module (CBAM) [Woo et al., 2018]

- Learning 'what' and 'where' to attend in the channel and spatial axes respectively
- **Channel** and **Spatial** attention modules



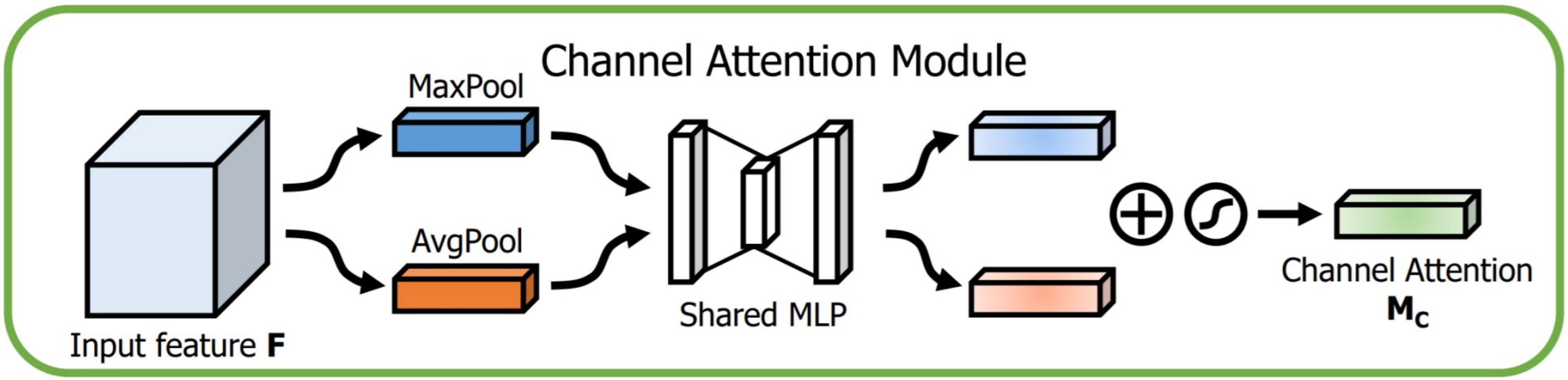
*source: Woo et al., "CBAM: Convolutional block attention module", ECCV, 2018

Motivation: SENet only considers the contribution of feature maps

- It ignores the spatial locality of the object in image
- The spatial location of the object has a vital role in understanding image

Channel attention module: It helps "what" to focus

- Both **average-pooling** and **max-pooling** are important
- **Max-pooling** provides the information of distinctive object features
- Both pooled features share a MLP with two fully-connected layers



$$\mathbf{M_c}(\mathbf{F}) = \sigma(MLP(AvgPool(\mathbf{F})) + MLP(MaxPool(\mathbf{F})))$$

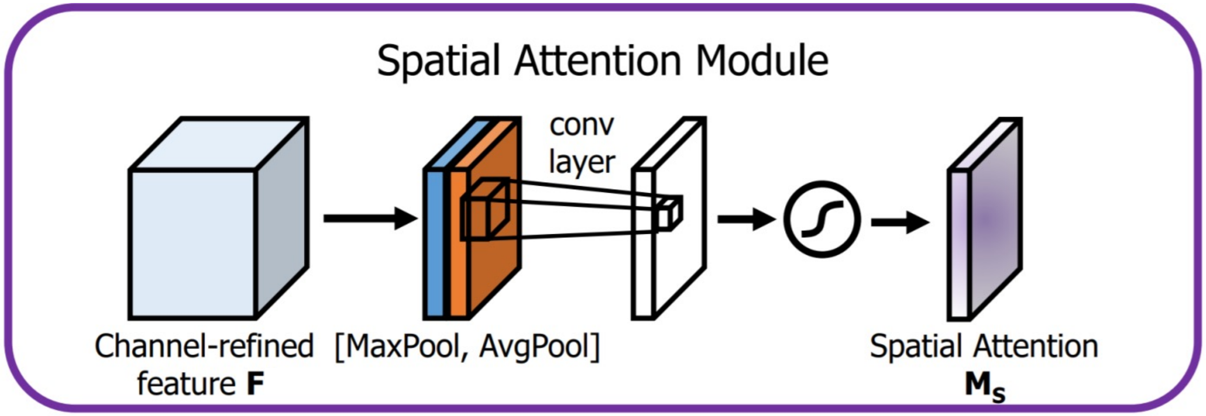
*source: Woo et al., "CBAM: Convolutional block attention module", ECCV, 2018

Motivation: SENet only considers the contribution of feature maps

- It ignores the spatial locality of the object in image
- The spatial location of the object has a vital role in understanding image

Spatial attention module: It helps "where" to focus

- Again, Both **average-pooling** and **max-pooling** are important
- It aggregates channel information of feature maps by using two pooling operations
- Capturing **spatial locality** via convolution



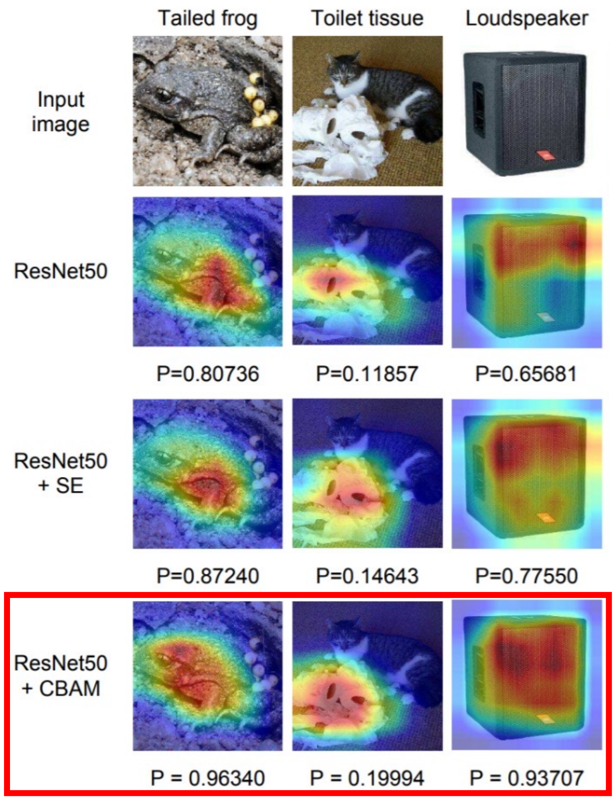
$$\mathbf{M_s}(\mathbf{F}) = \sigma(Conv([AvgPool(\mathbf{F}); MaxPool(\mathbf{F})]))$$

*source: Woo et al., "CBAM: Convolutional block attention module", ECCV, 2018

Motivation: SENet only considers the contribution of feature maps

- It ignores the spatial locality of the object in image
- The spatial location of the object has a vital role in understanding image

- **CBAM** module integrated with ResNet outperforms SE module



Grad-CAM visualization

Architecture	Param.	GFLOPs	Top-1 Error (%)	Top-5 Error (%)
ResNet18 [5]	11.69M	1.814	29.60	10.55
ResNet18 [5] + SE [28]	11.78M	1.814	29.41	10.22
ResNet18 [5] + CBAM	11.78M	1.815	**29.27**	**10.09**
ResNet34 [5]	21.80M	3.664	26.69	8.60
ResNet34 [5] + SE [28]	21.96M	3.664	26.13	8.35
ResNet34 [5] + CBAM	21.96M	3.665	**25.99**	**8.24**
ResNet50 [5]	25.56M	3.858	24.56	7.50
ResNet50 [5] + SE [28]	28.09M	3.860	23.14	6.70
ResNet50 [5] + CBAM	28.09M	3.864	**22.66**	**6.31**
ResNet101 [5]	44.55M	7.570	23.38	6.88
ResNet101 [5] + SE [28]	49.33M	7.575	22.35	6.19
ResNet101 [5] + CBAM	49.33M	7.581	**21.51**	**5.69**
WideResNet18 [6] (widen=1.5)	25.88M	3.866	26.85	8.88
WideResNet18 [6] (widen=1.5) + SE [28]	26.07M	3.867	26.21	8.47
WideResNet18 [6] (widen=1.5) + CBAM	26.08M	3.868	**26.10**	**8.43**
WideResNet18 [6] (widen=2.0)	45.62M	6.696	25.63	8.20
WideResNet18 [6] (widen=2.0) + SE [28]	45.97M	6.696	24.93	7.65
WideResNet18 [6] (widen=2.0) + CBAM	45.97M	6.697	**24.84**	**7.63**
ResNeXt50 [7] (32x4d)	25.03M	3.768	22.85	6.48
ResNeXt50 [7] (32x4d) + SE [28]	27.56M	3.771	**21.91**	6.04
ResNeXt50 [7] (32x4d) + CBAM	27.56M	3.774	21.92	**5.91**
ResNeXt101 [7] (32x4d)	44.18M	7.508	21.54	5.75
ResNeXt101 [7] (32x4d) + SE [28]	48.96M	7.512	21.17	5.66
ResNeXt101 [7] (32x4d) + CBAM	48.96M	7.519	**21.07**	**5.59**

