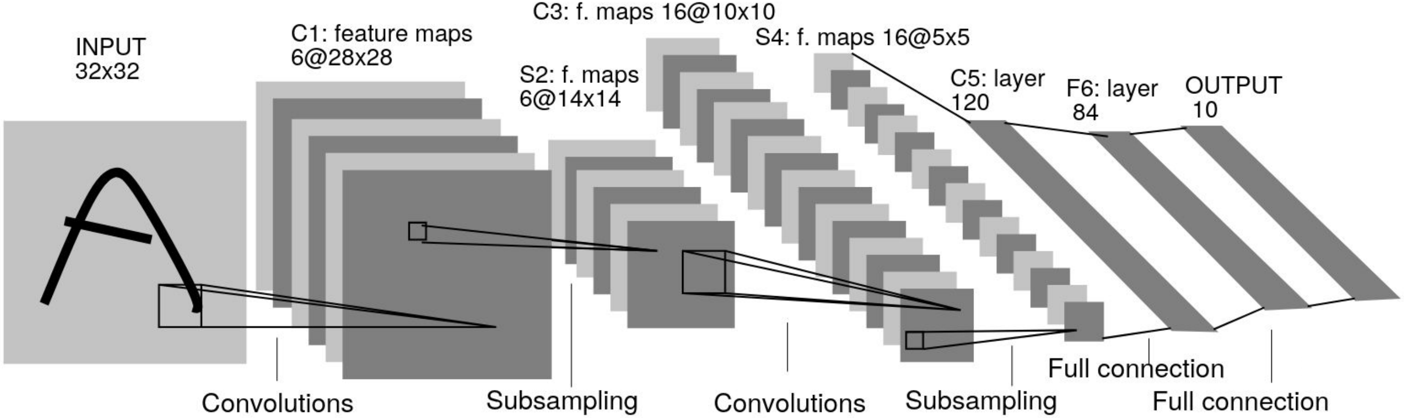
Convolutional Neural Networks



Stephan Alaniz
stephan.alaniz@telecom-paris.fr

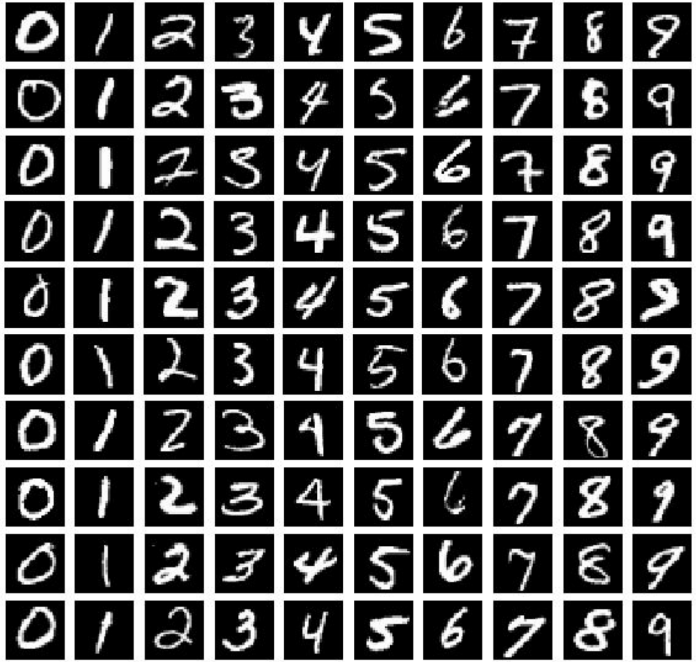
Avoiding Overfit: Regularization

- How to avoid overfit ?
  - Early stop: require a validation set : Split annotated dataset into *train, validation* and *test* samples
    - E.g: 60% train, 20% validation and 20% test
  - Introduce regularization factor λ

    $$J(\mathbf{w}) = E(\mathbf{w}) + \lambda R(\mathbf{w})$$
  - Dropout
  - Lower capacity/complexity model (fewer parameters)
  - Improve train set
    - Data augmentation

