Biological Applications of Deep Learning
Lecture 7

Alexander Schönhuth

UNIVERSITÄT BIELEFELD
Faculty of Technology

Bielefeld University

November 23, 2022

Contents Today

- Going deeper
  - Motivation
  - Datasets
- AlexNet
  - The breakthrough: 8 hidden layers
- VGG
  - Systematically investigating filter size
- Network in Network
  - Avoid fully connected layers altogether
- GoogLeNet
  - Let the network choose filter size by itself
- ResNet
  - Avoid learning the identity function
- Further: ResNeXt, DenseNet

Going Deeper – Architecture Development

MOTIVATION: REMINDER

THE UNIVERSAL APPROXIMATION THEOREM

- *Accuracy of approximation of (arbitrary) function f by NN $\hat{f}$ increases exponentially on increasing number of hidden layers*
- See [Cybenko, 1989, doi:10.1007/BF02551274], [Hornik, 1991, doi:10.1016/0893-6080(91)90009-T]
- See [Montufar et al., 2014]:
  `https://arxiv.org/pdf/1402.1869.pdf`
- Explanation:
  `http://neuralnetworksanddeeplearning.com/chap4.html`
- Further resources for the following:
  - `https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html`
  - `https://www.analyticsvidhya.com/blog/2017/08/10-advanced-deep-learning-architectures-data-scientists/`

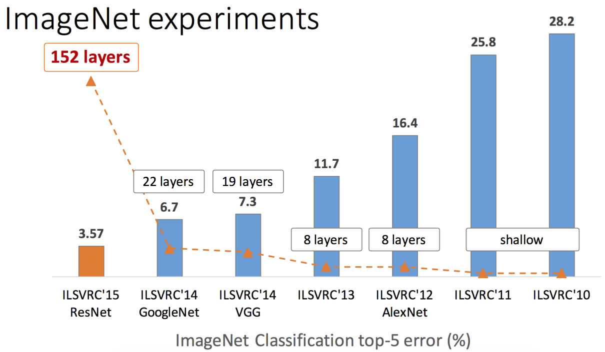
ImageNet and ILSVRC

Dataset and First Results

ImageNet examples: “beading plane”, “brown root rot fungus”, “scalded milk”, “common roundworm”

- *ImageNet dataset*: 16 million full color images; 20 000 categories
- *Starting point*: Le, Ranzato, Monga, Devin, Chen, Corrado, Dean & Ng: “Building high-level features using large scale unsupervised learning”, 2012, `https://ai.google/research/pubs/pub38115` achieved 15.3 % test accuracy
- *ILSVRC*: Image-Net Large-Scale Visual Recognition Challenge
  - *2012*: 1000 categories; Training 1.2 million images; Validation 50 000 images; Test 150 000 images

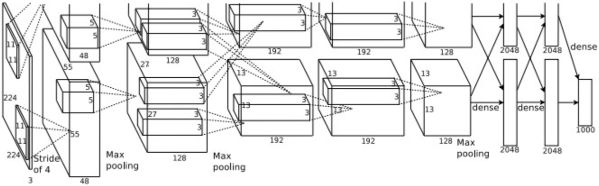
GOING DEEPER



https://icml.cc/2016/tutorials/icml2016_tutorial_deep_residual_networks_kaiminghe.pdf; Note: correct error rate for AlexNet is 15.4%

AlexNet

ALEXNET



AlexNet from Krizhevsky, Sutskever & Hinton, 2012,

`http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf`

- *Input layer*: $3 \times 224 \times 224$ (= 224×224 RGB pixels)
- *First hidden layer*: convolutional + max-pooling
    - 11×11 sized filters
    - stride length 4
    - 96 feature maps in total
    - 48 each are run on separate GPU;
    - max-pooling [also in later layers] is 3×3, 2 pixels apart (so overlap)