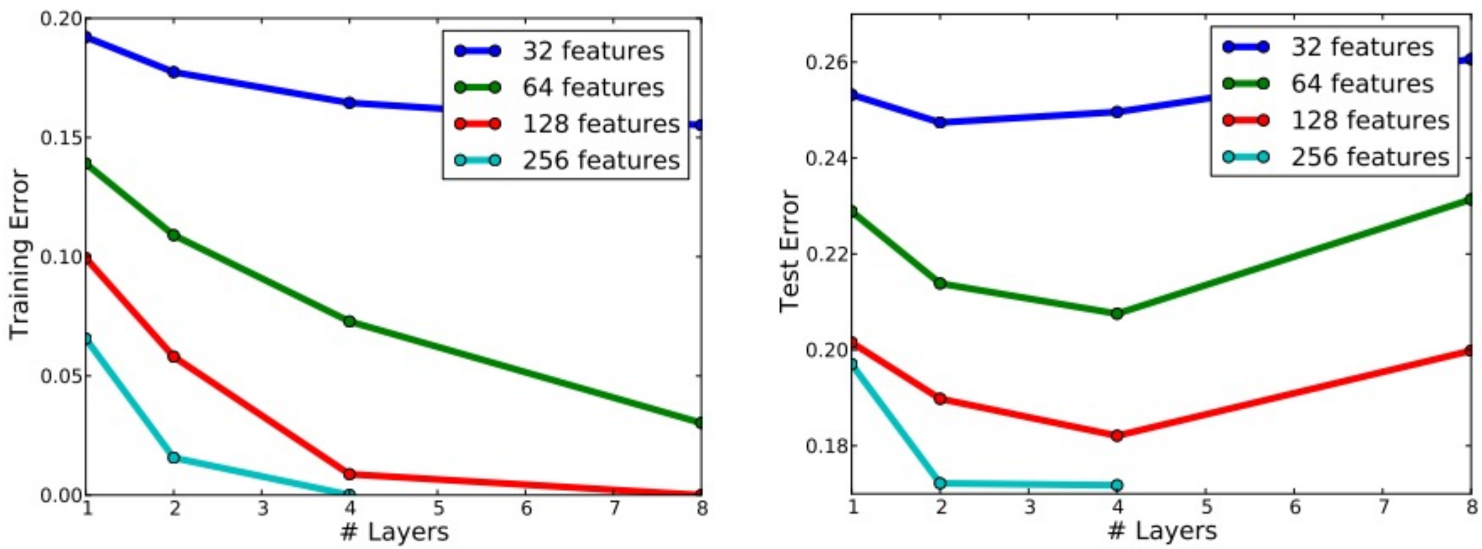
*source: Woo et al., “CBAM: Convolutional block attention module”, ECCV, 2018

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]

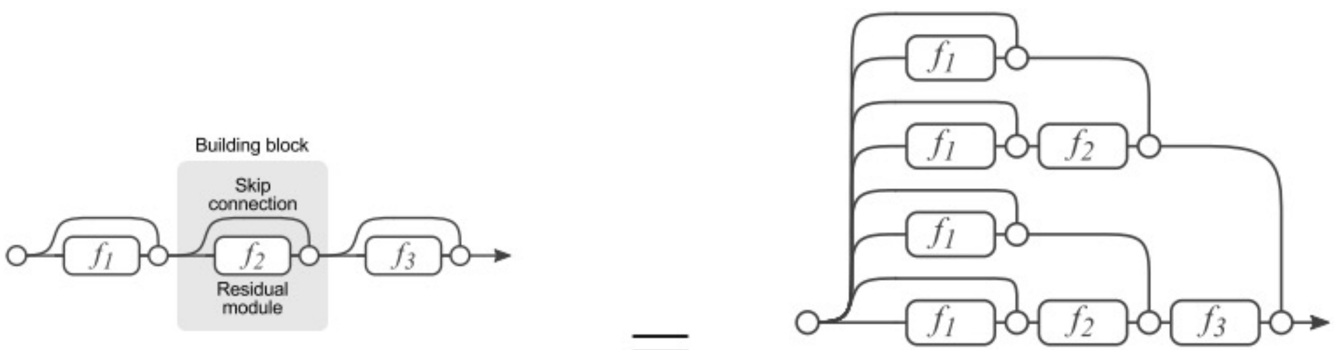


*source : Eigen et al., "Understanding Deep Architectures using a Recursive Convolutional Network", Arxiv 2014

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**

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



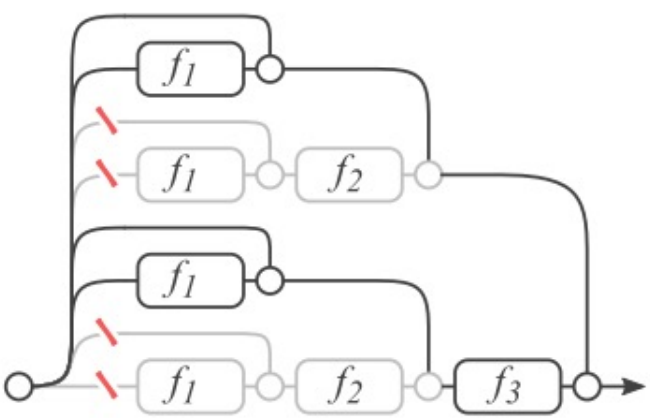
(a) Conventional 3-block residual network

(b) Unraveled view of (a)

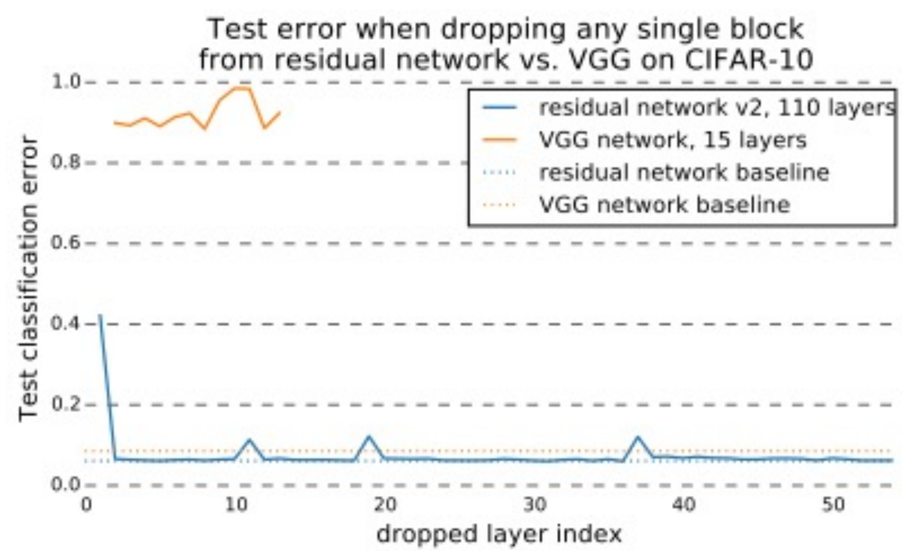
*source : Veit et al., “ResNets behave like ensembles of relatively shallow nets”, NIPS 2016

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

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 **Random directions**

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

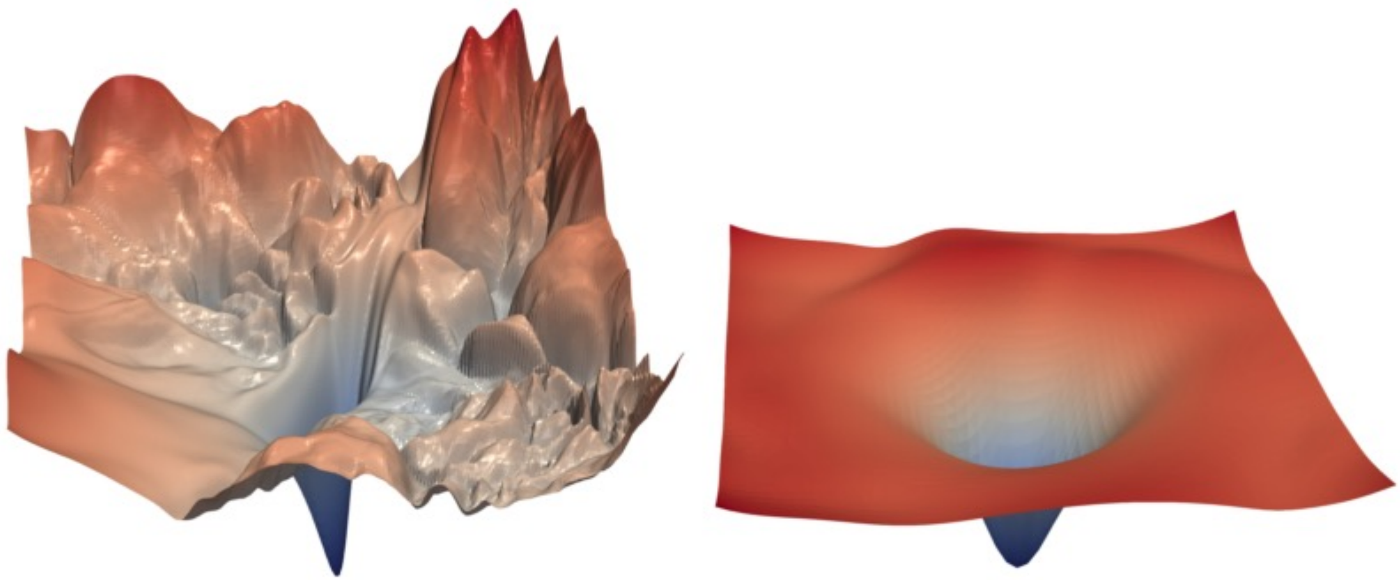
$$\delta_{i,j} \leftarrow \frac{\delta_{i,j}}{||\delta_{i,j}||}||\theta_{i,j}||$$

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



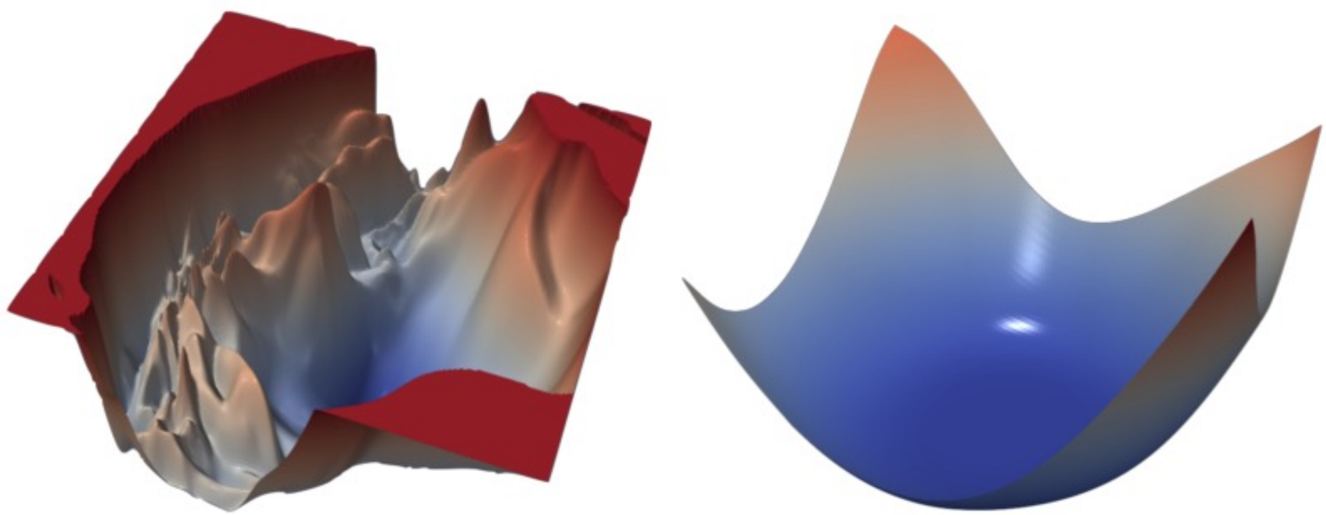
(a) without skip connections (b) with skip connections

