



**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Architectures

**A. Maier, V. Christlein, K. Breininger, S. Vesal, F. Meister, C. Liu, S. Gündel, S. Jaganathan, N. Maul,  
M. Vornehm, L. Reeb, F. Thamm, M. Hoffmann, C. Bergler, F. Denzinger, W. Fu, B. Geissler, Z. Yang**  
Pattern Recognition Lab, Friedrich-Alexander-Universität Erlangen-Nürnberg  
May 19, 2020



# Outline

**Early Architectures**

**Deeper Models**

**Learning Architectures**

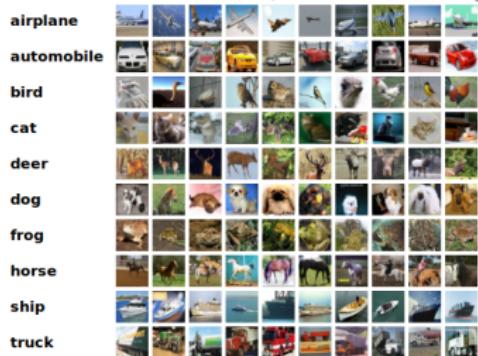
# Datasets for Object Detection / Image Classification

## ImageNet Dataset [11]

- 1000 classes
- > 14 Mio. images
- Subsets used for ImageNet Large Scale Visual Recognition Challenges (ILSVRC)
- Natural images of varying size

## CIFAR 10 / 100

- 10 / 100 classes
- 50 k training / 10 k testing
- $32 \times 32$  images





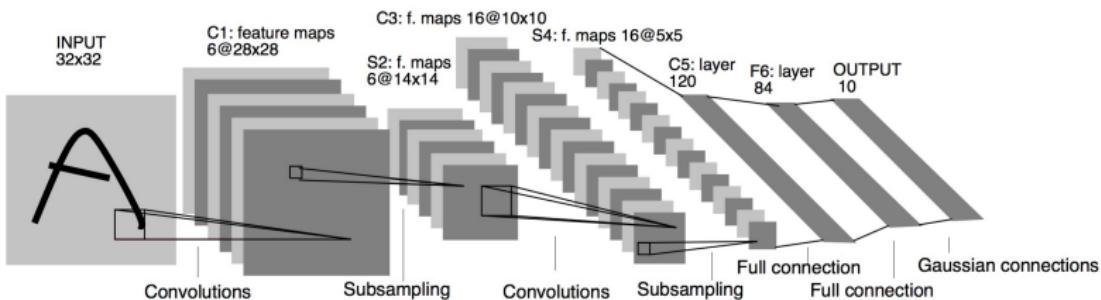
**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Early Architectures



## LeNet-5 (1998) [9]



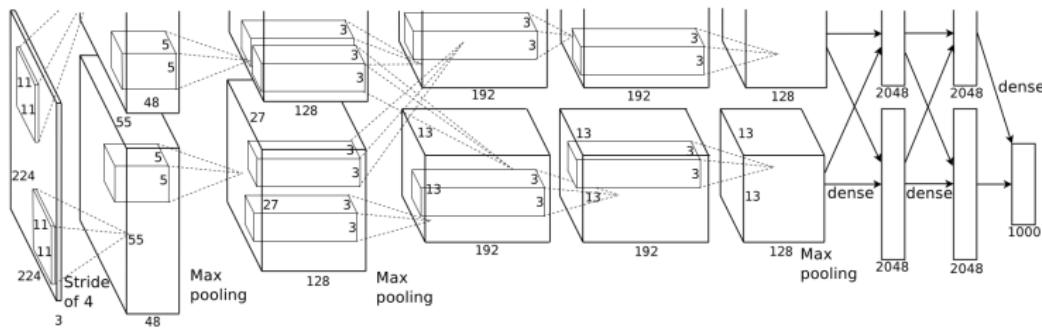
### Key features

- Convolution for spatial features
- Subsampling using average pooling
- Non-linearity: tanh
- Sparse connectivity between S2 and C3 (efficiency, robustness)
- MLP as final classifier
- Sequence: Convolution, pooling, non-linearity
- Foundation for many other architectures

('• Technique still used in recent architectures)

Source: [9]

## AlexNet (2012) [7]



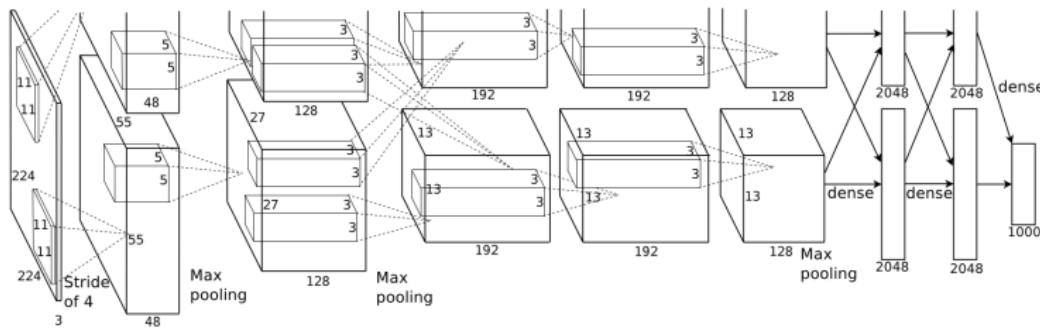
Winner of the ImageNet 2012 challenge  $\Rightarrow$  Breakthrough of CNNs

### Key features I

- 8 Layers
- Use of GPU(s) to reduce training time
- Overlapping max pooling (stride: 2, size: 3)
- Non-linearity: ReLU

Source: [7]

## AlexNet (2012) [7]



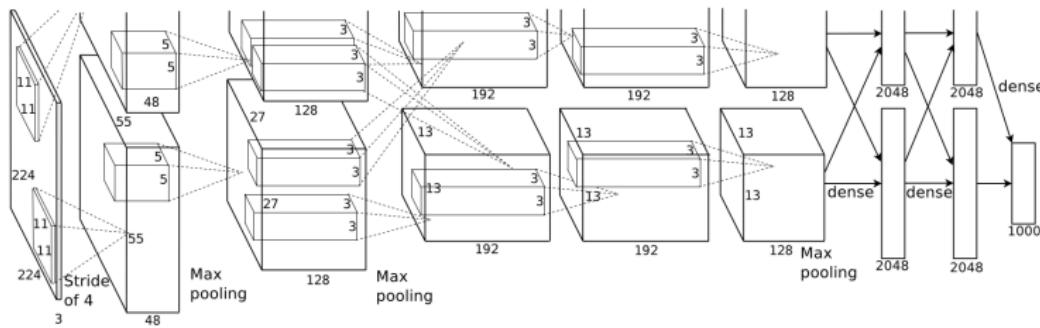
Winner of the ImageNet 2012 challenge  $\Rightarrow$  Breakthrough of CNNs

### Key features II

- Combat overfitting:
  - Dropout w.  $p = 0.5$  in the first two FC layers
  - Data augmentation (random transformations, random intensity variation)

Source: [7]

## AlexNet [7]



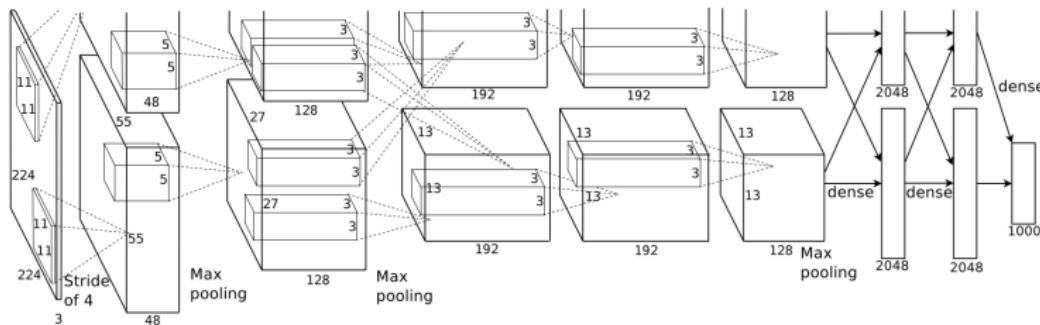
Winner of the ImageNet 2012 challenge ⇒ Breakthrough of CNNs

## Key features III

- Learning: mini-batch SGD w. momentum (0.9) + ( $L_2$ ) weight decay ( $5 \cdot 10^{-5}$ )
- Weight initialization:  $\mathcal{N}(0, 0.01)$

Source: [7]

## AlexNet [7]



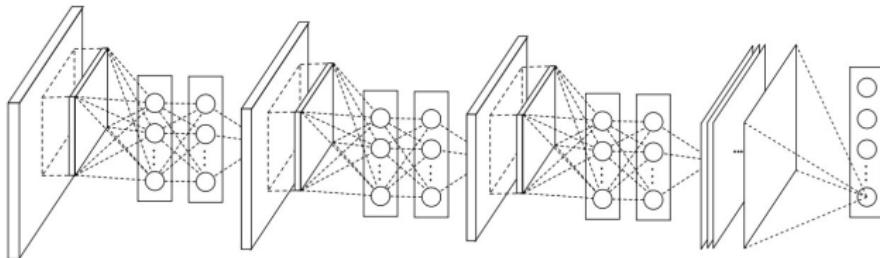
Winner of the ImageNet 2012 challenge ⇒ Breakthrough of CNNs

### Key features III

- Learning: mini-batch SGD w. momentum (0.9) + ( $L_2$ ) weight decay ( $5 \cdot 10^{-5}$ )
- Weight initialization:  $\mathcal{N}(0, 0.01)$
- Historical note: Small GPUs → network split across two GPUs

Source: [7]

## Network In Network [10]

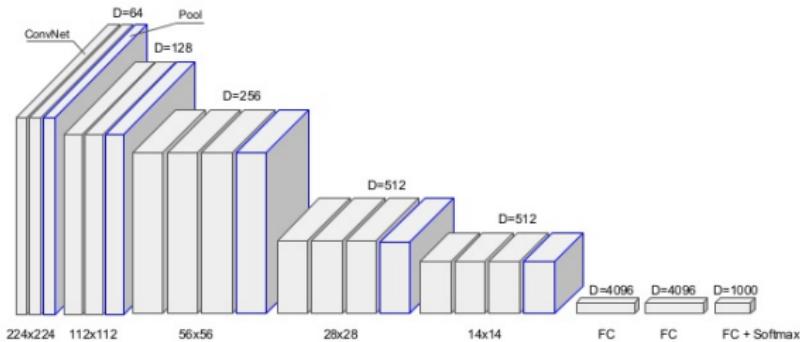


### Key features

- $1 \times 1$  filters
  - Conventional conv layers only learn linear functions of input
  - Connect conv layers by FC layers that can learn non-linear functions
  - Equivalent to FC layer: conv layer with  $1 \times 1$  filters
  - Very few parameters, shared across all activations
- Global (spatial) average pooling as last layer
  - Less prone to overfitting than final FC layers

Source: [10]