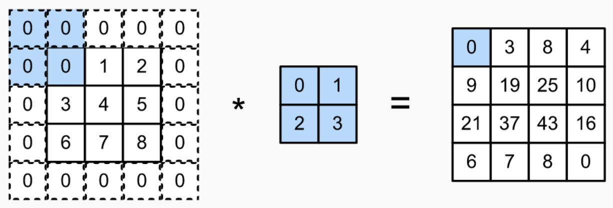
UNIVERSITÄT BIELEFELD

AlexNet

- *Second hidden layer*: convolutional + max-pooling
  - 5×5 sized filters
  - 256 feature maps in total
  - 128 each are run on separate GPU; each of those receive only 48 input channels
- *Third, fourth, fifth layer*: convolutional (no max-pooling)
  - *Third*: 384 feature maps, 3×3, and 256 input channels [with some inter-GPU communication]
  - *Fourth*: [384, 3×3, 192]; *Fifth*: [256, 3×3, 192]
- *Sixth, seventh layer*: fully connected, 4 096 neurons
- *Output layer*: 1000-unit softmax layer

Padding: Definition

Definition [Padding]:
Padding refers to pad images / feature maps with artificial zeros such that one can extend local receptive fields to their borders.



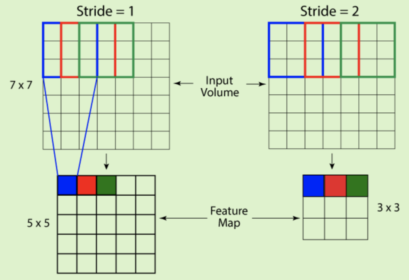
Extending 3x3 image (solid lines) to 5x5 image using padding (dashed lines), and applying 2x2 filters yields 4x4 feature map

From `http://d2l.ai/`

Stride: Definition

Definition [Stride]:

Stride is the amount of rows / columns (vertical / horizontal stride) to shift the local receptive field across the image / feature map.

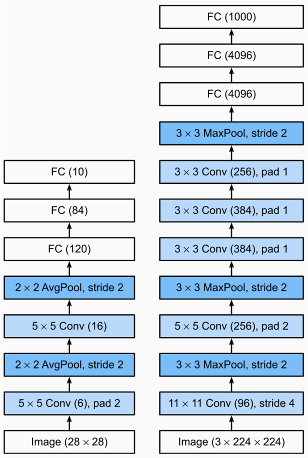


Applying 3x3 filter to 7x7 image, no padding, at identical vertical and horizontal stride.
Left: Stride 1 yields 5x5 feature map. *Right:* Stride 2 yields 3x3 feature map.

From `https://developersbreach.com/convolution-neural-network-deep-learning/`

AlexNet

Comparison with LeNet



Comparison: LeNet (left) versus AlexNet (right)

From `http://d2l.ai/`

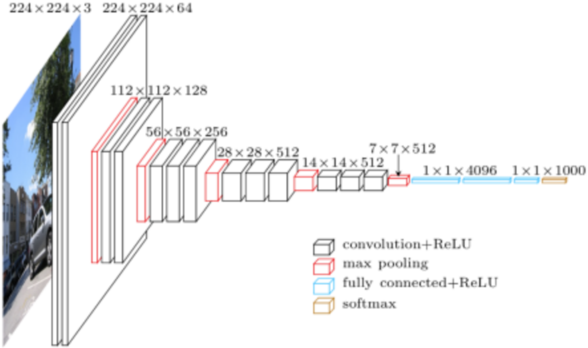
AlexNet

Further Features and ILSVRC performance

- Further architecture features:
  - Rectified linear neurons (original LeNet: sigmoid)
  - *Regularization*: L2 + dropout
  - *Optimization*: Momentum-based stochastic gradient descent
- *ILSVRC Performance*: Achieved 63.3 % accuracy; 84.7 % “top-5 accuracy” (if the correct label is among the top-5 predictions of the NN), followed by 73.8 % by the second-best performing contestant
- *Why important?* AlexNet’s amazing performance rate meant the “coming out” for CNNs in computer vision

VGG

VGG



VGG from Simonyan & Zisserman, 2014

`https://arxiv.org/pdf/1409.1556v6.pdf`

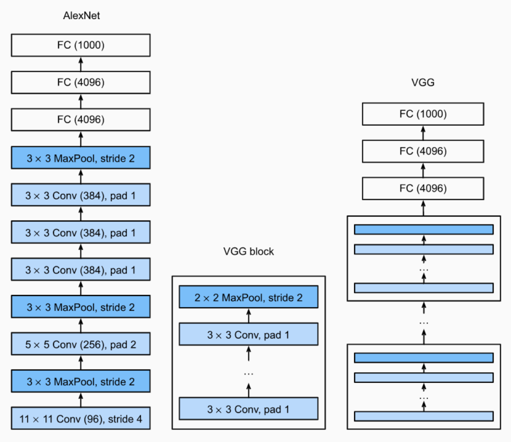
- From *Visual Geometry Group (VGG)* at Oxford University