ResNet-56

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

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

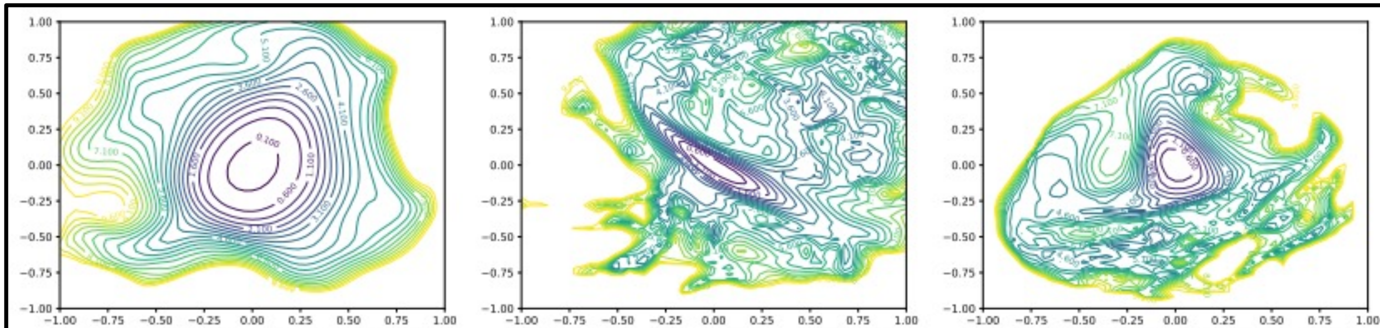
(b) DenseNet, 121 layers

*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

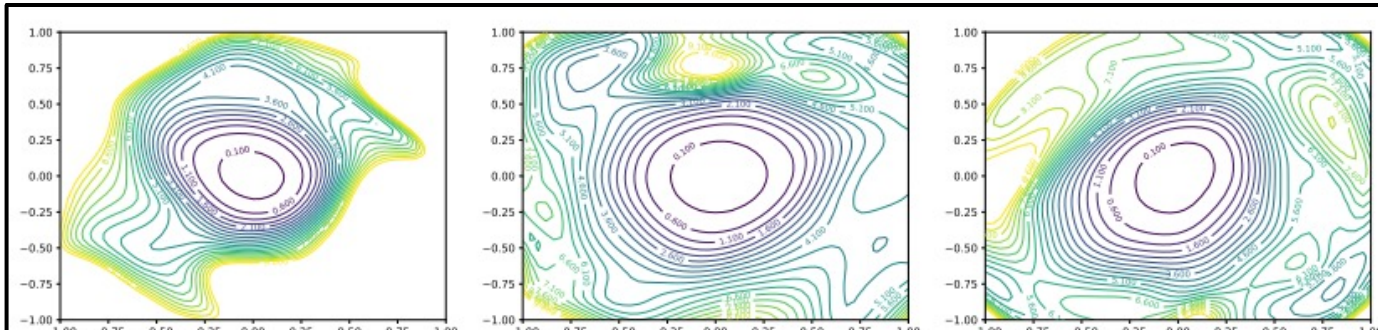
Modern architectures prevent the loss to be chaotic as depth increases

ResNet, **no shortcuts** ⇒ sharp minima



(d) ResNet-20-noshort, 8.18% (e) ResNet-56-noshort, 13.31% (f) ResNet-110-noshort, 16.44%

ResNet ⇒ flat minima



(a) ResNet-20, 7.37% (b) ResNet-56, 5.89% (c) ResNet-110, 5.79%

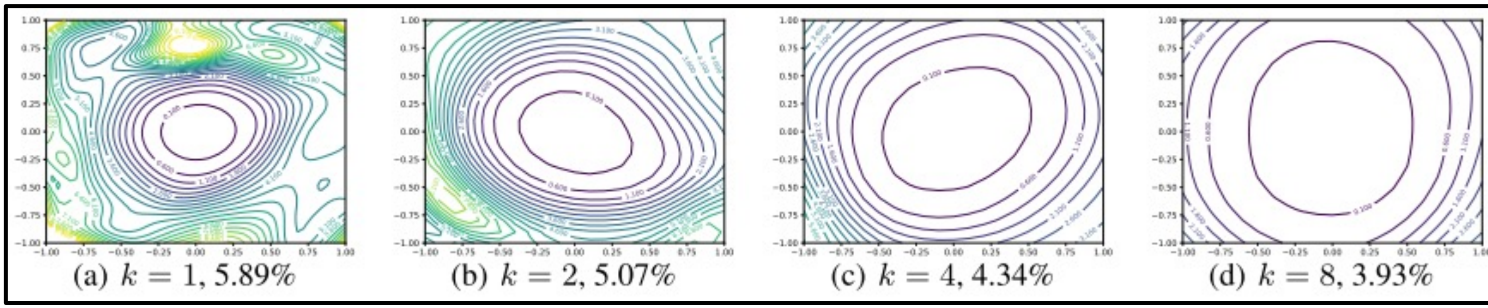
*source : Li et al., “Visualizing the loss landscape of neural nets”, ICLR Workshop 2018

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

Wide-ResNet lead the network toward more flat minimizer

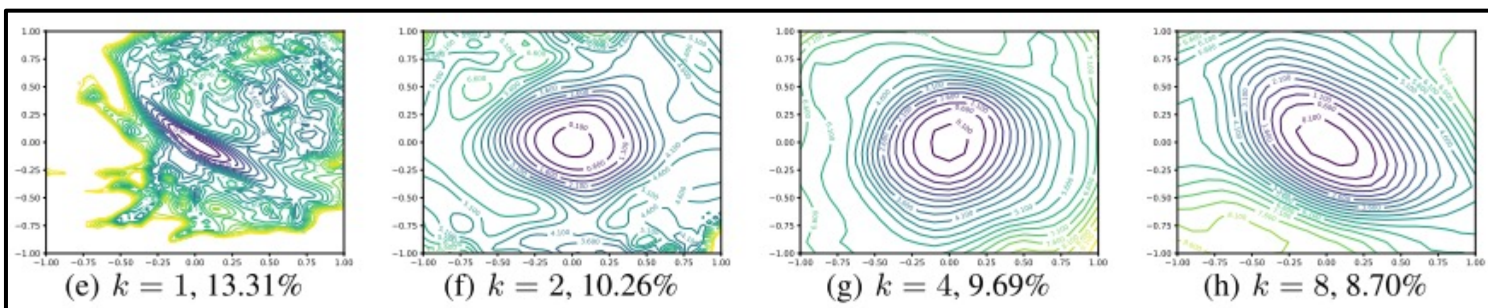
- WideResNet-56 with **width-multiplier** $k = 1, 2, 4, 8$
- Increased width flatten the minimizer in ResNet

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

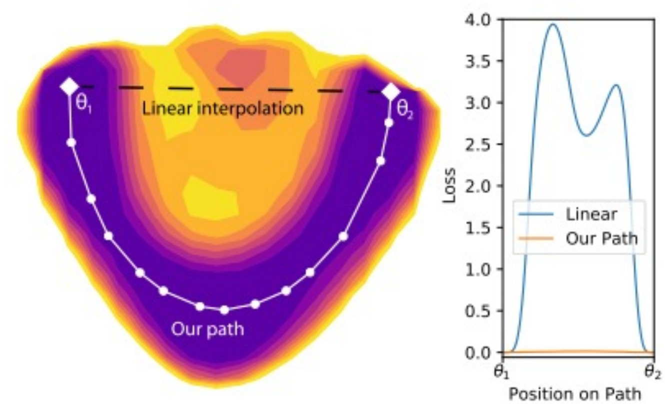
*source : Li et al., "Visualizing the loss landscape of neural nets", ICLR Workshop 2018

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]

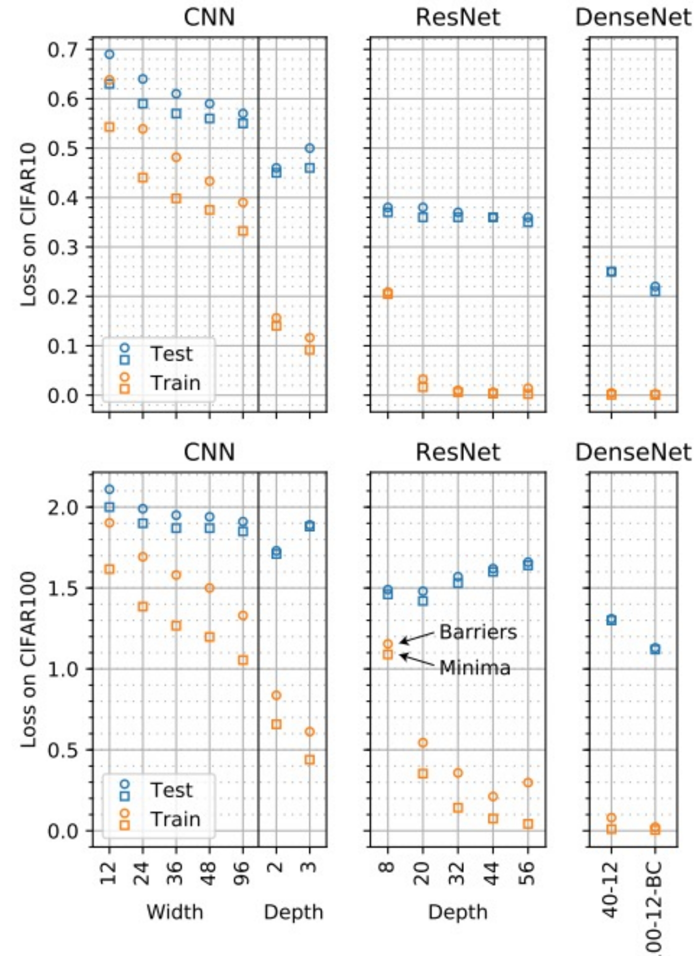
For a given model with two local minima θ_1 and θ_2, they applied **AutoNEB** [Kolsbjerg et al., 2016] to find a minimum energy path