Data Augmentation - MNIST

- Different people may write same digit in different ways
- More intra-class variance

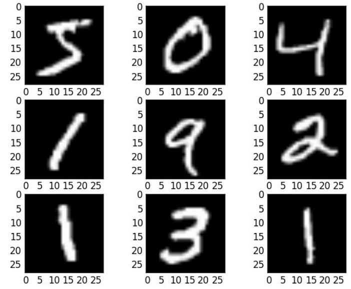


Data Augmentation - MNIST

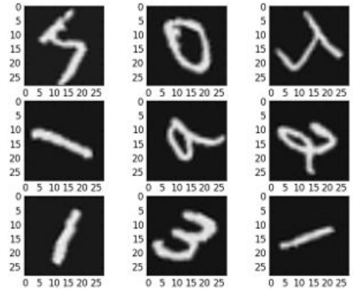
Original samples

https://www.codesofinterest.com/2018/02/using-data-augmentations-in-keras.html

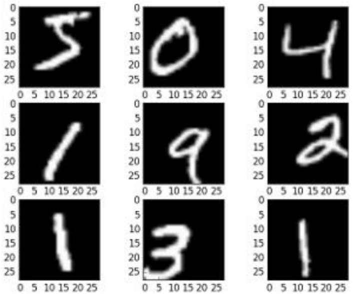
Data Augmentation - MNIST



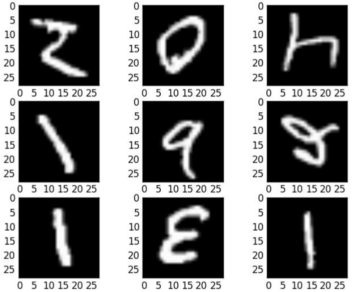
Original samples



Random rotations



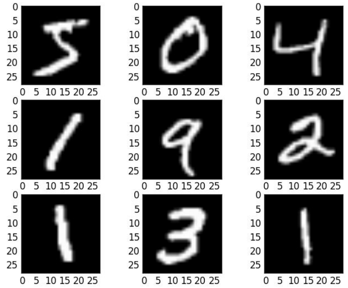
Random shift



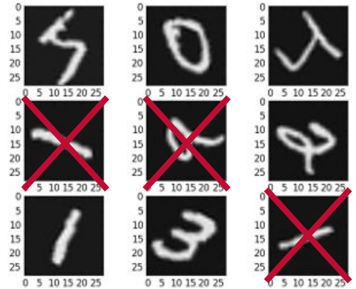
Random flips

https://www.codesofinterest.com/2018/02/using-data-augmentations-in-keras.html

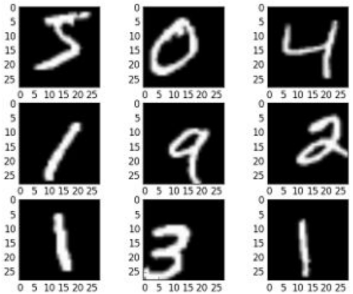
Data Augmentation - MNIST



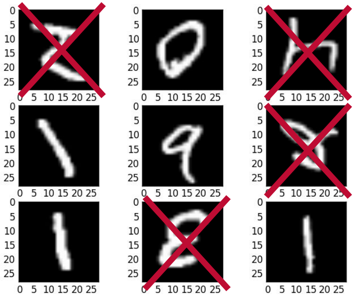
Original samples



Random rotations



Random shift



Random flips

https://www.codesofinterest.com/2018/02/using-data-augmentations-in-keras.html

Data Augmentation – LeNet300 over MNIST

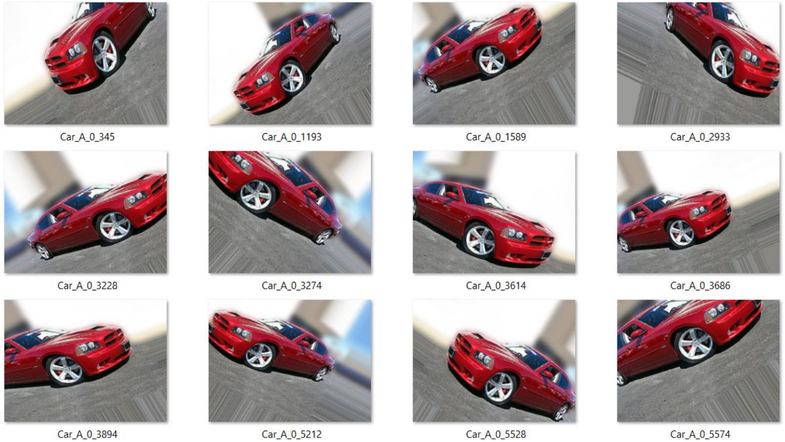