VGG: Motivation

- Transition from individual layers to blocks of layers
- Systematic exploration of depth in combination with filter size
- Exemplary question: what is better?
  - One layer using 5×5-filters
  - Two layers each using 3×3-filters
  - ☞ Both bring together same pixels
  - But: one 5×5-filter uses as many parameters (25) as three 3×3-filters (27)
- A *VGG block* consists of sequence of
  - 3×3-convolutions at padding 1 (preserving width and length)
  - followed by 2×2-pooling at stride 2 (halving width and length)

VGG: Motivation II

Comparison with AlexNet



Comparison: AlexNet (left) versus VGG (right)

From `http://d2l.ai/`

VGG

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Different architectures tested in Simonyan & Zisserman, 2014

The yellow architecture yielded best performance in ILSVRC 2014

VGG

Features

- Uses only 3×3 filters;
  - argues that consecutive application of 3×3 filters yields virtual larger filters
  - For example, two 3×3 in a row yield virtual 5×5 filter
- 19(!) layers
- Less parameters than AlexNet
- ReLU layers and batch gradient descent

VGG

Features

- *Advantages*:
  - Top-5 accuracy: 92.7% (vs. 83.7% from AlexNet)
  - Good architecture for benchmarking
  - Pre-trained networks available
- *Disadvantage*: very slow to train, training needs two to three weeks
- *Why important*:
  - Reinforced that CNNs have to be sufficiently deep to work well
  - "Keep it deep. Keep it simple."

Network in Network (NiN)

NiN: Motivation

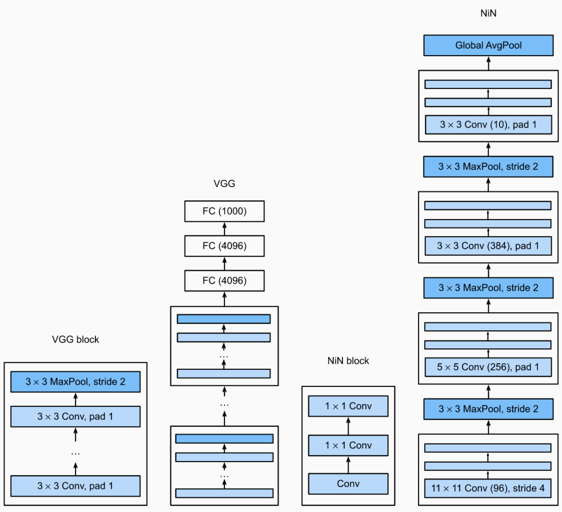
- AlexNet and VGG consume tremendous amount of parameters
- *Reason:* Fully connected layers towards the end of networks
- *Example:*
  - VGG-11, a simple VGG version, maintains a 25088×4096 matrix
  - The matrix alone occupies almost 400 MB of RAM in single precision (FP32)
- *Motivation:* Make CNN based analysis possible even on cell phones

NiN: Idea

- Fully connected layers provide additional non-linearity
- *Idea:* Re-establish non-linearity in other ways
- *Key Techniques:*
  - Make use of 1×1 convolution
    ☞ adds local non-linearity across channels
  - Use global *average pooling*
    ☞ Effective only because of the added non-linearity

NiN

Design



Comparison: VGG versus NiN

From `http://d2l.ai/`

NiN: Summary

- NiN avoids fully connected layers altogether
- *Consequence:* Dramatically less parameters than VGG and AlexNet
- 1×1 filters reduce parameters when implementing non-linearity across channels
- Global average pooling yields non-linearity across locations
  ☞ No learned parameter required at all
- Both 1×1 convolution and global average pooling influenced later CNN design significantly
- *Reference:*
  - "Network in network", Lin et al., 2014, `https://arxiv.org/pdf/1312.4400.pdf`

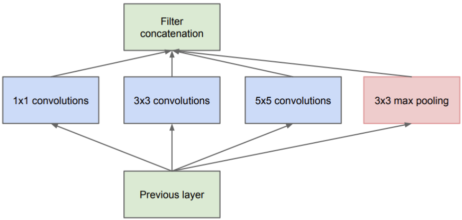
GoogLeNet a.k.a. Inception Network

How to choose filter size?