- 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%



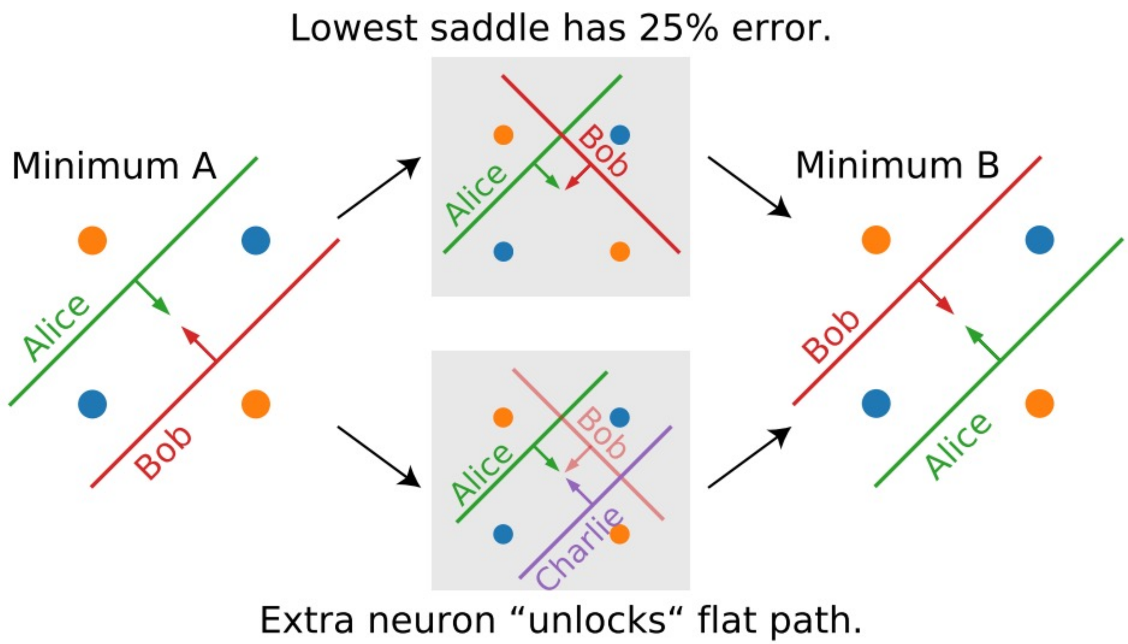
*source : Draxler et al., "Essentially no barriers in neural network energy landscape", ICML 2018

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

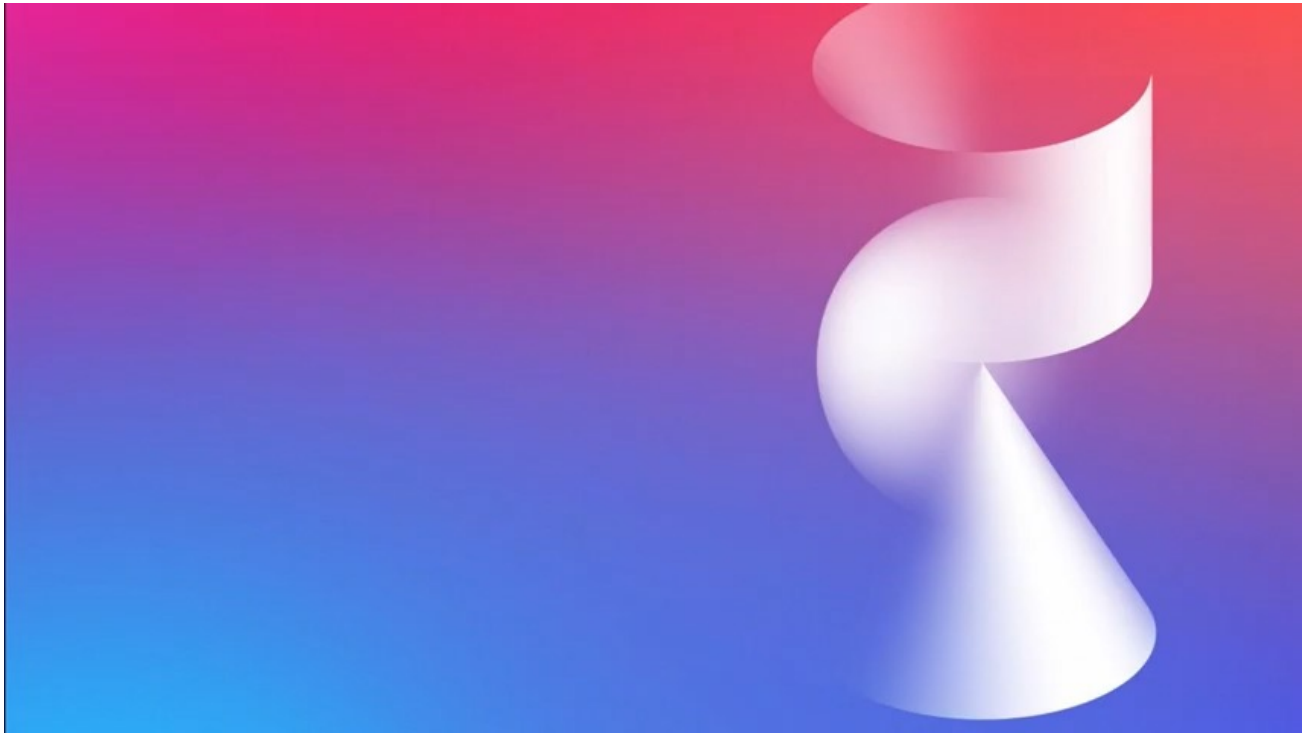


*source : Draxler et al., "Essentially no barriers in neural network energy landscape", ICML 2018

Table of Contents

Success of Transformer in Language: GPT-3

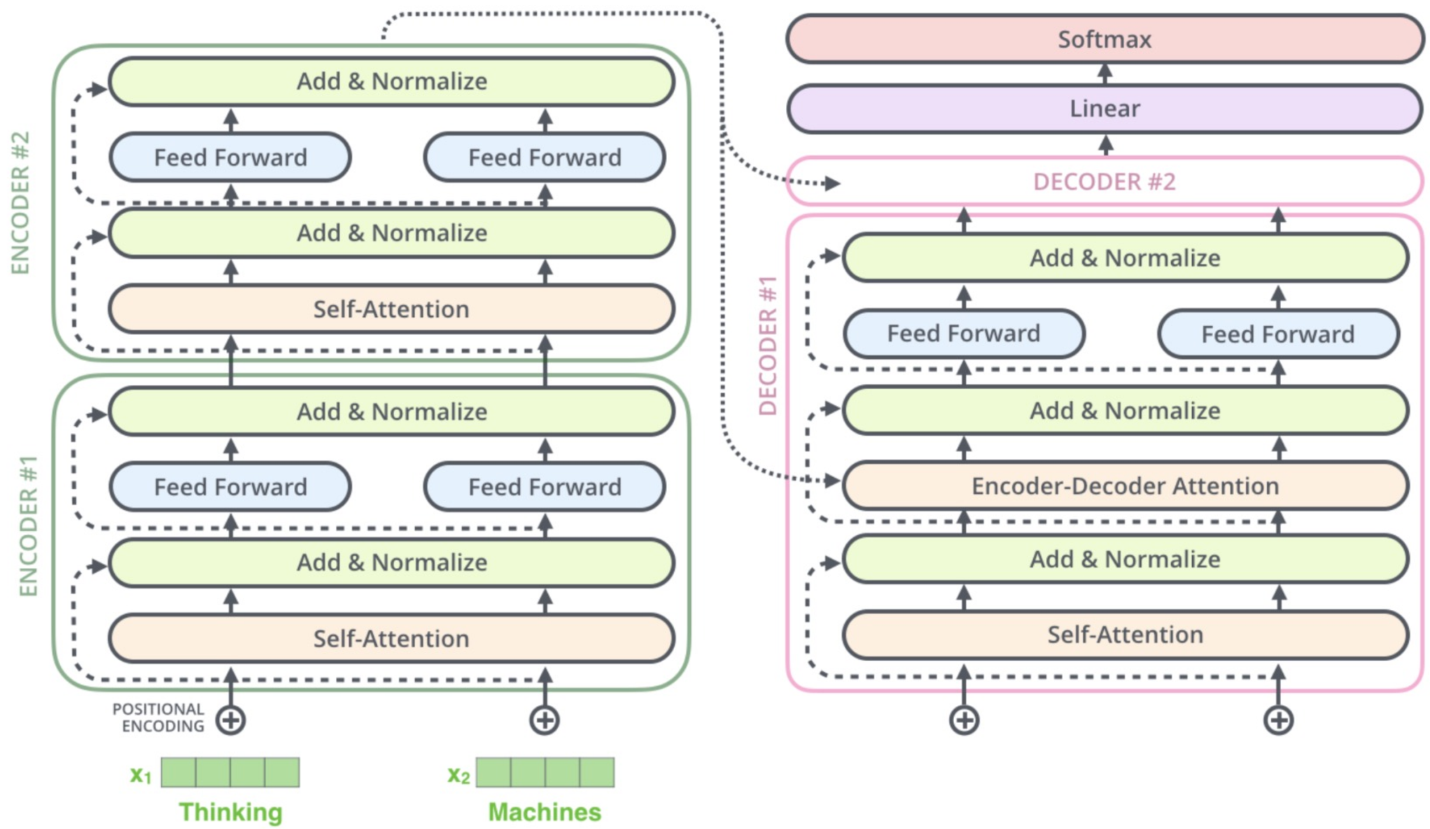
- In 2020, **GPT-3** achieved near-human results in various tasks
- OpenAI even trained a model with **175 billion** parameters (**350 GB** of memory) and showed near-human performance on various **few-shot** tasks



*source : https://youtu.be/CSe3_u9P-RM
Draxler et al., "Essentially no barriers in neural network energy landscape", ICML 2018

What is Transformer?