- Train set size does not change (50k images)
- Random augmentation at each epoch

Network Topology	Error [%]
1 hidden (300 U), 1 output (10 U)	4.7
1 hidden (300 U), 1 output (10 U), distorted train set	**3.6 (-1.1)**
2 hidden (300 + 100 U), 1 output (10 U)	3.05
2 hidden (300 + 100 U), distorted train set	**2.45 (-0.6)**

Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, November 1998

Data Augmentation – Another Example

Geometric transformations



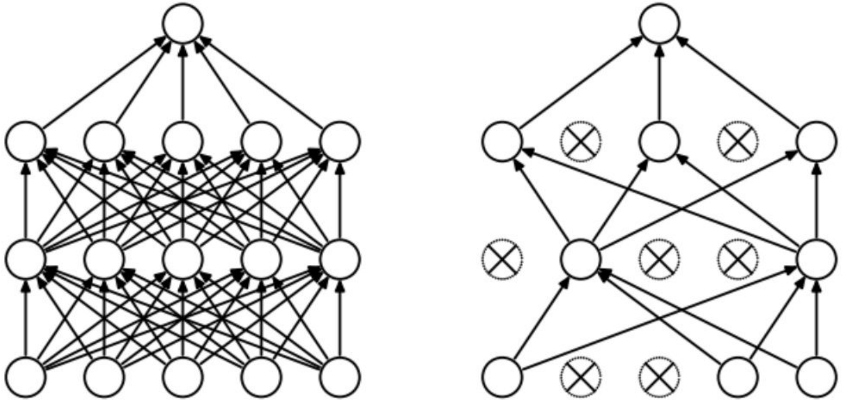
https://www.codesofinterest.com/2018/02/using-data-augmentations-in-keras.html

Color jittering

https://mxnet.apache.org/versions/1.5.0/tutorials/gluon/data_augmentation.html

Dropout

- Training: For each ***hidden*** layer, for each sample, for each iteration, ignore each neuron with p_{drop} =0.5



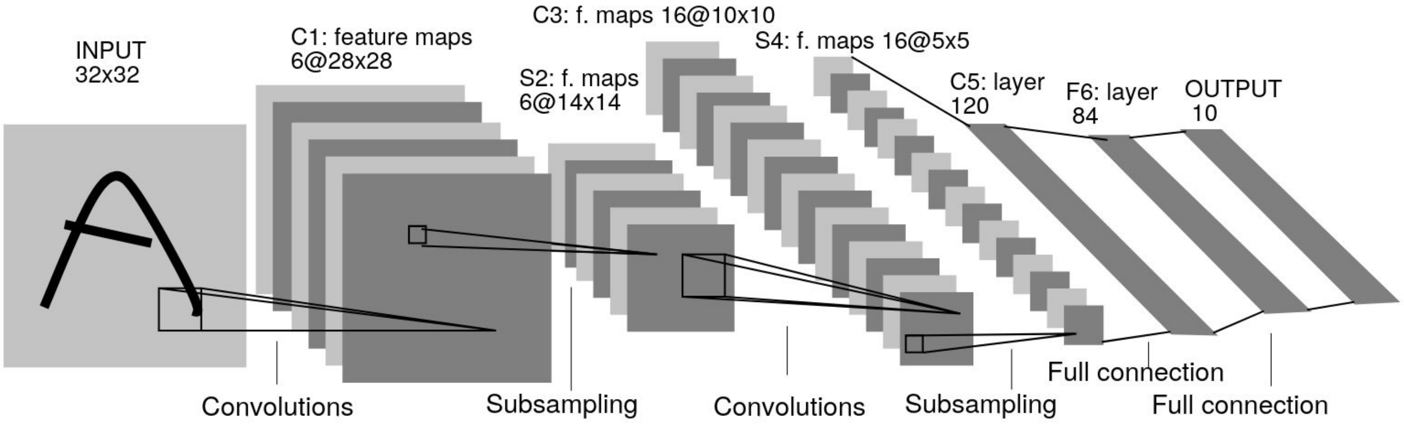
(a) Standard Neural Net (b) After applying dropout.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov,
"Dropout: a simple way to prevent neural networks from overfitting", JMLR 2014