## VGG Network (Visual Geometry Group – University of Oxford) [12]

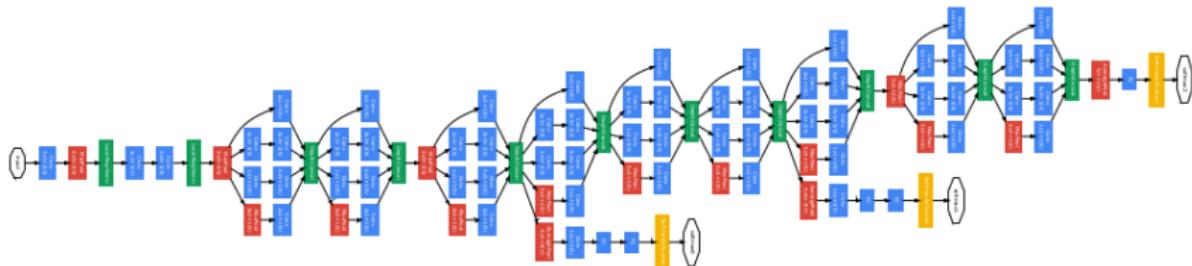


## Key features

- Small kernel sizes in each convolution ( $3 \times 3$ )
    - Combination of multiple smaller kernels emulate larger receptive fields
  - 16 / 19 layers, max pooling between some layers (stride: 2, size: 2)
  - Learning procedure similar to AlexNet
  - hard to train (in practice: pre-training with shallower networks)

Source: <https://www.slideshare.net/holbertonschool/deep-learning-class-2-by-louis-monier>

## GoogleNet (Inception-v1) [14]

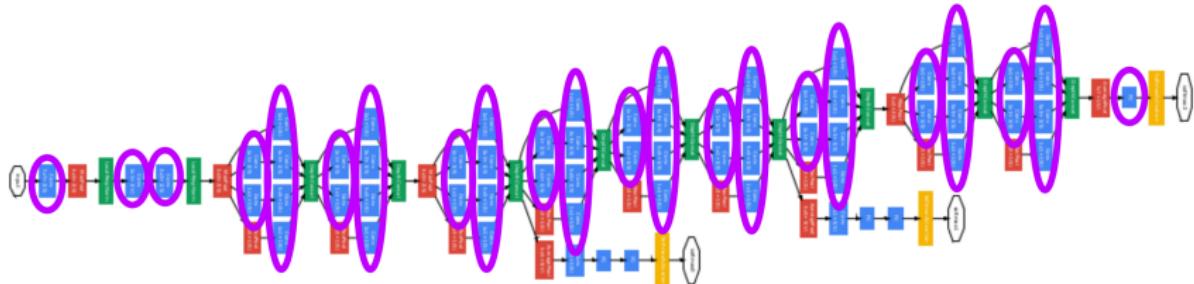


## Goal

- Network design with embedded hardware in mind
- maximum 1.5 billion MAD (multiply-add) operations at inference time

Source: [14]

## GoogleNet (Inception-v1) [14]

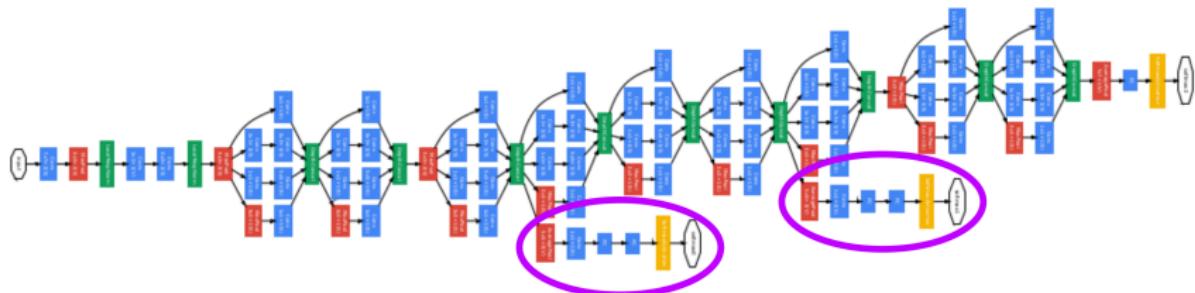


### Key features

- 22 layers + global average pooling as final layer

Source: [14]

## GoogleNet (Inception-v1) [14]

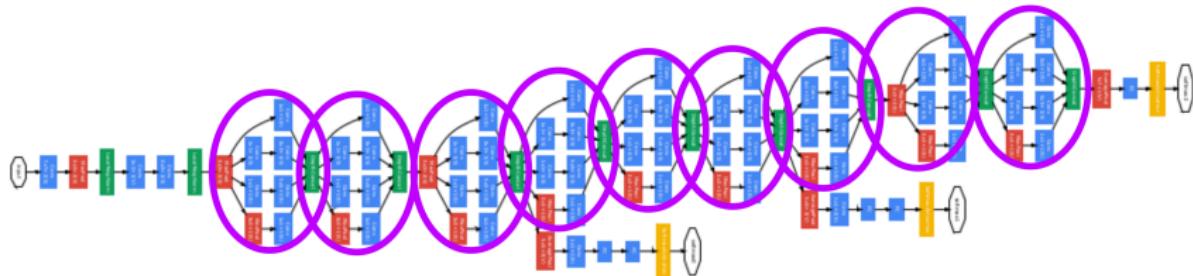


### Key features

- 22 layers + global average pooling as final layer
- Auxiliary classifiers (only at training): error weighted by 0.3 added to global error → additional regularization
- No fully connected layers (except for linear layer and auxiliary networks)

Source: [14]

## GoogleNet (Inception-v1) [14]

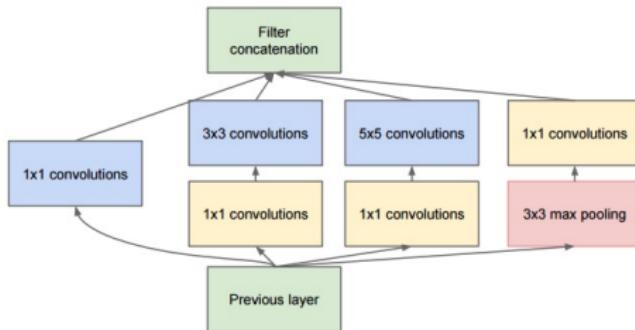


### Key features

- 22 layers + global average pooling as final layer
- Auxiliary classifiers (only at training): error weighted by 0.3 added to global error → additional regularization
- No fully connected layers (except for linear layer and auxiliary networks)
- Inception modules

Source: [14]

# GoogleNet (Inception-v1) [14]



## Inception Module

- Derived from NiN concept
- Parallel filter combinations (split-transform-merge strategy)
  - Network decides filter size by itself
- $1 \times 1$  filters serve as “bottleneck layer”
- Representational power of large and dense layers but with much lower computational complexity

Source: [14]

## Bottleneck Layer

Features are correlated, redundancy can be removed by  $1 \times 1$  filters

Example:

- Before: 256 input feature maps, 256 output feature maps,  $3 \times 3$  convolution  
→  $256 \times 3 \times 3 \times 256 \approx 600\text{k MAD}$
- Instead: Reduce number of feature maps that have to be convolved, e.g. 64

$$256 \times 1 \times 1 \times 64 \qquad \qquad \approx 16,000$$

$$64 \times 3 \times 3 \times 64 \qquad \qquad \approx 36,000$$

$$64 \times 1 \times 1 \times 256 \qquad \qquad \approx 16,000$$

→  $\approx 70\text{k MAD}$

**NEXT TIME  
ON DEEP LEARNING**



**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Architectures - Part 2

**A. Maier, V. Christlein, K. Breininger, S. Vesal, F. Meister, C. Liu, S. Gündel, S. Jaganathan, N. Maul,  
M. Vornehm, L. Reeb, F. Thamm, M. Hoffmann, C. Bergler, F. Denzinger, W. Fu, B. Geissler, Z. Yang**  
Pattern Recognition Lab, Friedrich-Alexander-Universität Erlangen-Nürnberg  
May 19, 2020





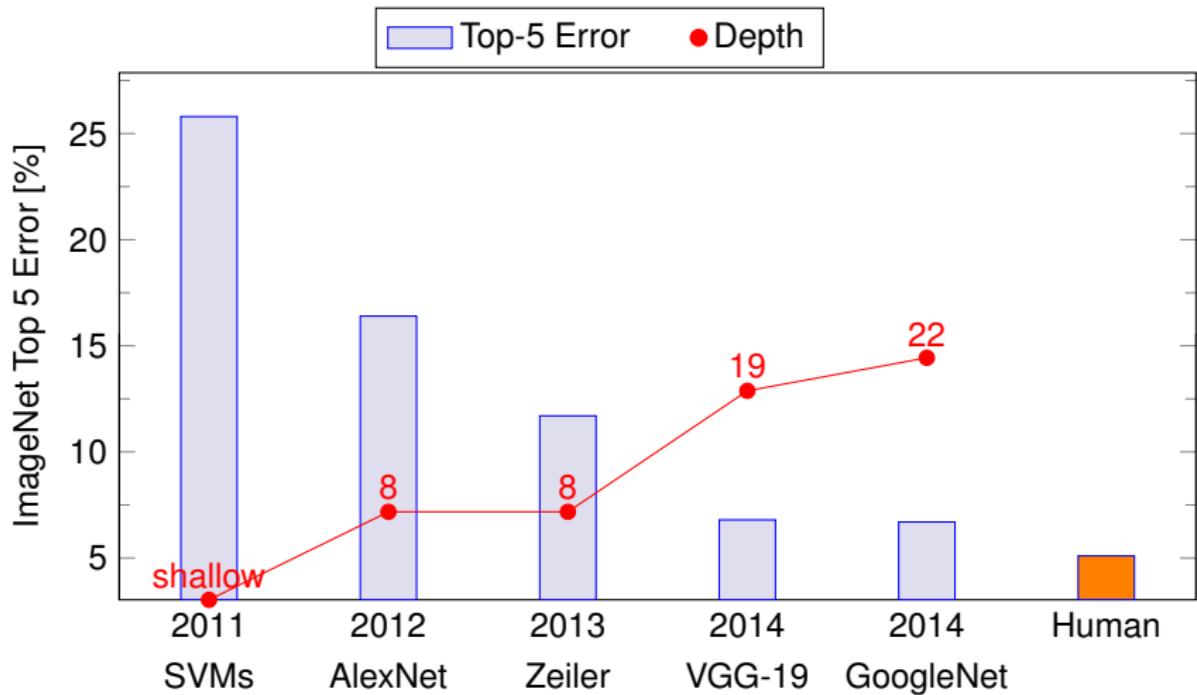
**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Deeper Models