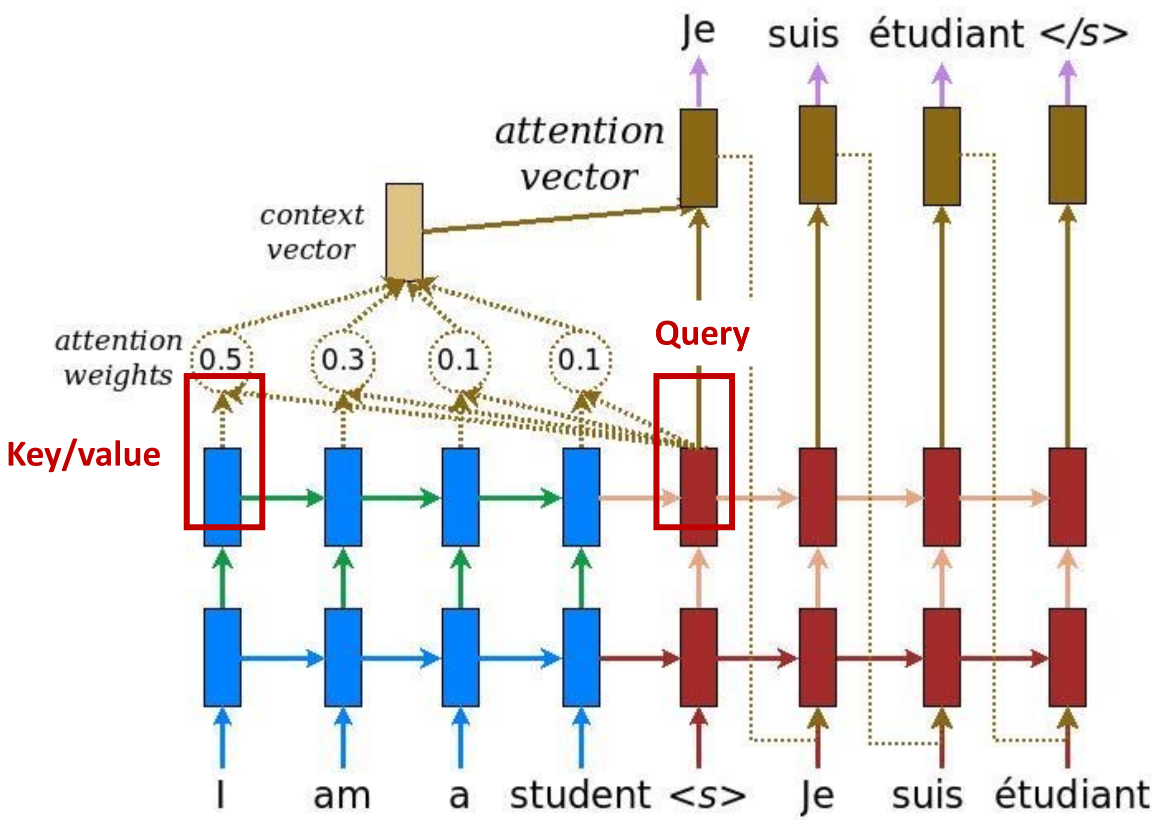
- Transformer [Vaswani et al., 2017] has an **encoder-decoder** structure and they are composed of multiple block with **self-attention** module



*source: http:// http://jalammar.github.io/illustrated-transformer

What is Transformer?

- Transformer [Vaswani et al., 2017] has an **encoder-decoder** structure and they are composed of multiple block with **self-attention** module
- The self-attention is a function of **query** (e.g., “Je”) and **key/value** (e.g., “I”)
  - It shows powerful performances in learning **sequential input-output relations**



Attention mechanism can be used for other type of input data, e.g. image

- Self-attention operation scales **quadratically** with the sequence length

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Question: How to transform an image to **sequence data**?

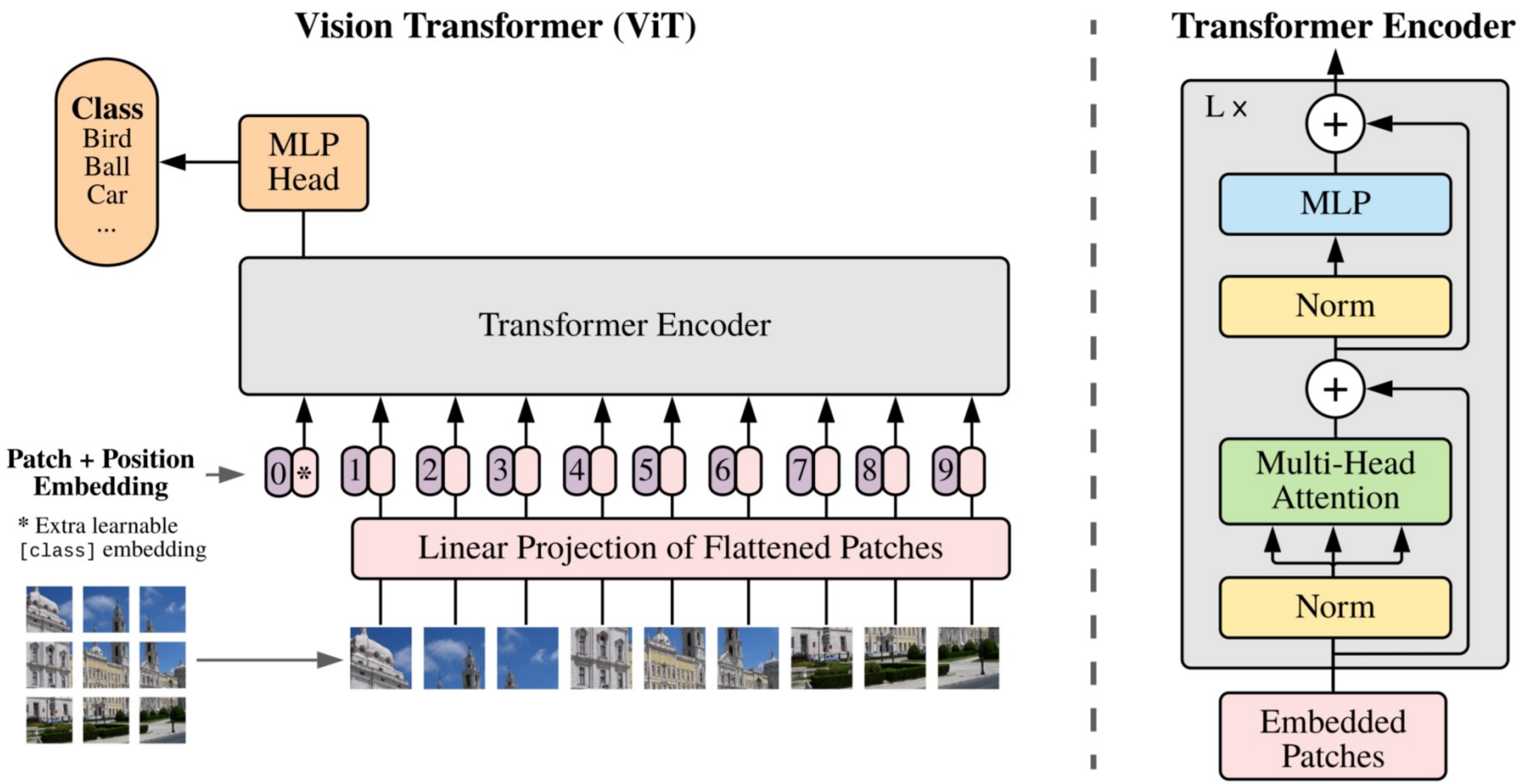
- Dosovitskiy et al. (2021): splits an **image** into **patches**

Sequence of patch images

*source: [Chen et al. 2020] Generative Pretraining from Pixels, ICML 2020
[Dosovitskiy et al. 2021] An image is worth 16x16 words: Transformers for image recognition at scale, ICLR 2021

Vision Transformer [Dosovitskiy et al., 2021]

- Splitting an image into fixed-size patches (16x16)
  - Linearly embedding each of them
- Adding position embedding & [class] token



*source: [Dosovitskiy et al. 2021] An image is worth 16x16 words: Transformers for image recognition at scale, ICLR 2021

Vision Transformer [Dosovitskiy et al., 2021]

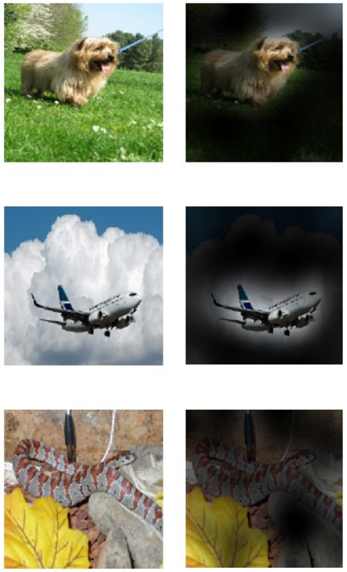
- Splitting an image into fixed-size patches (16x16)
  - Linearly embedding each of them
- Adding position embedding & [class] token

- **Dosovitskiy et al.** (2021) pre-trains models on larger datasets (14M-300M images)
  - Vision Transformer achieves **competitive performances** compared to CNNs

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** $\pm$ 0.04	87.76 $\pm$ 0.03	85.30 $\pm$ 0.02	87.54 $\pm$ 0.02	88.4/88.5*
ImageNet ReaL	**90.72** $\pm$ 0.05	90.54 $\pm$ 0.03	88.62 $\pm$ 0.05	90.54	90.55
CIFAR-10	**99.50** $\pm$ 0.06	99.42 $\pm$ 0.03	99.15 $\pm$ 0.03	99.37 $\pm$ 0.06	–
CIFAR-100	**94.55** $\pm$ 0.04	93.90 $\pm$ 0.05	93.25 $\pm$ 0.05	93.51 $\pm$ 0.08	–
Oxford-IIIT Pets	**97.56** $\pm$ 0.03	97.32 $\pm$ 0.11	94.67 $\pm$ 0.15	96.62 $\pm$ 0.23	–
Oxford Flowers-102	99.68 $\pm$ 0.02	**99.74** $\pm$ 0.00	99.61 $\pm$ 0.02	99.63 $\pm$ 0.03	–
VTAB (19 tasks)	**77.63** $\pm$ 0.23	76.28 $\pm$ 0.46	72.72 $\pm$ 0.21	76.29 $\pm$ 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Vision Transformer **CNNs**

Input Attention



*source: [Dosovitskiy et al. 2021] An image is worth 16x16 words: Transformers for image recognition at scale, ICLR 2021

Various architectures now are based on Vision Transformer

1. **Modification for patch splitting**
   - Token-to-Token Vision Transformer [Li et al., 2021]
   - Swin Transformer [Liu et al., 2021]
2. **Modification for hierarchical structure**
   - Pooling-based Vision Transformer [Heo et al., 2021]
   - Swin Transformer [Liu et al., 2021]

Question: What's a good way to split an **image** into a **sequence of patches**?