- In convolutional neural networks, how to determine the optimal filter size?
- 3x3, 5x5, 7x7? (for example); which one to use in which layer?
- Idea: offer all, and let the network learn by itself
- *Under any circumstances:*

"We need to go deeper"
[Inception, Christopher Nolan, 2010]

Inception Module

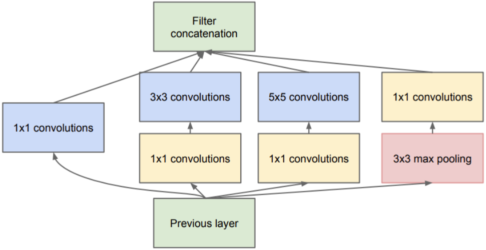


`https://arxiv.org/pdf/1409.4842.pdf`

(Szegedy et al., original paper)

- *Idea*: one layer looks like above
- During training, NN chooses way through filters by itself
- *Problem*: Non-negligible increase in parameters / channels

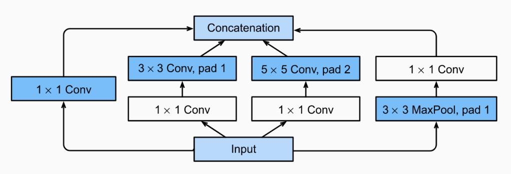
SOLUTION: 1X1 CONVOLUTION

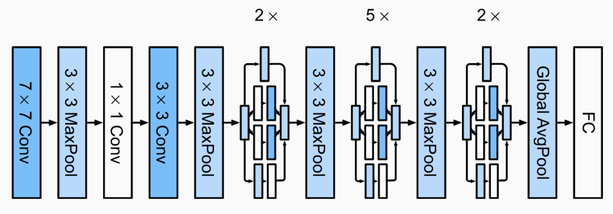


From `https://arxiv.org/pdf/1409.4842.pdf`

- 1x1 convolution mends the issue
- *Example:* Applying 20 $(60 \times 1 \times 1)$-filters to $60 \times 100 \times 100$-layer yields $20 \times 100 \times 100$-layer
- *Explanations:* `https://www.youtube.com/watch?v=HunX473yXEI`

GoogleLeNet: Architecture





Inception block (top) and GoogLeNet (a.k.a. Inception Network) (bottom)

From `http://d2l.ai/chapter_convolutional-modern/googlenet.html`

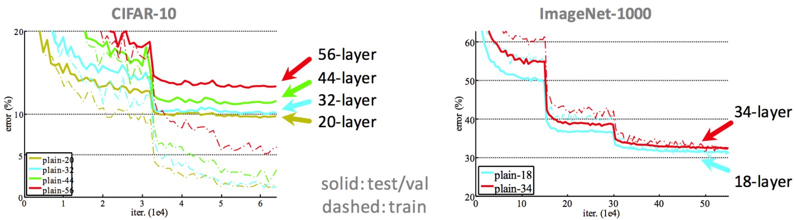
GoogLeNet: Features

- *Advantages*:
  - Top-5 accuracy: 93.4% (vs. 92.7% from VGG)
  - Performs various computational operations in parallel, and
  - stacks amazing 22 layers, while
  - being computationally reasonable: training needs a week
- *Why important*:
  - Points out that CNNs do not need to be stacked sequentially
  - Set the stage for further amazing architectures

ResNet

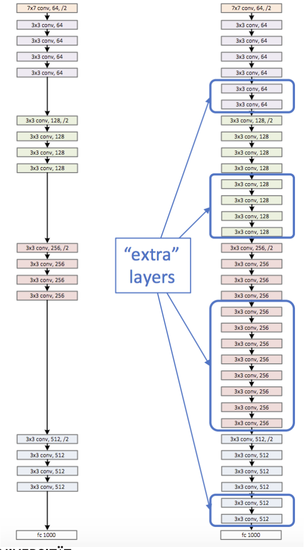
Motivation

Are we ready to simply stack layers?