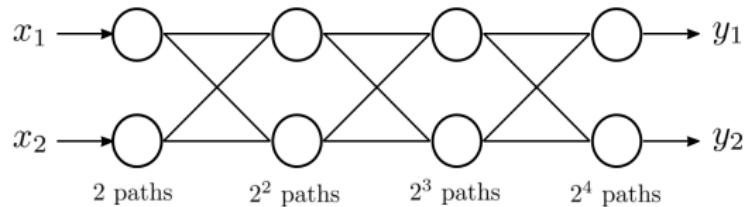


## Evolution of Depth


 Source: [image-net.org](http://image-net.org), Russakovsky et al. 2015

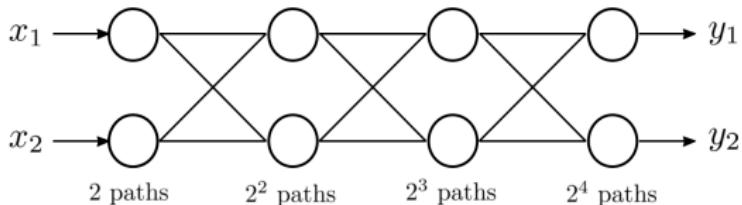
## Advantages of Deeper Networks

- Exponential feature reuse

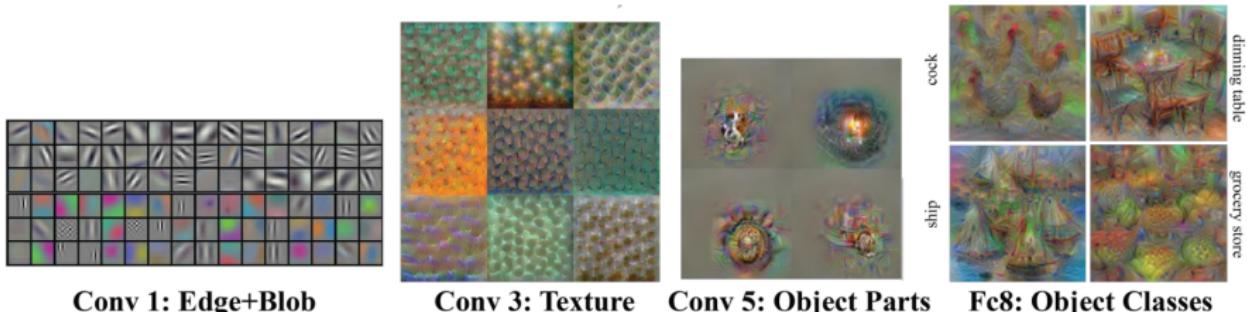


## Advantages of Deeper Networks

- Exponential feature reuse

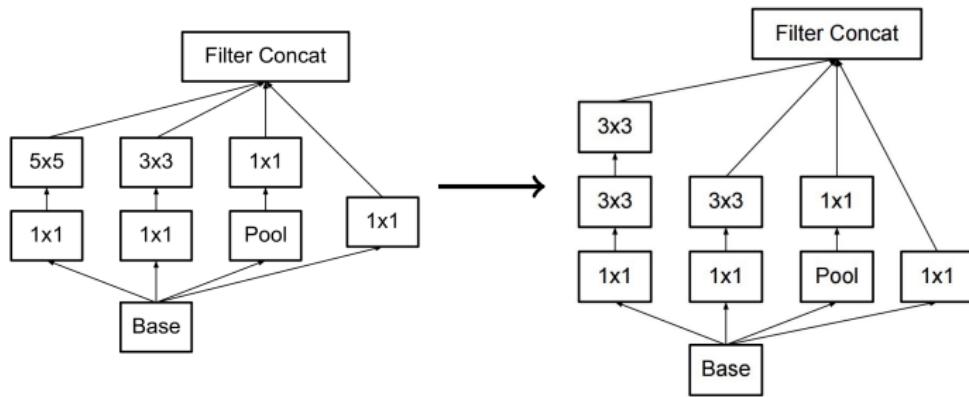


- Increasingly abstract features



Source: [http://vision03.csail.mit.edu/cnn\\_art/index.html](http://vision03.csail.mit.edu/cnn_art/index.html)

## Inception-v2 [15]

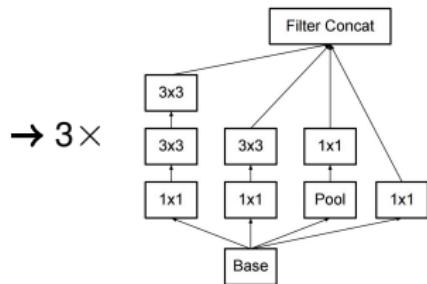


### Key features

- Change basic inception layer: Replace  $7 \times 7$  and  $5 \times 5$  filters by multiple  $3 \times 3$  convolutions.

Source: [15]

## Inception-v2 [15]

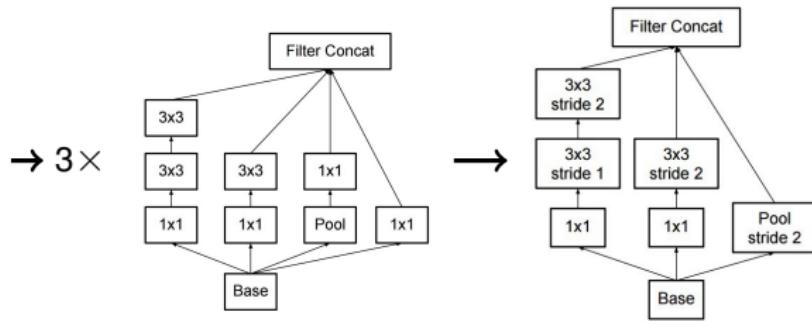


### Key features II

- 42 layers – start with several  $3 \times 3$  convolutions and 3 modified inception modules

Source: [15]

## Inception-v2 [15]

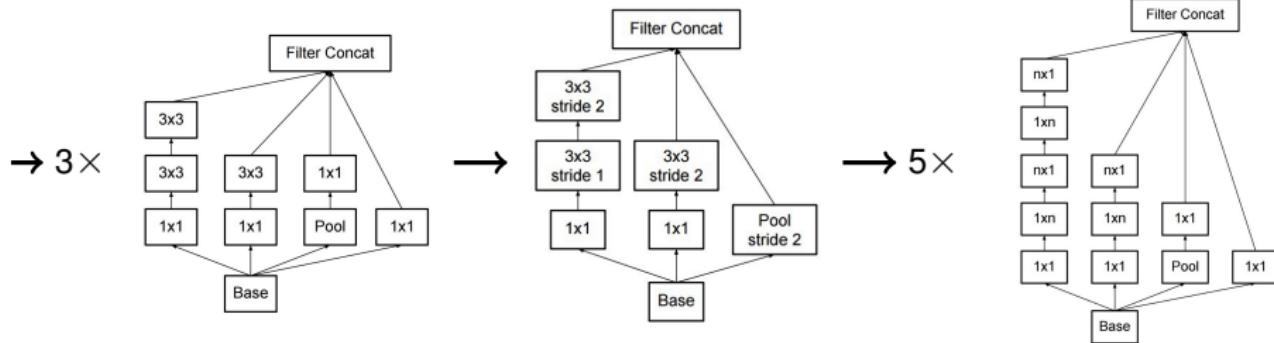


### Key features II

- 42 layers – start with several  $3 \times 3$  convolutions and 3 modified inception modules
- Efficient grid size reduction

Source: [15]

## Inception-v2 [15]



### Key features II

- 42 layers – start with several  $3 \times 3$  convolutions and 3 modified inception modules
- Efficient grid size reduction
- 5 modules of flattened convolutions ( $n = 7$ )
- Efficient grid size reduction + average pooling + softmax

Source: [15]

## Inception-v3 [15]

Inception-v2 +

- RMSProp
- Batch-normalization also in the FC layers of auxiliary classifiers
- Label-smoothing regularization

## Label-smoothing regularization

Standard label distribution:

$$q(k|x) = \delta_{k,y}$$

$x$ : training sample,  $k \in \{1, \dots, K\}$ : specific label,  $y$ : ground truth label

- For Softmax to predict exactly 0 / 1,  $|\text{activations}| \mapsto \infty$
- Continue to learn larger and larger weights, making more extreme predictions
- Use weight decay and/or label-smoothing

## Label-smoothing regularization

Standard label distribution:

$$q(k|x) = \delta_{k,y}$$

$x$ : training sample,  $k \in \{1, \dots, K\}$ : specific label,  $y$ : ground truth label

- For Softmax to predict exactly 0 / 1,  $|\text{activations}| \mapsto \infty$
- Continue to learn larger and larger weights, making more extreme predictions
- Use weight decay and/or label-smoothing

### Label-smoothing [8]

Exchange label distribution with

$$q'(k|x) = (1 - \epsilon)\delta_{k,y} + \epsilon u(k)$$

where authors chose:  $u(k) = 1/K$ ,  $\epsilon = 0.1$

- + Prevent hard probabilities without discouraging correct classification.

**NEXT TIME  
ON DEEP LEARNING**

**Problems with going deeper  
... why not just stack more layers?**



**FAU**

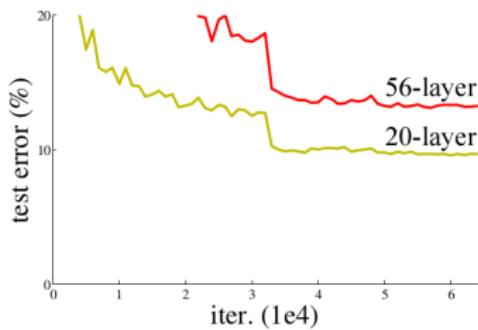
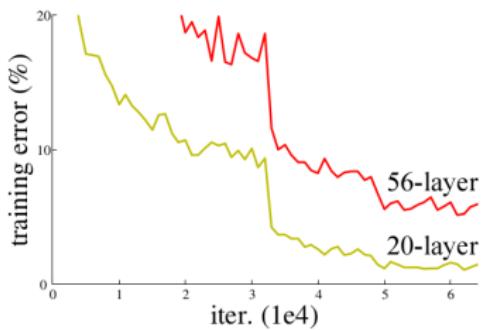
FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Architectures - Part 3

**A. Maier, V. Christlein, K. Breininger, S. Vesal, F. Meister, C. Liu, S. Gündel, S. Jaganathan, N. Maul,  
M. Vornehm, L. Reeb, F. Thamm, M. Hoffmann, C. Bergler, F. Denzinger, W. Fu, B. Geissler, Z. Yang**  
Pattern Recognition Lab, Friedrich-Alexander-Universität Erlangen-Nürnberg  
May 19, 2020

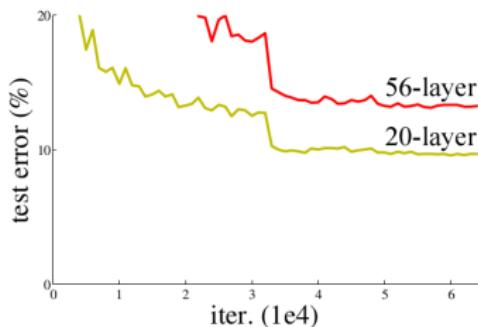
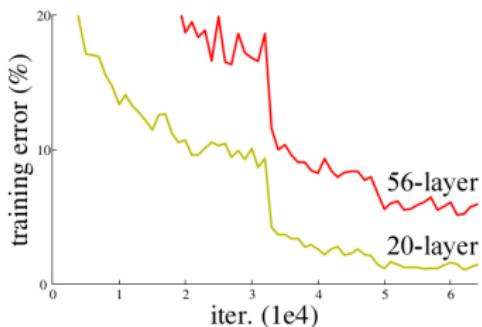


## Degradation of Training Error



Deeper models tend to have higher **training & test error** than shallower models  
 → Not just caused by overfitting!