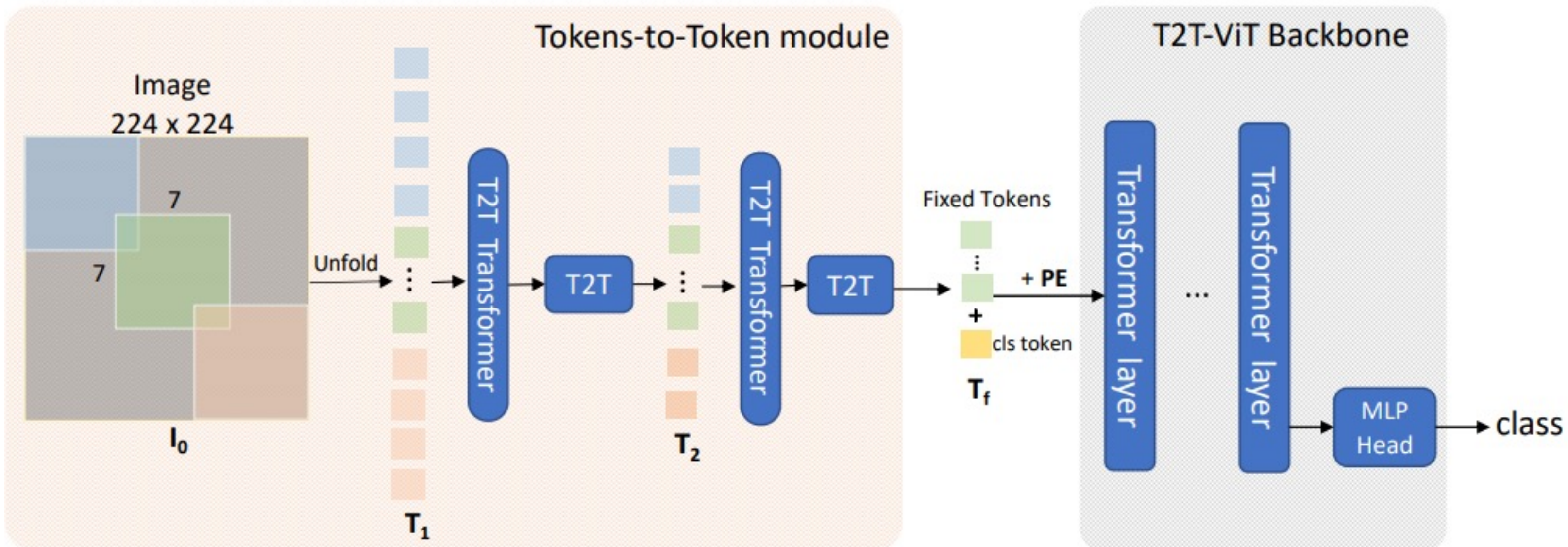
- Vision Transformer splits an image into a **fixed grid-shape** of **non-overlapping** patches

Sequence of patch images

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

Token-to-Token Vision Transformer [Li et al., 2021]

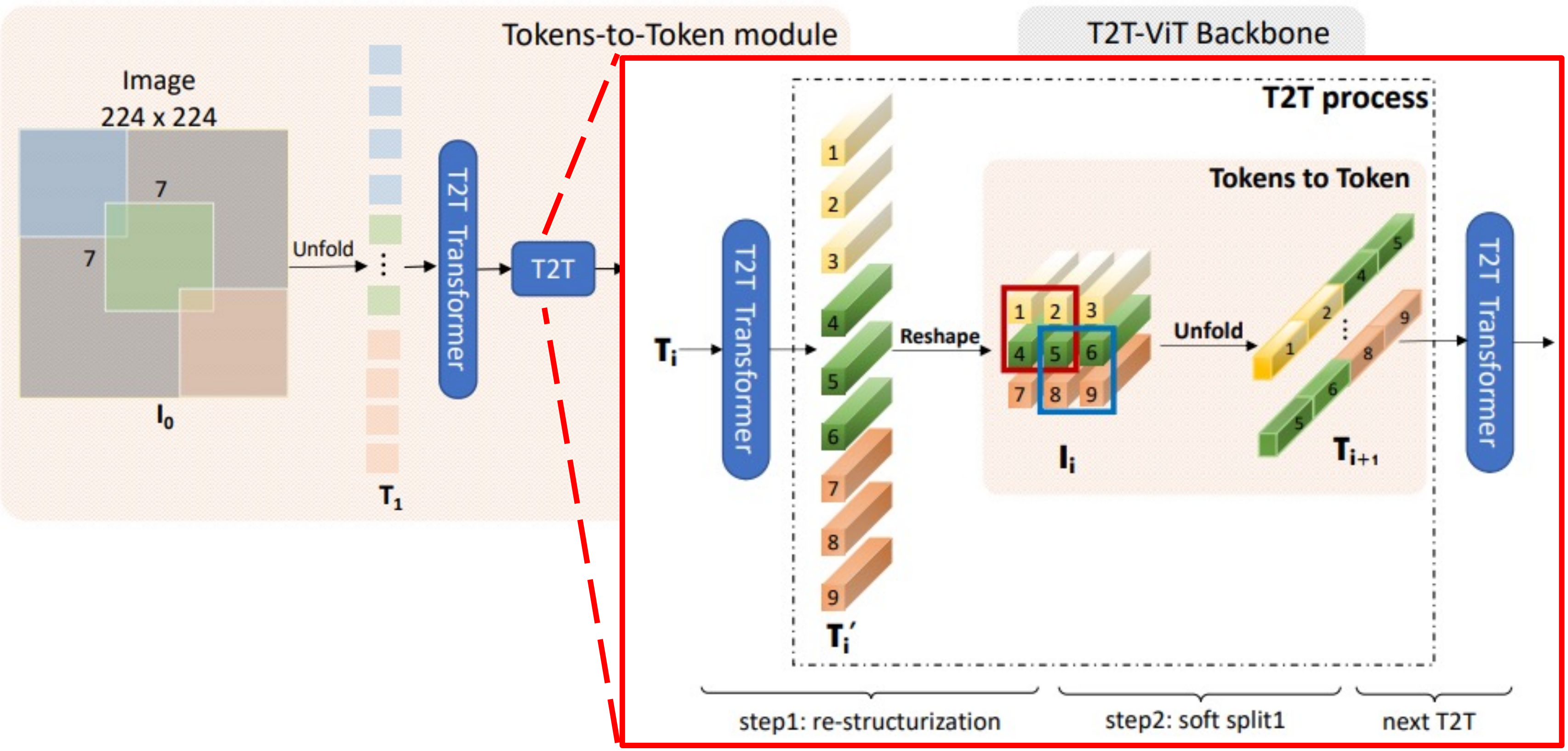
- **(Soft-split)** Splitting an image into overlapping patches
- **(Re-structurization)** Rearranging patch sequences into 2D image shape
- Iterating re-structurization and soft-split before Transformer backbone



*source: [Li et al. 2021] Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet, ICCV 2021

Token-to-Token Vision Transformer [Li et al., 2021]

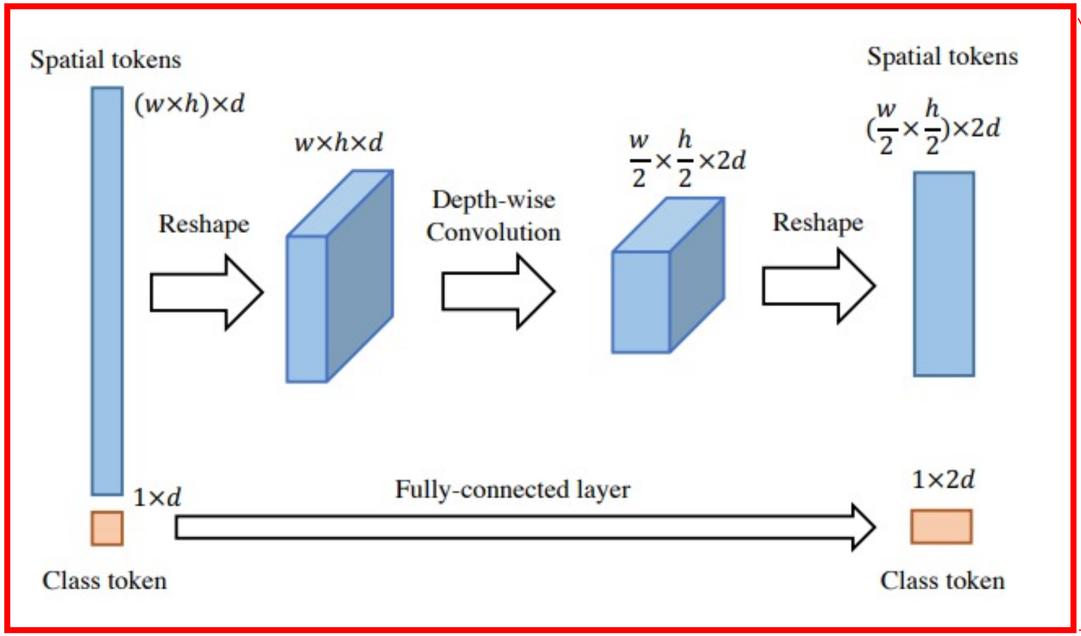
- **(Soft-split)** Splitting an image into overlapping patches
- **(Re-structurization)** Rearranging patch sequences into 2D image shape
- Iterating re-structurization and soft-split before Transformer backbone