- Training(!) accuracy decreases on stacking layers
- Despite having solved the vanishing gradient problem
- What else could be the reason?

Motivation: Practical Hints



- Solution by construction:
  - Original layers fixed: copied from shallower model
  - Extra layers: should learn identity function
  - However: training error increases in deeper model

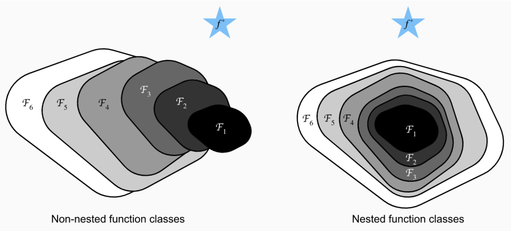
☞ Solvers have trouble learning identity function

  -

- In other words: deep models struggle with excessive layers

Motivation: Theoretical Hints

Are we ready to simply stack layers?



From `http://d2l.ai/chapter_convolutional-modern/resnet.html`

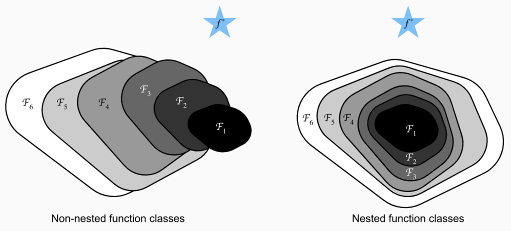
Reminder:

- Let f^* be true function, and $\mathcal{F}$ a (parameterized) class of functions
  - For example, $\mathcal{F}$ reflect certain NN's following a particular architecture
- Let $L(.,.)$ be a loss function. The task is to determine

$$\hat{f} = \arg\min_{f \in \mathcal{F}} L(f, f^*)$$

Motivation: Theoretical Hints

Are we ready to simply stack layers?



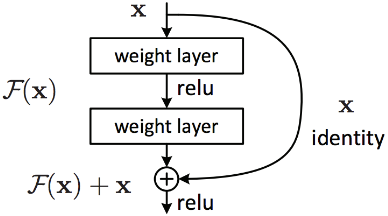
From `http://d2l.ai/chapter_convolutional-modern/resnet.html`

Insight:

- Consider class of functions $\mathcal{F}_2$ larger than $\mathcal{F}$
- $\hat{f} \in \mathcal{F}_2$ only guaranteed when $\mathcal{F} \subset \mathcal{F}_2$

Ensure that earlier solutions can be reproduced on increasing depth

ResNet Unit

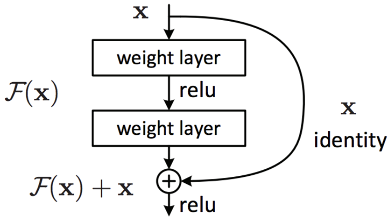


https://arxiv.org/pdf/1512.03385.pdf

(original paper)

- *Idea*: Bypass two layers by identity
- If identify function is to be learnt (or small modifications)
  - $F(x)$ is to be learnt as zero ☞ easy for NN's
  - ☞ Earlier solutions can be reproduced on stacking layers

ResNet Unit



https://arxiv.org/pdf/1512.03385.pdf

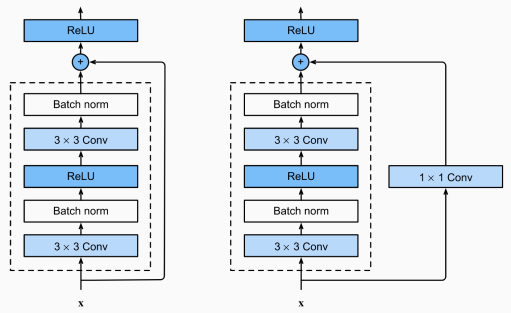
(original paper)