## Degradation of Training Error

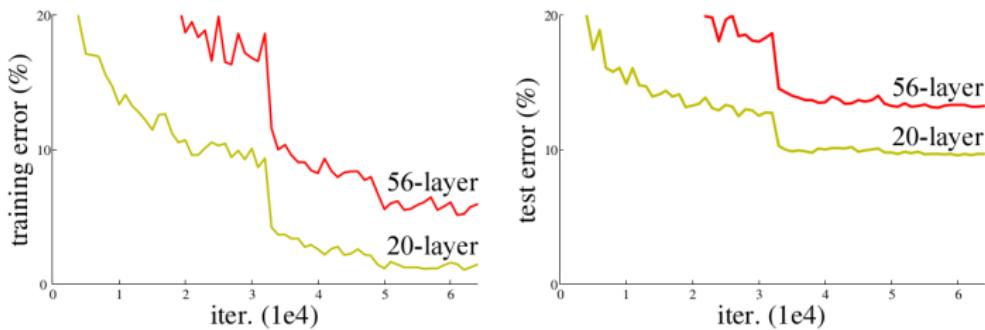


Deeper models tend to have higher **training & test error** than shallower models  
 → Not just caused by overfitting!

## Possible Reasons

- Vanishing gradient problem
  - ReLU (or successors)
  - Proper initialization

## Degradation of Training Error

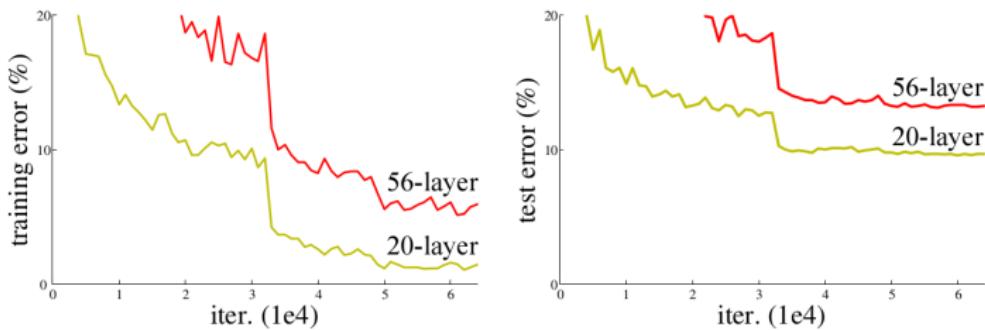


Deeper models tend to have higher **training & test error** than shallower models  
 → Not just caused by overfitting!

## Possible Reasons

- Vanishing gradient problem
  - ReLU (or successors)
  - Proper initialization
- Internal co-variate shift
  - Batch normalization
  - ELU / SELU

## Degradation of Training Error



Deeper models tend to have higher **training & test error** than shallower models  
 → Not just caused by overfitting!

## Possible Reasons

- Vanishing gradient problem
  - ReLU (or successors)
  - Proper initialization
- Degradation problem: poor propagation of activations and gradients
- Internal co-variate shift
  - Batch normalization
  - ELU / SELU

Source: [2]

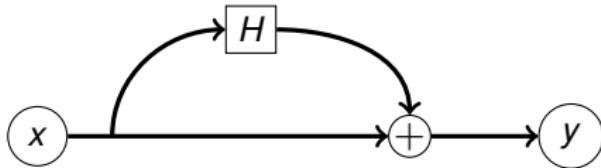
## (One) Solution: Residual Units

## Residual Units [2], [3]

Idea: Simplify “identity solution”

- Non-residual nets: learn mapping  $F(x)$
- Instead: learn residual mapping:

$$H(x) = F(x) - x \Leftrightarrow F(x) = H(x) + x$$



## Residual Block [2], [3]

- General form of the  $l$ -th residual unit over  $K$  layers:

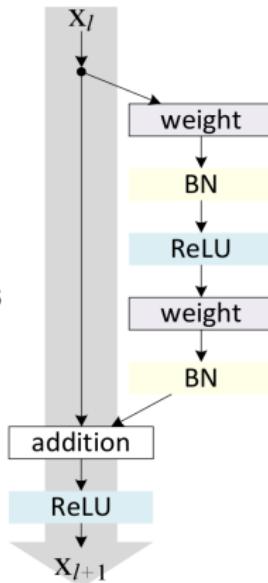
$$\mathbf{x}_{l+1} = h(g(\mathbf{x}_l) + H_{l+1}(\mathbf{x}_l, \{\mathbf{W}_{l+1,K}\}))$$

$\mathbf{x}_l, \mathbf{x}_{l+1}$  : input, output activations

$$\{\mathbf{W}_{l+1,K}\} = \{\mathbf{W}_{l+1,k} \mid 1 \leq k \leq K\}$$

$g, h$  : activation functions

- BN: Batch normalization
- Typically  $K = 2$  or  $K = 3$
- Original:  $g$ : identity,  $h$ : ReLU



Source: [3]

## Residual Block [2], [3]

- General form of the  $l$ -th residual unit over  $K$  layers:

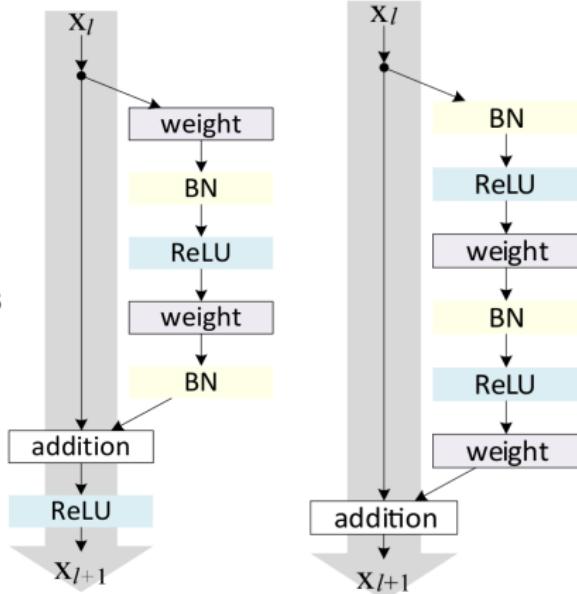
$$\mathbf{x}_{l+1} = h(g(\mathbf{x}_l) + H_{l+1}(\mathbf{x}_l, \{\mathbf{W}_{l+1,K}\}))$$

$\mathbf{x}_l, \mathbf{x}_{l+1}$  : input, output activations

$$\{\mathbf{W}_{l+1,K}\} = \{\mathbf{W}_{l+1,k} \mid 1 \leq k \leq K\}$$

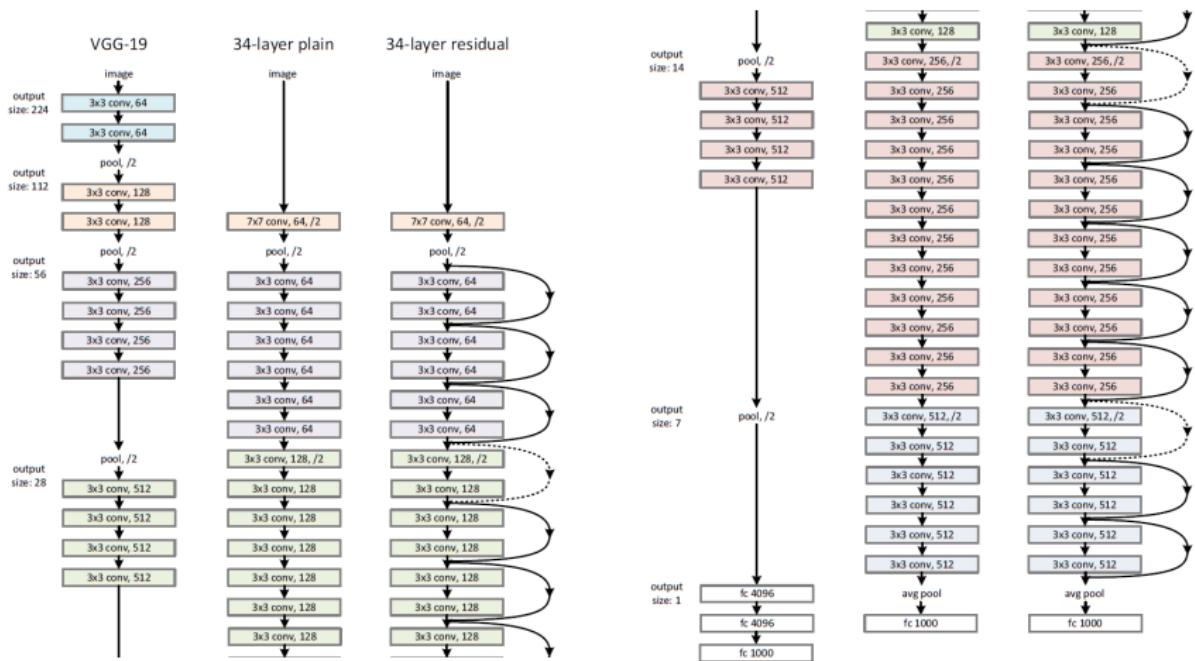
$g, h$  : activation functions

- BN: Batch normalization
- Typically  $K = 2$  or  $K = 3$
- Original:  $g$ : identity,  $h$ : ReLU
- Better:  $g$  and  $h$ : identity  $\Rightarrow$  pre-activation



Source: [3]

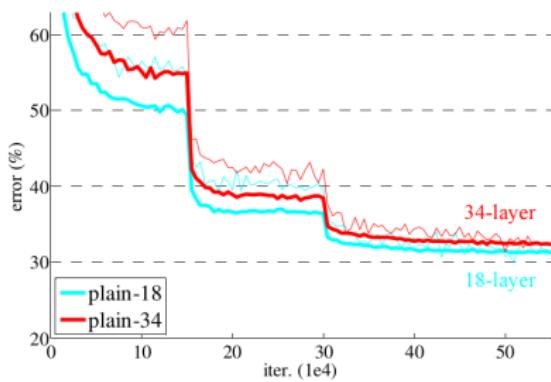
## Residual Networks [2], [3]



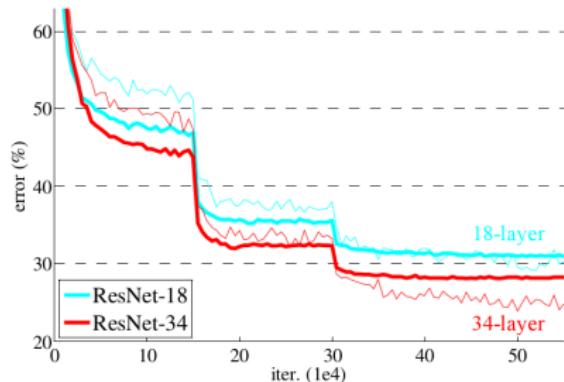
VGG: 19.6 billion FLOPs, Plain / ResNet: 3.6 billion FLOPs

Source: [2]

## Residual Networks [2], [3]



Plain networks

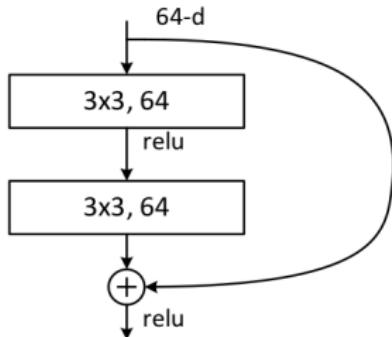


Residual networks

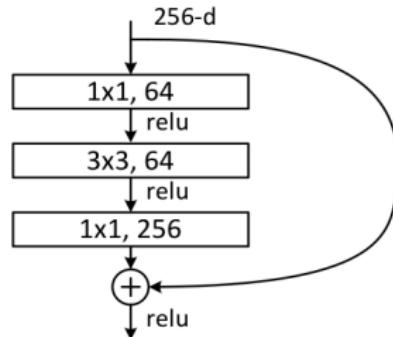
- Training / validation error of deeper nets is now lower!