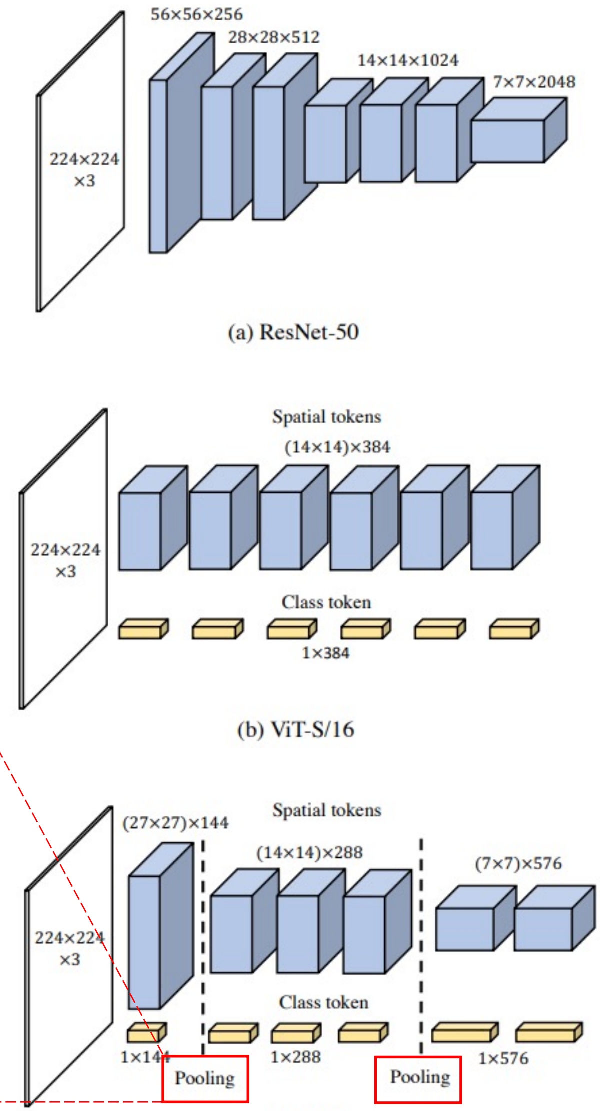


*source: [Li et al. 2021] Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet, ICCV 2021

Pooling-based Vision Transformer [Heo et al., 2021]

- Design of a hierarchical structure
  - **Motivation:** ResNet gradually downsamples the features from the input to the output
- Downsampling via the pooling layer based on depth-wise convolution
- Spatial reduction with small parameters





(a) ResNet-50

(b) ViT-S/16

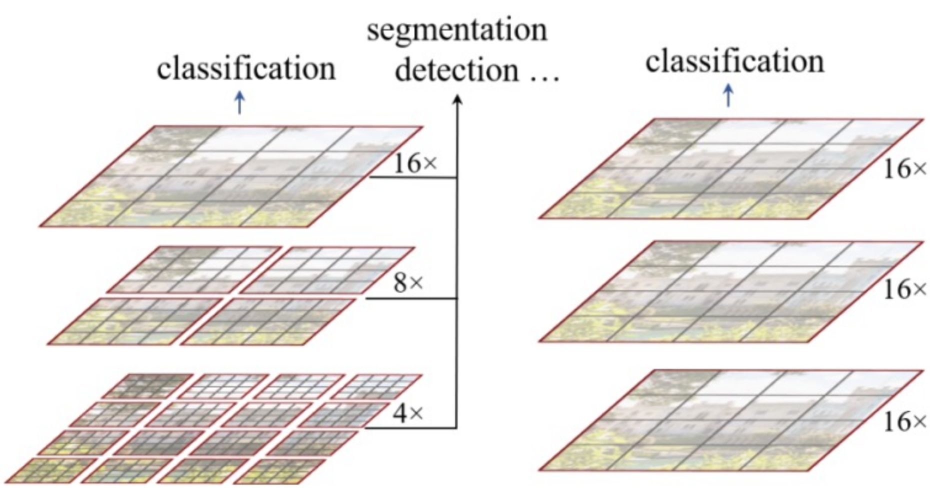
(c) PiT-S

*source: [Heo et al. 2021] Rethinking Spatial Dimensions of Vision Transformers , ICCV 2021

Swin Transformer [Liu et al., 2021]

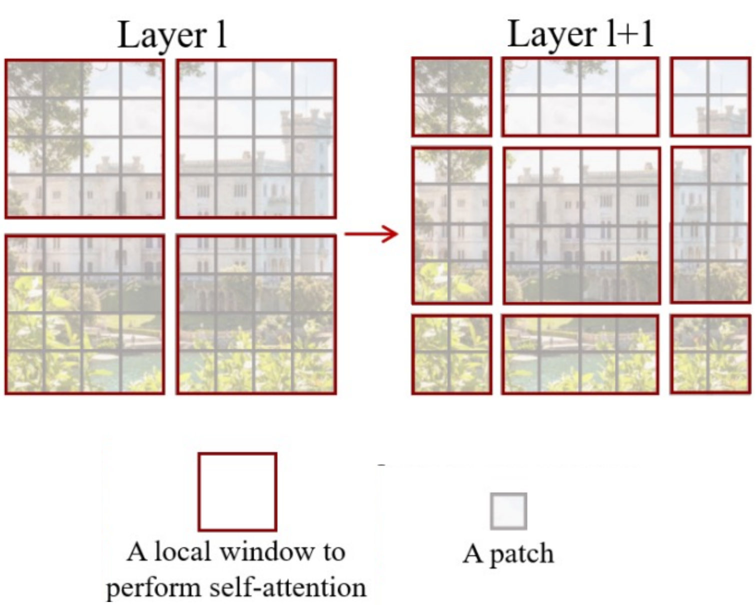
- Design of a hierarchical structure
- Various spatial resolutions (e.g., patch-shape) can be handled via shifted windows
- Efficient self-attention computation by using shifted windows scheme
- Concatenating 2 × 2 neighboring patches for downsampling operation
- Powerful performances in dense prediction tasks

  e.g., object detection and semantic segmentation



(a) Swin Transformer (ours) (b) ViT

Shifted window scheme

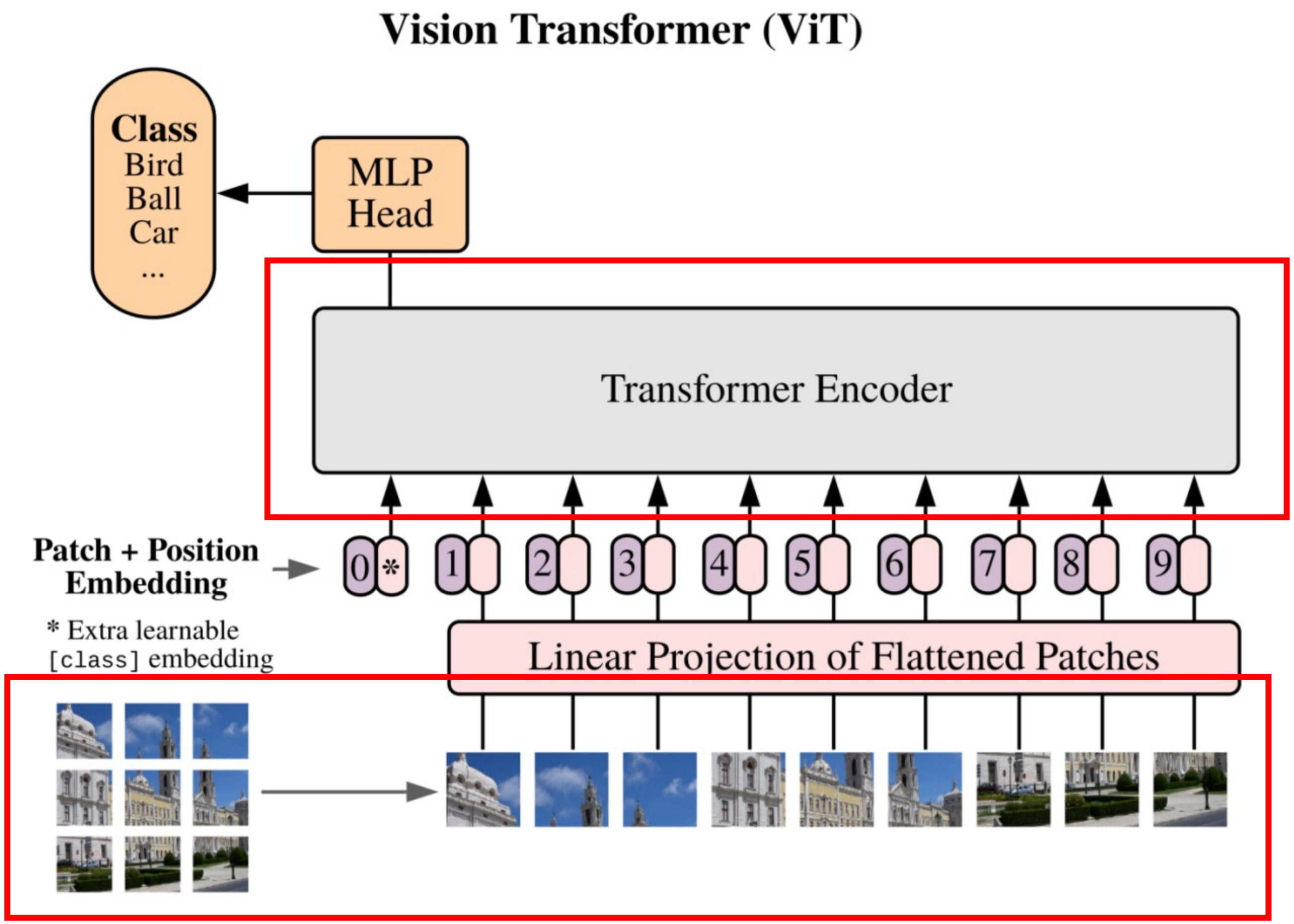


*source: [Liu et al. 2021] Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, ICCV 2021

Table of Contents

Question: Is the success of **Vision Transformers** due to

1. the powerful Transformer architecture?
2. using patches as the input representation?