- Let network function be $G(x) = F(x) + x$
- *Name:*

$$F(x) = G(x) - x$$

referred to as *residual mapping*

ResNet Block

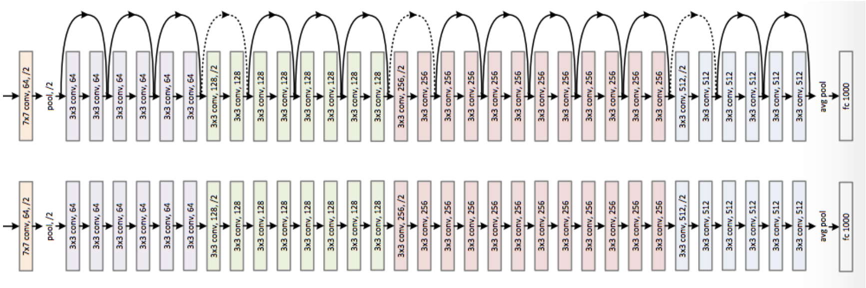


Full ResNet block without (left) or with (right) 1×1 convolution

From `http://d2l.ai/chapter_convolutional-modern/resnet.html`

- ResNet blocks follow VGG design, but add residual link
- Apply 1×1-convolution to match number of channels

FULL RESIDUAL NETWORK (RESNET)

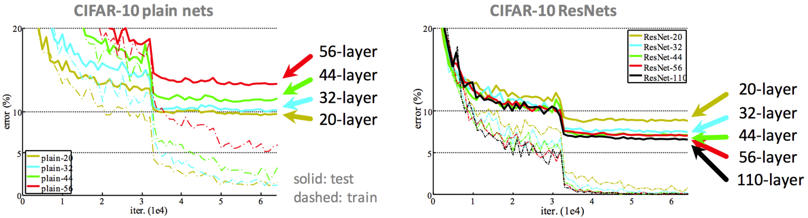


Plain network (bottom) vs. ResNet (top)

`https://arxiv.org/pdf/1512.03385.pdf`

(original paper)

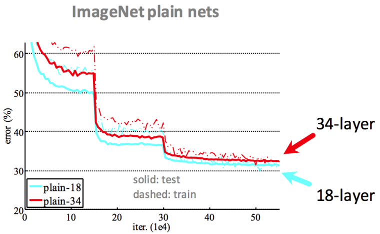
ResNet Results

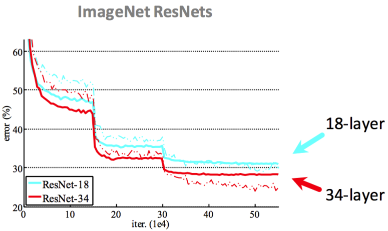


https://arxiv.org/pdf/1512.03385.pdf

(original paper)

ResNet Results





https://arxiv.org/pdf/1512.03385.pdf

(original paper)

ResNet

Features

- *Advantages*:
  - Top-5 accuracy: 96.4(!)% (vs. 93.4% from GoogLeNet)
  - “Ultra-deep”: 152(!) layers
  - Just quite simply the best (back in 2016)
- *Why important*:
  - Breakthrough in going ultra-deep
  - Stacking ResNet units do not require extra performance upgrades (while necessary with ordinary CNNs)

ResNeXt

ResNeXt: Motivation I

- *Goal:* Increase level of non-linearity; possible strategies:
  - Increase depth (explored!)
  - Increase size of convolution filters (explored!)
  - Increase number of channels (explore here!)
- *Drawback:* c_i input and c_o output channels amount to

$$\mathcal{O}(c_i \cdot c_o)$$

  computational cost
- *Idea – Grouped Convolution:*
  - Break up convolution from c_i to c_o channels into ...
  - ... g groups, each reflecting convolution from c_i/g to c_o/g channels

ResNeXt: Motivation II

- *Idea – Grouped Convolution:*
  - Break up convolution from c_i to c_o channels into ...
  - ... g groups, each reflecting convolution from c_i/g to c_o/g channels
- *Cost:* Reduction from $\mathcal{O}(c_i \cdot c_o)$ to

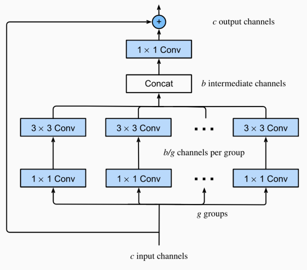
  $$\mathcal{O}(g \cdot c_i/g \cdot c_o/g) = \mathcal{O}(c_i \cdot c_o/g)$$

  so speed-up by factor of g
- *Number of parameters* also reduced by factor of g:
  - From matrices of size $c_i \times c_o$ to ...
  - ... g matrices of size $c_i \cdot c_o/g^2$