Source: [2]

## Bottleneck Residual Block



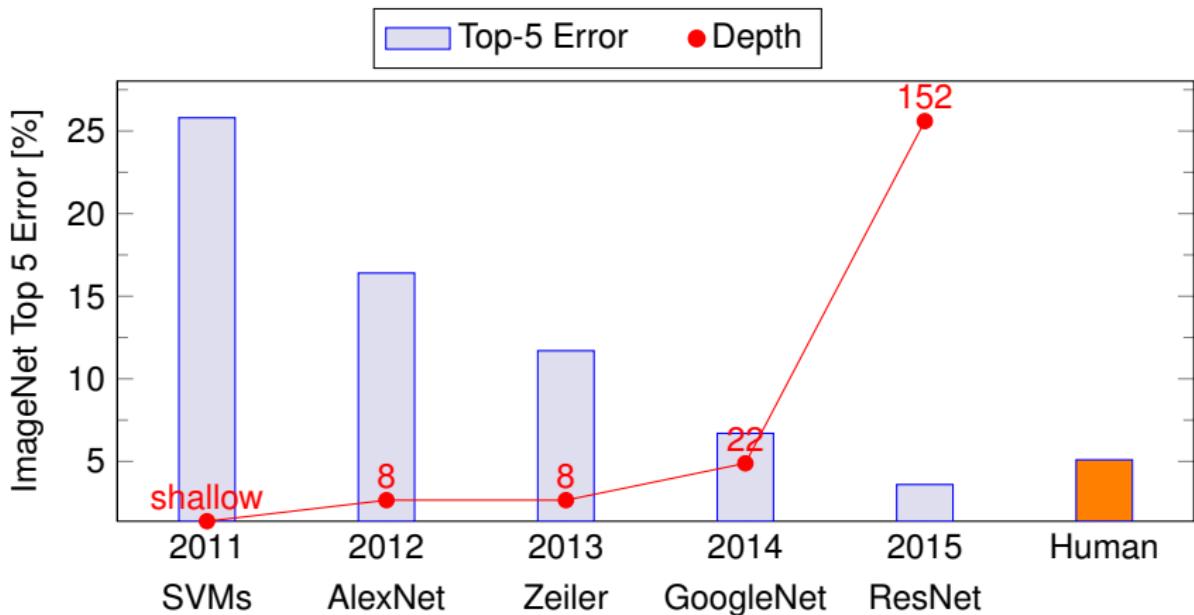
Example standard building block



Example bottleneck building block

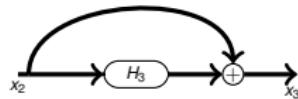
- Standard building block:  
For networks up to 34 layers or small input image sizes
- Bottleneck building block:  
For deeper networks and larger input image sizes

## Evolution of Depth



Source: [image-net.org](http://image-net.org), Russakovsky et al. 2015

## The Ensemble View [17]

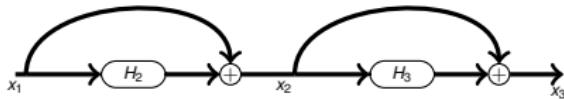


$$\mathbf{x}_{l+1} = \mathbf{x}_l + H_{l+1}(\mathbf{x}_l)$$

Consider 3 layer ResNet:

$$\mathbf{x}_3 = \mathbf{x}_2 + H_3(\mathbf{x}_2)$$

## The Ensemble View [17]

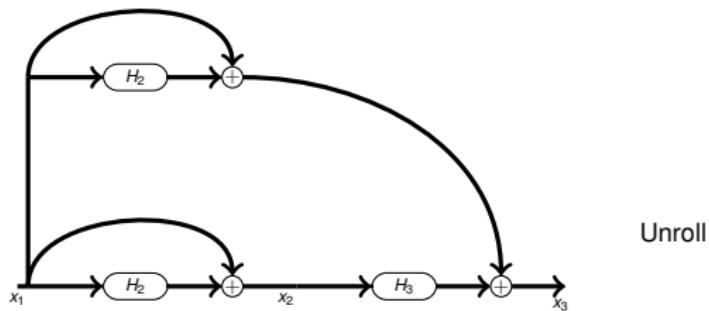


$$\mathbf{x}_{l+1} = \mathbf{x}_l + H_{l+1}(\mathbf{x}_l)$$

Consider 3 layer ResNet:

$$\begin{aligned}\mathbf{x}_3 &= \mathbf{x}_2 + H_3(\mathbf{x}_2) \\ &= [\mathbf{x}_1 + H_2(\mathbf{x}_1)] + H_3(\mathbf{x}_1 + H_2(\mathbf{x}_1))\end{aligned}$$

## The Ensemble View [17]

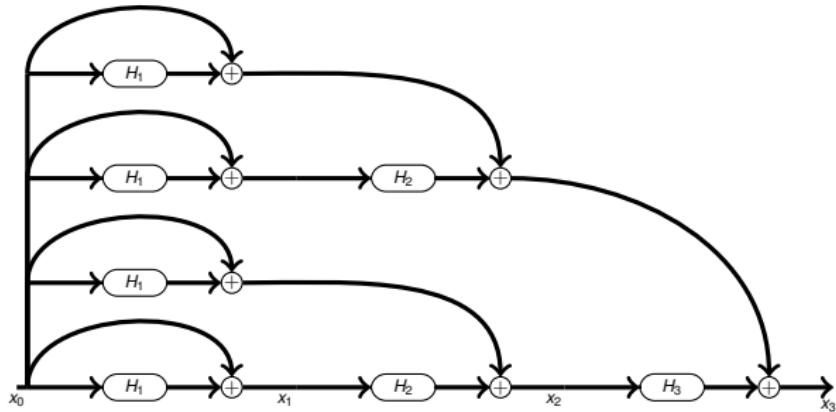


$$\mathbf{x}_{l+1} = \mathbf{x}_l + H_{l+1}(\mathbf{x}_l)$$

Consider 3 layer ResNet:

$$\begin{aligned}
 \mathbf{x}_3 &= \mathbf{x}_2 + H_3(\mathbf{x}_2) \\
 &= [\mathbf{x}_1 + H_2(\mathbf{x}_1)] + H_3(\mathbf{x}_1 + H_2(\mathbf{x}_1))
 \end{aligned}$$

## The Ensemble View [17]

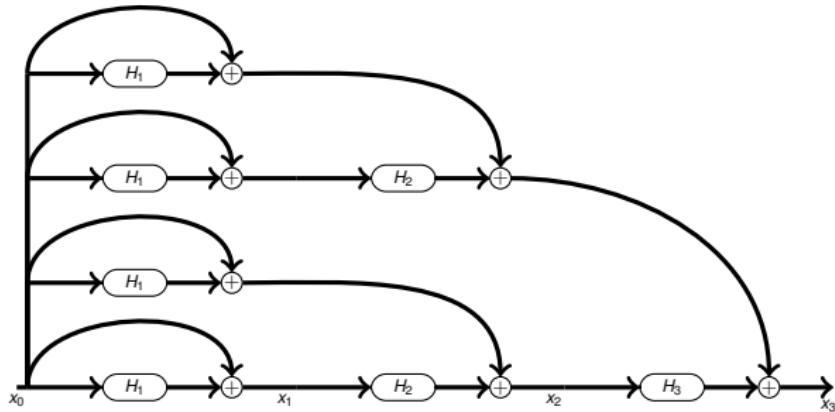


$$\mathbf{x}_{l+1} = \mathbf{x}_l + H_{l+1}(\mathbf{x}_l)$$

Consider 3 layer ResNet:

$$\begin{aligned}
 \mathbf{x}_3 &= \mathbf{x}_2 + H_3(\mathbf{x}_2) \\
 &= [\mathbf{x}_1 + H_2(\mathbf{x}_1)] + H_3(\mathbf{x}_1 + H_2(\mathbf{x}_1)) \\
 &= [\mathbf{x}_0 + H_1(\mathbf{x}_0) + H_2(\mathbf{x}_0 + H_1(\mathbf{x}_0))] + H_3(\mathbf{x}_0 + H_1(\mathbf{x}_0) + H_2(\mathbf{x}_0 + H_1(\mathbf{x}_0)))
 \end{aligned}$$

## The Ensemble View [17]



ResNets behave like ensemble of shallow networks  
 → Implicitly average exponentially many networks

## Classical Feed-forward Network vs. Residual Networks

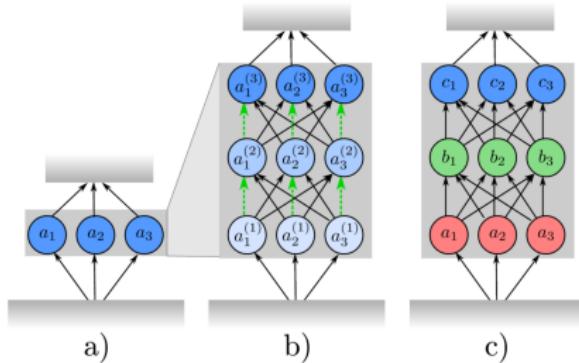
Classical feed-forward network:

- At layer level: one single path
- At neuron level: many different paths of **same** length  
(= exponential in #layers)

Residual networks:

- At layer level:  $2^n$  paths
- At neuron layer: many different paths of **varying** length going through different subsets of layers.