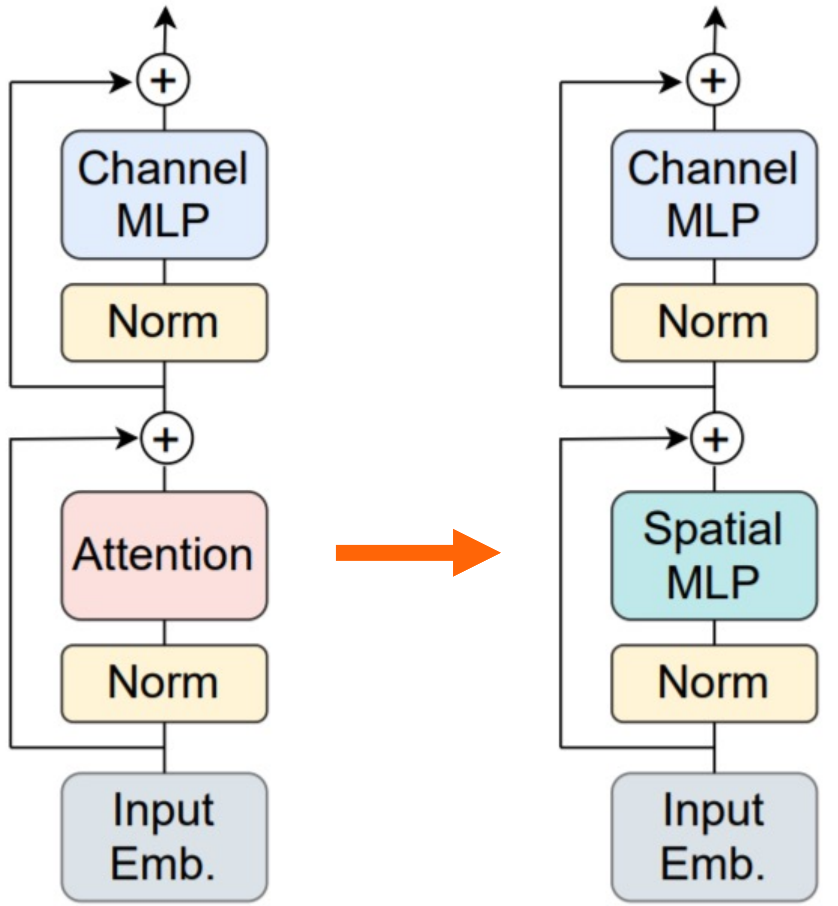
*source: [Dosovitskiy et al. 2021] An image is worth 16x16 words: Transformers for image recognition at scale, ICLR 2021

- **Tolstikhin et al.** (2021) suggests MLP module as an alternative of self-attention module
  - For a given Image I,

$$Z = \mathrm{MLP}\big(\mathrm{Norm}(Y)\big) + Y$$
$$= \sigma(\mathrm{Norm}(Y)W_1)W_2 + Y$$

$$Y = \color{red}{\mathrm{SelfAttn}}\big(\mathrm{Norm}(X)\big) + X$$

$$X = \mathrm{InputEmbed}(I)$$



Transformer architectures **MLP architectures**

$$Z = \mathrm{MLP}\big(\mathrm{Norm}(Y)\big) + Y$$
$$= \sigma(\mathrm{Norm}(Y)W_1)W_2 + Y$$

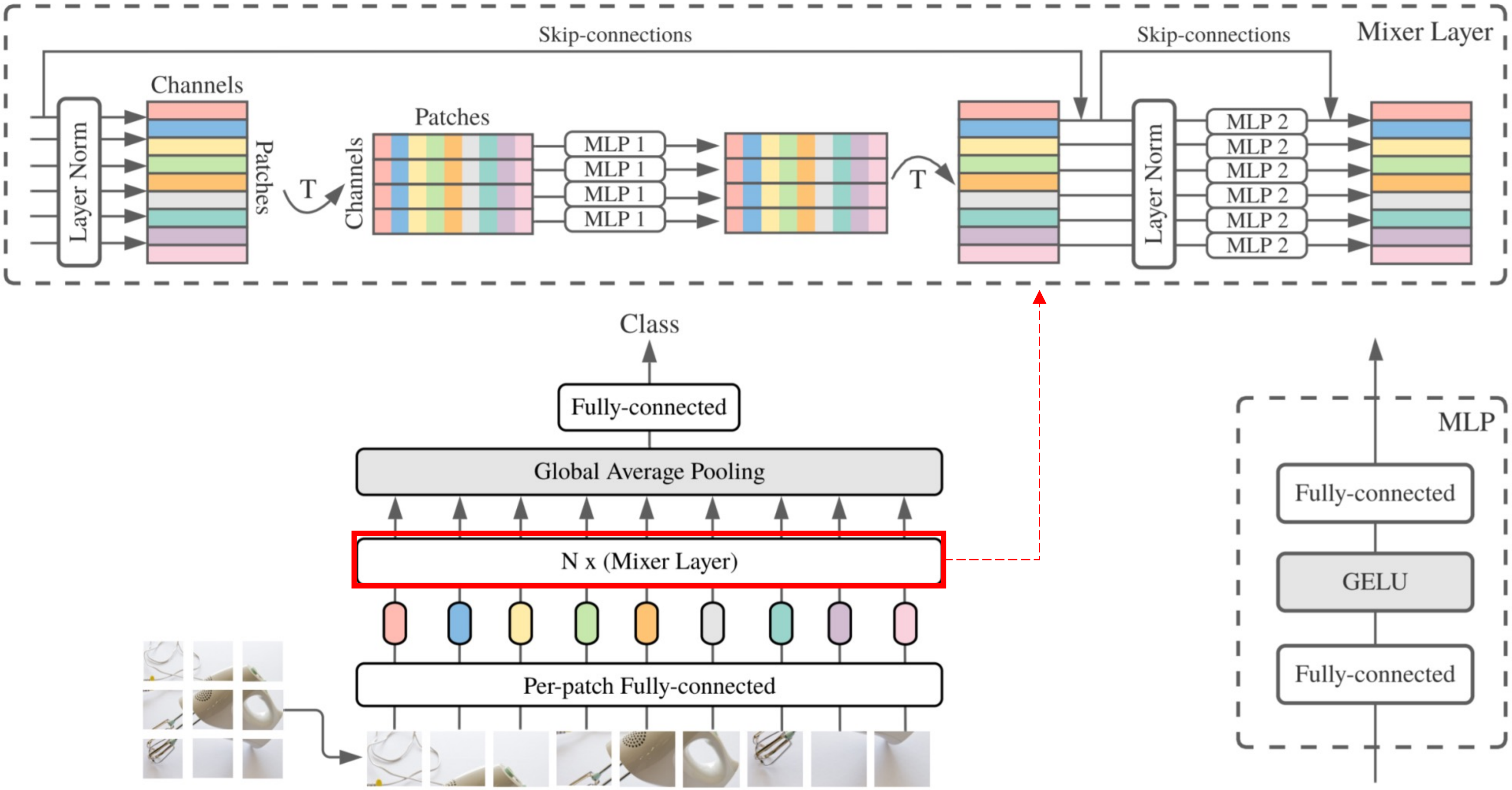
$$Y = \color{red}{\mathrm{MLP}}\big(\mathrm{Norm}(X)\big) + X$$

$$X = \mathrm{InputEmbed}(I)$$

*source : [Yu et al., 2021] MetaFormer is Actually What You Need for Vision, arxiv 2021

MLP-Mixer [Tolstikhin et al., 2021]

- Replacing the self-attention into MLP layers
- Removing position embedding & [class] token
- Mixing spatial & channel dimension separately



MLP-Mixer [Tolstikhin et al., 2021]

- Replacing the self-attention into MLP layers
- Removing position embedding & [class] token
- Mixing spatial & channel dimension separately

- **MLP-Mixer** shows competitive performances compared to Vision Transformers

	ImNet top-1	ReaL top-1	Avg 5 top-1	VTAB-1k 19 tasks	Throughput img/sec/core	TPUv3 core-days
Pre-trained on ImageNet-21k (public)						
HaloNet [51]	85.8	—	—	—	120	0.10k
Mixer-L/16	84.15	87.86	93.91	74.95	105	0.41k
ViT-L/16 [14]	85.30	88.62	94.39	72.72	32	0.18k
BiT-R152x4 [22]	85.39	—	94.04	70.64	26	0.94k
Pre-trained on JFT-300M (proprietary)						
NFNet-F4+ [7]	89.2	—	—	—	46	1.86k
Mixer-H/14	87.94	90.18	95.71	75.33	40	1.01k
BiT-R152x4 [22]	87.54	90.54	95.33	76.29	26	9.90k
ViT-H/14 [14]	88.55	90.72	95.97	77.63	15	2.30k

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 **network design and scaling**

- Many types of **CNN modules** are explored to capture detailed spatial information
    - Dilated and deformable convolution
    - Attention based modules
    - Many **observational study** supports the advantages of modern CNN architectures

- Recently, various types of architectures using **patch-based input shape** are explored
    - Transformer architecture: **Vision Transformer**
    - MLP architecture: **MLP-Mixer**

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/

[Yu et al., 2016] Yu, F., & Koltun, V.. (2016) Multi-Scale Context Aggregation by Dilated Convolutions. In International Conference on Learning Representations.
link : https://arxiv.org/abs/1511.07122

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

[Dai et al., 2017] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 764-773).
link : https://arxiv.org/abs/1703.06211v3

[Chen et al., 2017] Chen, L. C., Papandreou, G., Schroff, F., & Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.
link : https://arxiv.org/abs/1706.05587

[Howard et al., 2017] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.
link: https://arxiv.org/abs/1704.04861

References

[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

[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

[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-

References

[Hu et al., 2018] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7132-7141).
link : https://arxiv.org/abs/1709.01507

[Woo et al., 2018] Woo, S., Park, J., Lee, J. Y., & Kweon, I. S. (2018). Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 3-19).
link : https://arxiv.org/abs/1807.06521v2

[Tan et al., 2019] Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V. (2019). Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2820-2828).
link : https://arxiv.org/abs/1807.11626

[Dosovitskiy et al., 2021] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.
link : https://arxiv.org/abs/2010.11929

[Liu et al., 2021] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 10012-10022).
link :
https://openaccess.thecvf.com/content/ICCV2021/html/Liu_Swin_Transformer_Hierarchical_Vision_Transformer_Using_Shifted_Windows_ICCV_2021_paper.html

[Tolstikhin et al., 2021] Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., ... & Dosovitskiy, A. (2021). Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, *34*.
link : https://proceedings.neurips.cc/paper/2021/hash/cba0a4ee5ccd02fda0fe3f9a3e7b89fe-Abstract.html

References

[Heo et al., 2021] Heo, B., Yun, S., Han, D., Chun, S., Choe, J., & Oh, S. J. (2021). Rethinking spatial dimensions of vision transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 11936-11945).
link : https://arxiv.org/abs/2103.16302

[Li et al., 2021] Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z. H., ... & Yan, S. (2021). Tokens-to-token vit: Training vision transformers from scratch on imagenet. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 558-567).
link : https://arxiv.org/abs/2101.11986