ResNeXt: Challenge

- *Grouped Convolution – Challenge:* No exchange of information between groups
- *Solution:* Sandwich grouped convolution in between layers of 1×1-convolution
  - Before, from c to b/g channels, g times
    ☞ cost $\mathcal{O}(g \cdot c \cdot b/g) = \mathcal{O}(c \cdot b)$
  - After, from b back to c channels
    ☞ cost $\mathcal{O}(b \cdot c)$ as well
- *Additional benefit:* Group convolutions at cost $\mathcal{O}(b^2/g)$
- *Residual connection:* 1×1 convolution; cost $\mathcal{O}(c^2)$

ResNeXt Block



Full ResNeXt block

From http://d2l.ai/chapter_convolutional-modern/resnet.html

- Idea dates back to AlexNet: Distribute convolutions across GPU's
- Although less complex, no losses in accuracy observed
- *Reference:* "Aggregated residual transformations for deep neural networks.", Xie et al., CVPR 2017, https://openaccess.thecvf.com/content_cvpr_2017/papers/Xie_Aggregated_Residual_Transformations_CVPR_2017_paper.pdf

DenseNet

DenseNet: Motivation I

- *Inspiration:* Recall *Taylor expansion*

$$f(x) = f(0) + x\left[f'(0) + x\left[\frac{f''(0)}{2!} + x\left[\frac{f'''(0)}{3!} + \dots\right]\right]\right] \tag{1}$$

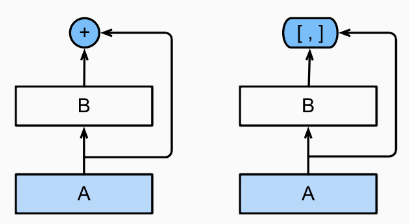
- ResNet decomposes network function $f(x)$ into (earlier notation changed!)

$$f(\mathbf{x}) = \mathbf{x} + g(\mathbf{x}) \tag{2}$$

  into simple linear term ($\mathbf{x}$) and complex non-linear one ($g(\mathbf{x})$)

- *DenseNet – Idea:* Try more complex decompositions of $f(\mathbf{x})$

DenseNet Block: Design



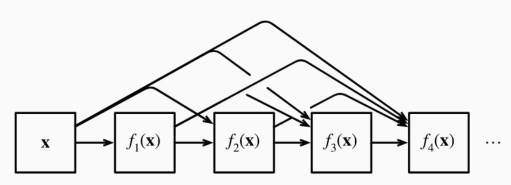
Instead of adding (ResNet; left) concatenate output (DenseNet; right)

From `http://d2l.ai/chapter_convolutional-modern/densenet.html`

- Concatenate output of branches instead of adding it
- This yields formula

$$\mathbf{x} \to [\mathbf{x}, f_1(\mathbf{x}), f_2([\mathbf{x}, f_1(\mathbf{x})]), f_3([\mathbf{x}, f_1(\mathbf{x}), f_2([\mathbf{x}, f_1(\mathbf{x})])]), \ldots] \qquad (3)$$

DenseNet: Architecture



Chain of DenseNet layers.

From `http://d2l.ai/chapter_convolutional-modern/resnet.html`

- *Characteristics:*
  - Number of features increases on increasing depth
  - Last layer connected to all layers before
- *Name:* Dependency graph is *dense*

DenseNet: Challenge

- *Challenge:* Keep number of features at affordable level
- *Solution:* DenseNet has *dense blocks* and *transition layers*
- *Dense blocks:*
  - Follow modified version of ResNet blocks, implementing batch normalization, activation and convolution
  - Consist of multiple convolution blocks, each having same number of output channels
  - During forward propagation, dimensionality increases due to concatenation
- *Transition layers:*
  - Reduce number of channels by 1×1-convolution
  - Halves height and width via average pooling at stride 2

References

- http://neuralnetworksanddeeplearning.com/, Chapter 6, “Recent progress in image recognition”
- http://d2l.ai/chapter_convolutional-modern/index.html, Chapter 8
- Recurrent neural networks

Outlook

- Biological Applications I
  - Framing problems as image recognition problems
- Biological Applications II
  - Convolving sequence data

Thanks for your attention