# The Representation View [1]

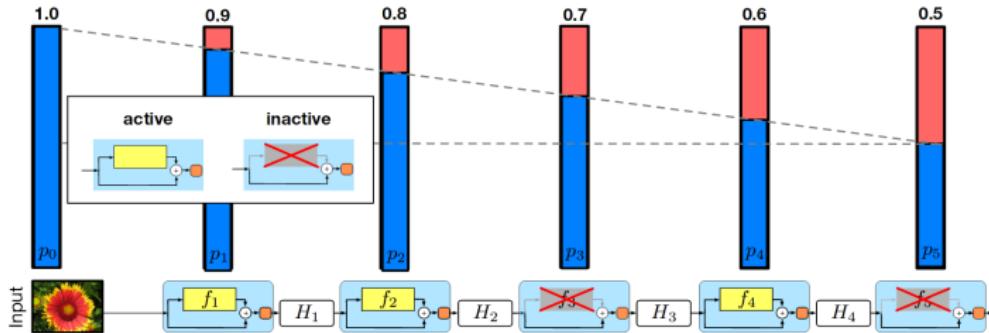


- (a) Single layer: direct computation
- (b) Residual Network: unrolled iterative estimation
- (c) Classic network: produces new representation at each layer

- Residual blocks do not compute entirely new representations
- They instead iteratively refine their input representations
- Residual networks allow removal of connections without significant drop in performance
- This can even be exploited: stochastic depth

Source: [1]

## Deep Networks with Stochastic Depth [5]



- Stochastic depth: layer-wise dropout, i.e., drop random layers and bypass with identity
- Ensemble of exponentially many small networks
- Networks are short during training → decreased training time
- 1200 layers trainable (CIFAR-10 Error: 4.91%)

Source: [5]

**NEXT TIME  
ON DEEP LEARNING**

## The rise of residual connections



**FAU**

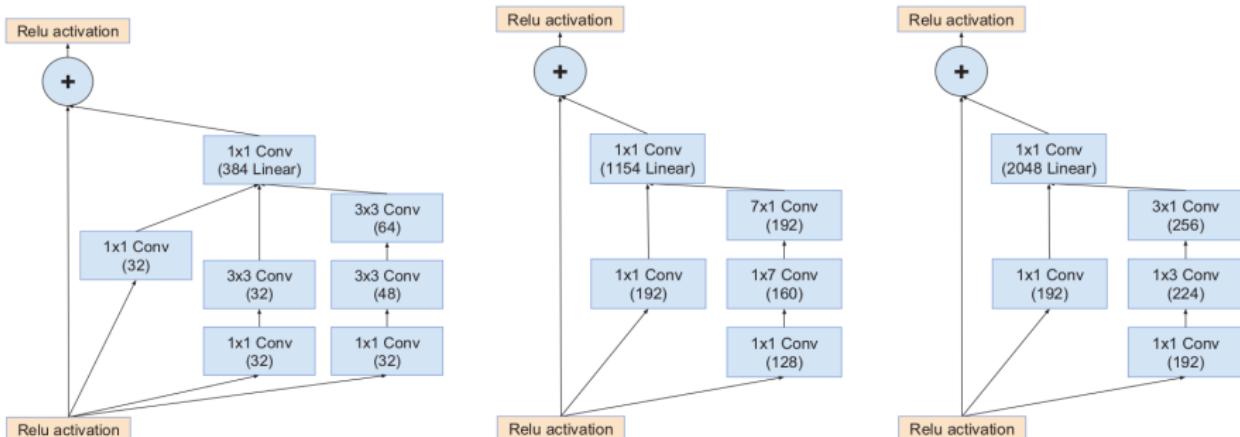
FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Architectures - Part 4

**A. Maier, V. Christlein, K. Breininger, S. Vesal, F. Meister, C. Liu, S. Gündel, S. Jaganathan, N. Maul,  
M. Vornehm, L. Reeb, F. Thamm, M. Hoffmann, C. Bergler, F. Denzinger, W. Fu, B. Geissler, Z. Yang**  
Pattern Recognition Lab, Friedrich-Alexander-Universität Erlangen-Nürnberg  
May 19, 2020



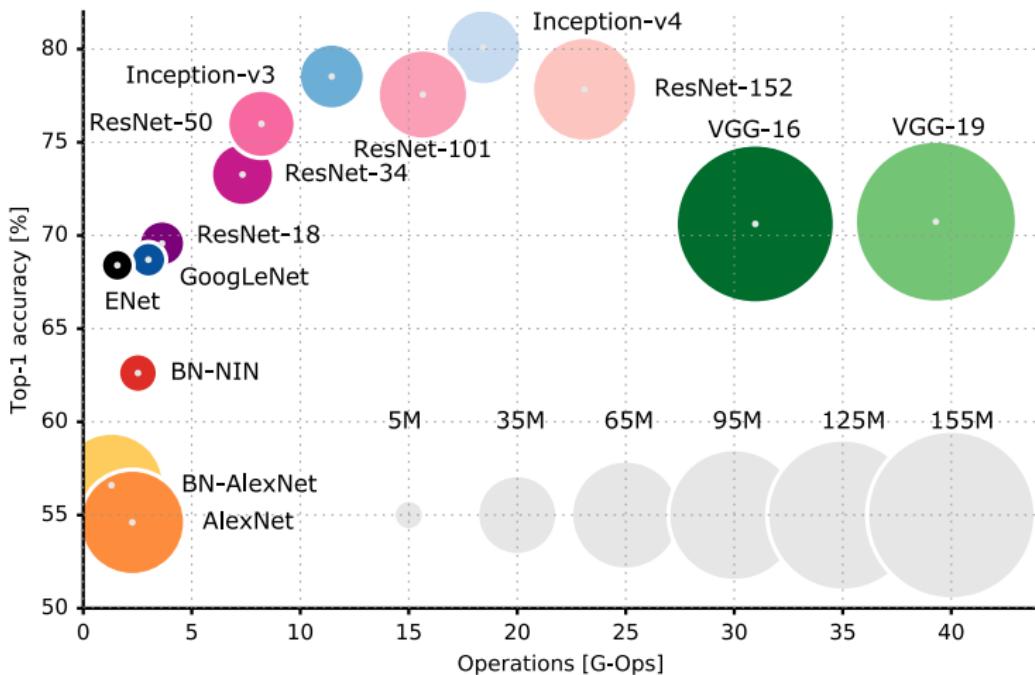
# Inception-ResNet [16]



- Combination of inception architecture and residual connections
- Faster convergence and better performance than without residual connections

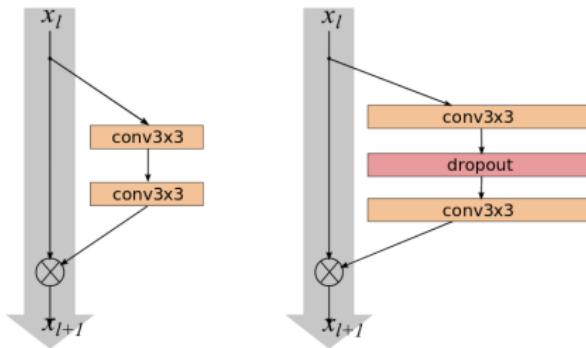
Source: [16]

## Top1 vs. Operations



Source: <https://towardsdatascience.com/neural-network-architectures-156e5bad51ba> (visited 2017/12/01), s. also Canziani et al., 2016

## Wide Residual Networks [20]

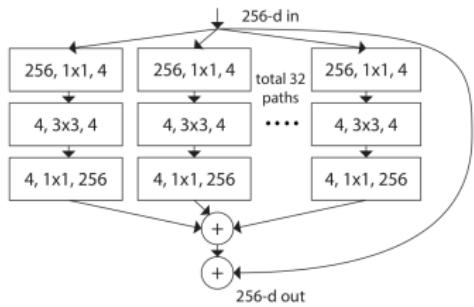


### Key features

- Decrease depth, increase width of ResNet blocks
- Use dropout in residual block
- 16 layer deep network w. similar #params outperforms 1000 layer deep network
- Power not from depth but from residual connections

Source: [20]

## ResNeXt [18]

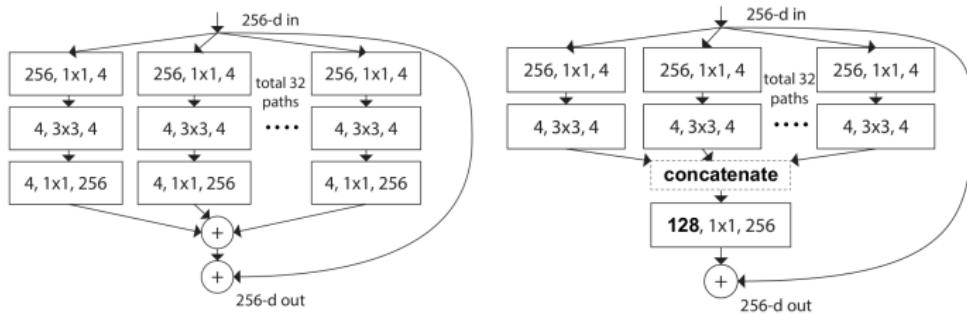


### Key features

- Aggregated residual transformations (inception layer w. same trafo)

Source: [18]

# ResNeXt [18]

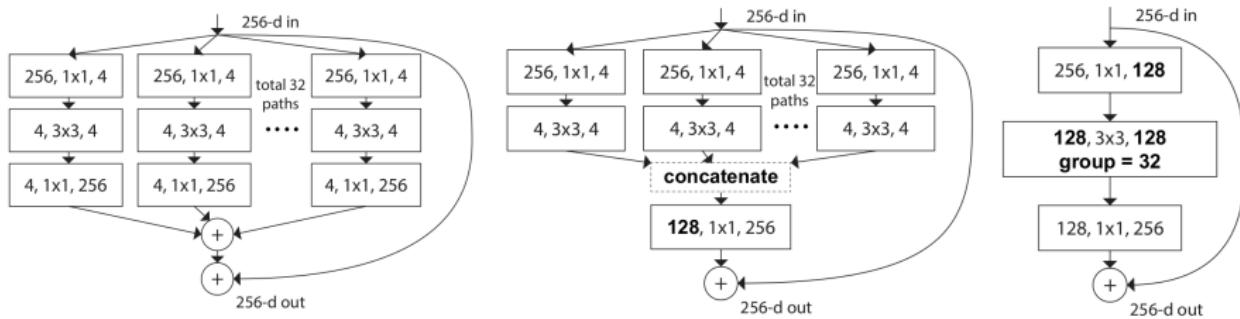


## Key features

- Aggregated residual transformations (inception layer w. same trafo)
- Equivalent: early concatenation

Source: [18]

# ResNeXt [18]

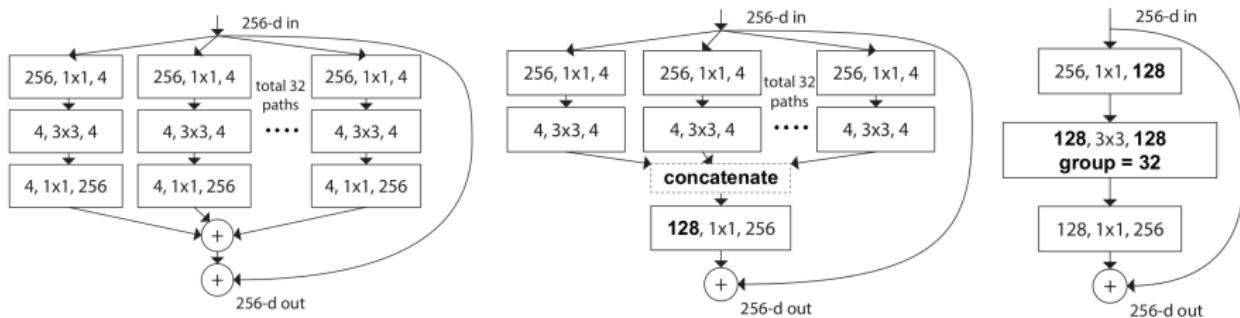


## Key features

- Aggregated residual transformations (inception layer w. same trafo)
- Equivalent: early concatenation
- Equivalent: grouped convolution (input/output chans are divided into groups, convolutions separately performed within each group)

Source: [18]

## ResNeXt [18]

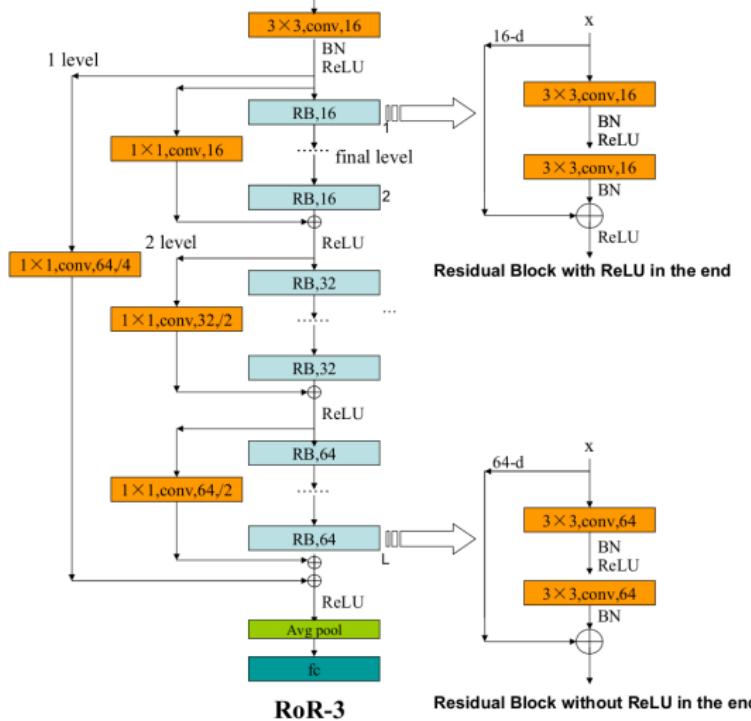


### Key features

- Aggregated residual transformations (inception layer w. same trafo)
  - Equivalent: early concatenation
  - Equivalent: grouped convolution (input/output chans are divided into groups, convolutions separately performed within each group)
  - Similar FLOPS and #params than ResNet bottleneck block
- But:** wider, sparsely connected module!

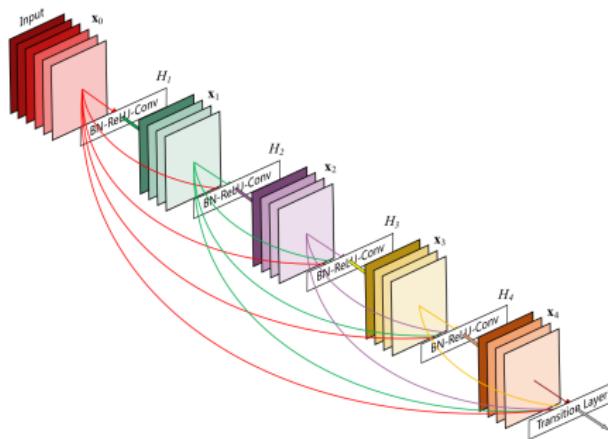
Source: [18]

# ResNet-of-ResNets [21]



Source: [21]

# DenseNets: Densely Connected Convolutional Networks [6]



- Layer input: feature-maps of all preceding layers
- Feature propagation, feature reuse
- Alleviates the vanishing-gradient problem
- Up to 264 Layers – needs actually  $\approx 1/3$  less params for same performance than ResNet due to transition layers using  $1 \times 1$  convolutions

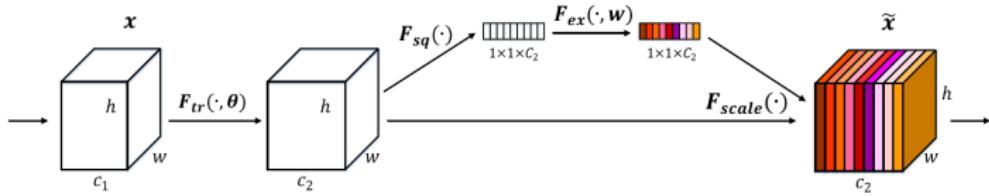
Source: [6]

## Squeeze-and-Excitation Networks (SENet) [4]

- ImageNet Challenge winner (classification) 2017: 2.3 % top-5 error
- **Motivation:** Explicitly model channel interdependencies : channels have different relevance depending on content
- Example: “Dog features” not important when differentiating cars

## Squeeze-and-Excitation Networks (SENet) [4]

- ImageNet Challenge winner (classification) 2017: 2.3 % top-5 error
- **Motivation:** Explicitly model channel interdependencies : channels have different relevance depending on content
- Example: “Dog features” not important when differentiating cars
- **Idea:** Add trainable module that allows **rescaling of channels** depending on input

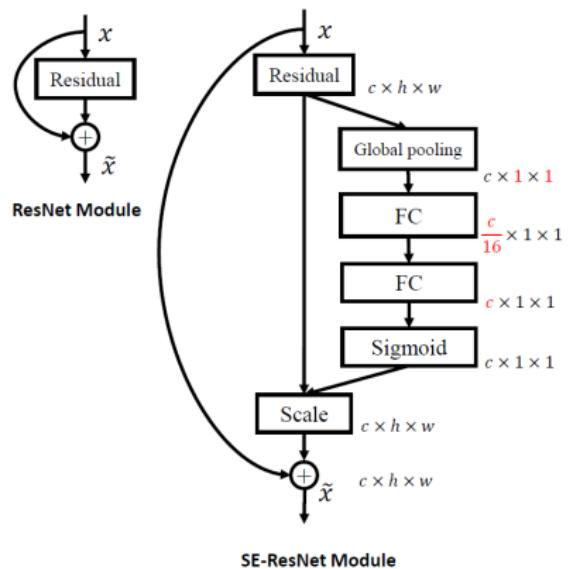


SENet module with scaled channels

Source: [4]

## Squeeze-and-Excitation Networks (SENet) [4] (cont.)

- **Squeeze:** Compress each channel into one value (global avg. pooling)  
 → Vector of size  $c$ ,  $c = \# \text{ channels}$

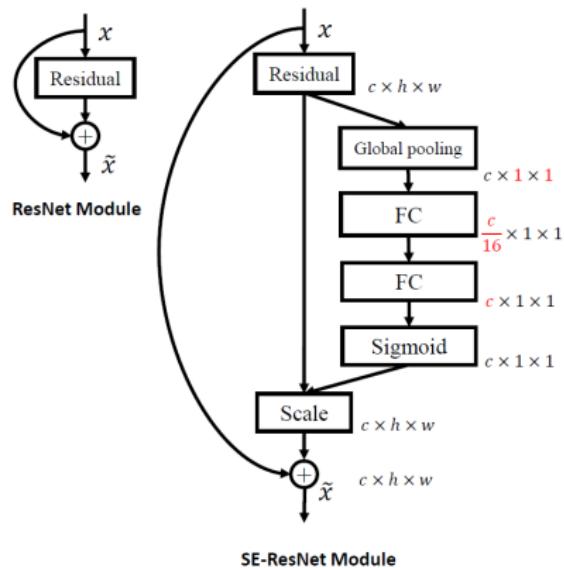


SEN extension for ResNet module

Source: [4]

## Squeeze-and-Excitation Networks (SENet) [4] (cont.)

- **Squeeze:** Compress each channel into one value (global avg. pooling)  
→ Vector of size  $c$ ,  $c = \# \text{ channels}$
- **Excitation:** FC layers & sigmoid to achieve scaling vector → compare to gating in LSTMs (next week)

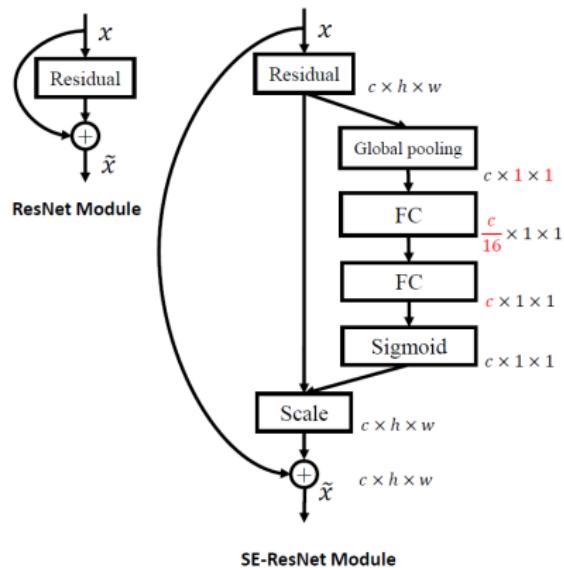


SEN extension for ResNet module

Source: [4]

## Squeeze-and-Excitation Networks (SENet) [4] (cont.)

- **Squeeze:** Compress each channel into one value (global avg. pooling)  
→ Vector of size  $c$ ,  $c = \# \text{ channels}$
- **Excitation:** FC layers & sigmoid to achieve scaling vector → compare to gating in LSTMs (next week)
- **Scale:** Scale input feature maps

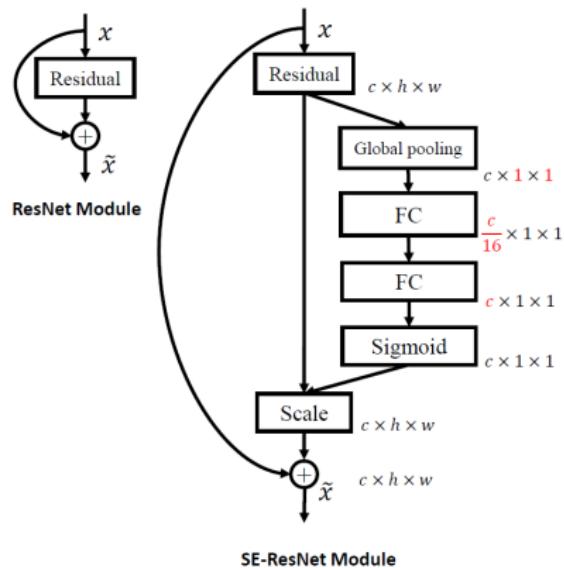


SEN extension for ResNet module

Source: [4]

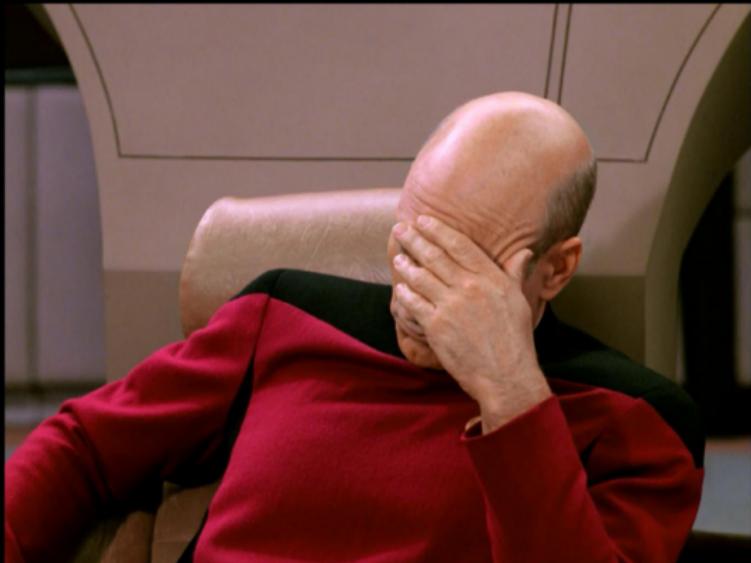
## Squeeze-and-Excitation Networks (SENet) [4] (cont.)

- **Squeeze:** Compress each channel into one value (global avg. pooling)  
→ Vector of size  $c$ ,  $c = \# \text{ channels}$
- **Excitation:** FC layers & sigmoid to achieve scaling vector → compare to gating in LSTMs (next week)
- **Scale:** Scale input feature maps
- Can be combined with most architectures, e.g., Inception, ResNet, ResNeXt, ...



SEN extension for ResNet module

Source: [4]



**NOT ANOTHER ARCHITECTURE**

**NEXT TIME  
ON DEEP LEARNING**



**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Architectures - Part 5

**A. Maier, V. Christlein, K. Breininger, S. Vesal, F. Meister, C. Liu, S. Gündel, S. Jaganathan, N. Maul,  
M. Vornehm, L. Reeb, F. Thamm, M. Hoffmann, C. Bergler, F. Denzinger, W. Fu, B. Geissler, Z. Yang**  
Pattern Recognition Lab, Friedrich-Alexander-Universität Erlangen-Nürnberg  
May 19, 2020





**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# Learning Architectures



# Learning Architectures

## Goal: Self-developing network structures

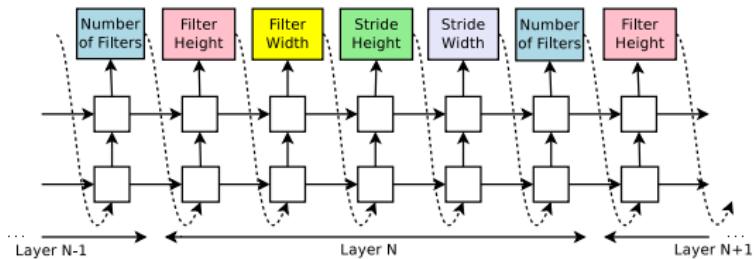
Optimized with respect to

- Accuracy
- FLOPs

Possible option: Grid-search typically too time-consuming

# Learning Architectures

## With reinforcement learning [22]



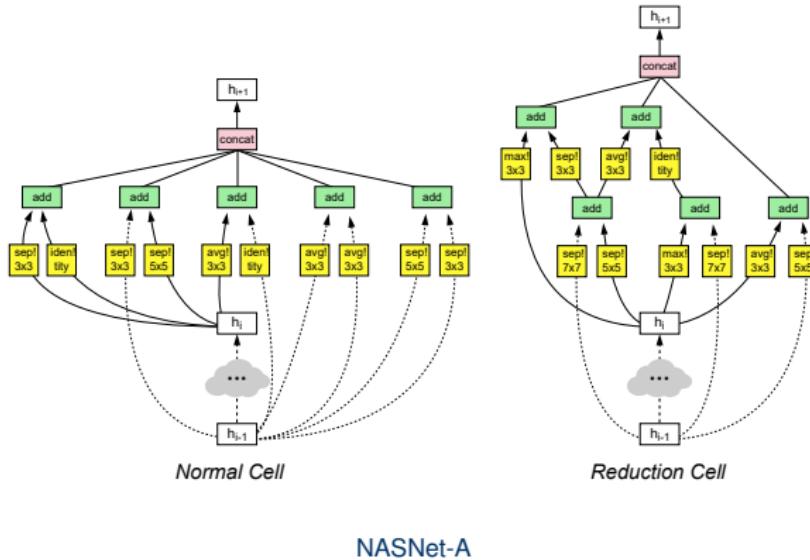
- Recurrent neural network (RNN) to generate model descriptions of networks
- Train RNN with reinforcement learning to maximize expected accuracy

## Other options

- Reinforcement learning for small building blocks transferred to large CNNs
- Genetic algorithms
- Energy-based
- ...

Source: [22]

## What do the learned architectures look like?



- Performance for ImageNet on par with SENets, with lower computational costs
- Optimization of networks of different size, e.g., for mobile platforms

Source: [22]

## ImageNet Challenge - Where are we?

- ImageNet results for classification typically <5 % in most submissions
- Substantial and significant improvements more and more difficult to show
- Last “official” challenge (with CVPR workshop) in 2017, now on Kaggle

## ImageNet Challenge - Where are we?

- ImageNet results for classification typically <5 % in most submissions
- Substantial and significant improvements more and more difficult to show
- Last “official” challenge (with CVPR workshop) in 2017, now on Kaggle
- New data sets are generated/needed, e.g., 3D scenes and human-level understanding
- **Examples:** MS COCO (<http://cocodataset.org>), Visual Genome Dataset (<https://visualgenome.org/>)
- Additional research directions: Speed and size of networks on mobile platforms

# Conclusion

## Summary

- $1 \times 1$  filters to reduce parameters and add regularization
- Inception layers
- Residual connections
- New architectures can be learned

Rise of deeper models (from 5 layers to more than 1000)

## However

- Often a smaller net is sufficient
- Dependent on amount of training data
- Deep vs. wide layers

**NEXT TIME  
ON DEEP LEARNING**

## Coming Up

- Recurrent neural networks
- (Truncated) Backpropagation through time
- Long short-term memory
- Gated recurrent unit

## Comprehensive Questions

- What are the advantages of deeper models in comparison to shallow networks?
- Why can we say that residual networks learn an ensemble of shallow networks?
- How does a bottleneck layer work?
- What is the standard inception module and how can it be improved?

## Further Reading

- Current state of the art networks:
  - Dual Path Networks  
<http://papers.nips.cc/paper/7033-dual-path-networks>,
  - Squeeze-and-Excitation Networks <https://arxiv.org/abs/1709.01507>
- Some interesting state-of-the-art works can be found here:  
<https://medium.com/@karpathy/iclr-2017-vs-arxiv-sanity-d1488ac5c131>  
(visited: 03-12-2017)
- MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications <https://arxiv.org/abs/1704.04861>
- Deep networks without residual connections:
  - <https://arxiv.org/abs/1706.00388>,
  - <https://arxiv.org/abs/1703.01827>



**FAU**

FRIEDRICH-ALEXANDER-  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG  
SCHOOL OF ENGINEERING

# References



## References I

- [1] Klaus Greff, Rupesh K. Srivastava, and Jürgen Schmidhuber. "Highway and Residual Networks learn Unrolled Iterative Estimation". In: International Conference on Learning Representations (ICLR). Toulon, Apr. 2017. arXiv: 1612.07771.
- [2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. "Deep Residual Learning for Image Recognition". In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, June 2016, pp. 770–778. arXiv: 1512.03385.
- [3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. "Identity mappings in deep residual networks". In: Computer Vision – ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, Sept. 2016, pp. 630–645. arXiv: 1603.05027.

## References II

- [4] J. Hu, L. Shen, and G. Sun. "Squeeze-and-Excitation Networks". In: [ArXiv e-prints](#) (Sept. 2017). arXiv: 1709.01507 [cs.CV].
- [5] Gao Huang, Yu Sun, Zhuang Liu, et al. "Deep Networks with Stochastic Depth". In: [Computer Vision – ECCV 2016, Proceedings, Part IV](#). Cham: Springer International Publishing, 2016, pp. 646–661.
- [6] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: [2017 IEEE Conference on Computer Vision and Pattern Recognition \(CVPR\)](#). Honolulu, July 2017. arXiv: 1608.06993.
- [7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: [Advances In Neural Information Processing Systems 25](#). Curran Associates, Inc., 2012, pp. 1097–1105. arXiv: 1102.0183.

## References III

- [8] Yann A LeCun, Léon Bottou, Genevieve B Orr, et al. "Efficient BackProp". In: Neural Networks: Tricks of the Trade: Second Edition. Vol. 75. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 9–48.
- [9] Y LeCun, L Bottou, Y Bengio, et al. "Gradient-based Learning Applied to Document Recognition". In: Proceedings of the IEEE 86.11 (Nov. 1998), pp. 2278–2324. arXiv: 1102.0183.
- [10] Min Lin, Qiang Chen, and Shuicheng Yan. "Network in network". In: International Conference on Learning Representations. Banff, Canada, Apr. 2014. arXiv: 1102.0183.
- [11] Olga Russakovsky, Jia Deng, Hao Su, et al. "ImageNet Large Scale Visual Recognition Challenge". In: International Journal of Computer Vision 115.3 (Dec. 2015), pp. 211–252.

## References IV

- [12] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: International Conference on Learning Representations (ICLR). San Diego, May 2015. arXiv: 1409.1556.
- [13] Rupesh Kumar Srivastava, Klaus Greff, Urgen Schmidhuber, et al. "Training Very Deep Networks". In: Advances in Neural Information Processing Systems 28. Curran Associates, Inc., 2015, pp. 2377–2385. arXiv: 1507.06228.
- [14] C. Szegedy, Wei Liu, Yangqing Jia, et al. "Going deeper with convolutions". In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). June 2015, pp. 1–9.

## References V

- [15] C. Szegedy, V. Vanhoucke, S. Ioffe, et al. "Rethinking the Inception Architecture for Computer Vision". In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). June 2016, pp. 2818–2826.
- [16] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning". In: Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17) Inception-v4, San Francisco, Feb. 2017. arXiv: 1602.07261.
- [17] Andreas Veit, Michael J Wilber, and Serge Belongie. "Residual Networks Behave Like Ensembles of Relatively Shallow Networks". In: Advances in Neural Information Processing Systems 29. Curran Associates, Inc., 2016, pp. 550–558.

## References VI

- [18] Di Xie, Jiang Xiong, and Shiliang Pu. "All You Need is Beyond a Good Init: Exploring Better Solution for Training Extremely Deep Convolutional Neural Networks with Orthonormality and Modulation". In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, July 2017. arXiv: 1703.01827.
- [19] Lingxi Xie and Alan Yuille. Genetic CNN. Tech. rep. 2017. arXiv: 1703.01513.
- [20] Sergey Zagoruyko and Nikos Komodakis. "Wide Residual Networks". In: Proceedings of the British Machine Vision Conference (BMVC). BMVA Press, Sept. 2016, pp. 87.1–87.12.

## References VII

- [21] K Zhang, M Sun, X Han, et al. "Residual Networks of Residual Networks: Multilevel Residual Networks". In:  
IEEE Transactions on Circuits and Systems for Video Technology PP.99  
(2017), p. 1.
- [22] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, et al.  
Learning Transferable Architectures for Scalable Image Recognition.  
Tech. rep. 2017. arXiv: 1707.07012.