From Shallow to Deep Architectures

Recap - LeNet5

- Repeated *convolve-and-pool* pattern
- Multiple FC layers at the end, in total ~60k parameters
- As we go deeper: Width, Height ⬇, Number of filter ⬆



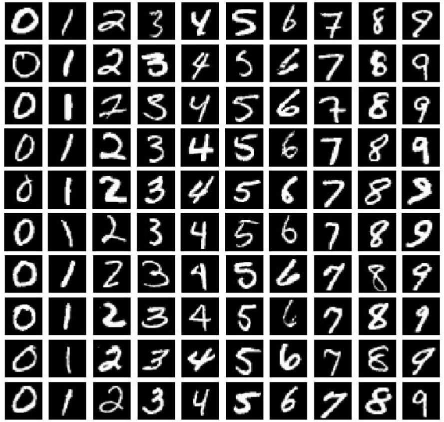
Convolutional layers
(feature extraction)

FC layers
(classification)

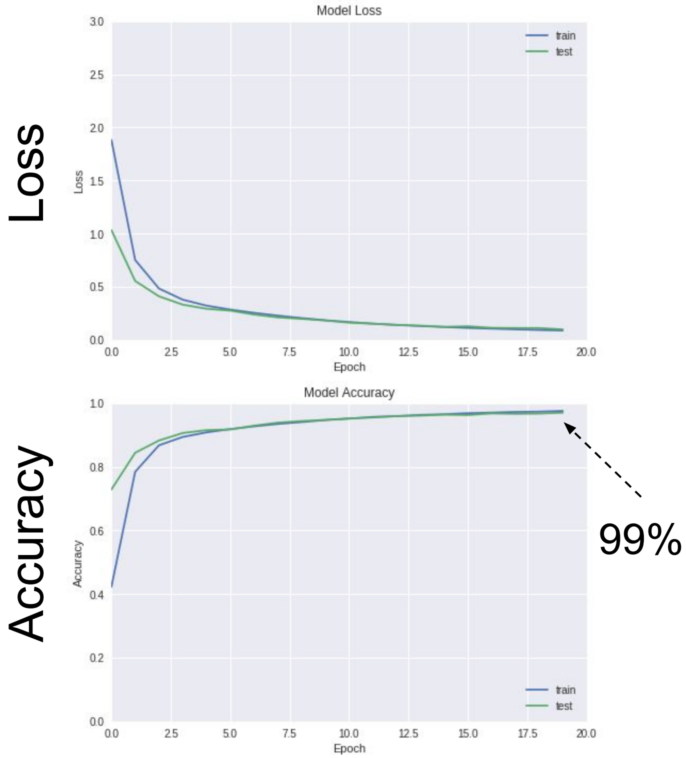
Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: Gradient-Based Learning Applied to Document Recognition,
Proceedings of the IEEE, November 1998 (PDF available online)

Recap - Convolutional Networks

Network	Architecture	Error [%]
LeNet300	1 FC output layer (10 U)	12.0
	1 hidden FC (300 U), 1 out FC (10 U)	4.7
	2 hidden FCs (300 + 100 U), 1 out FC (10 U)	3.05
LeNet5	2 Conv (3 F), 1 out FC (*LeNet1)*	1.7
	2 conv (6+16 F), 3 FC layer	0.95



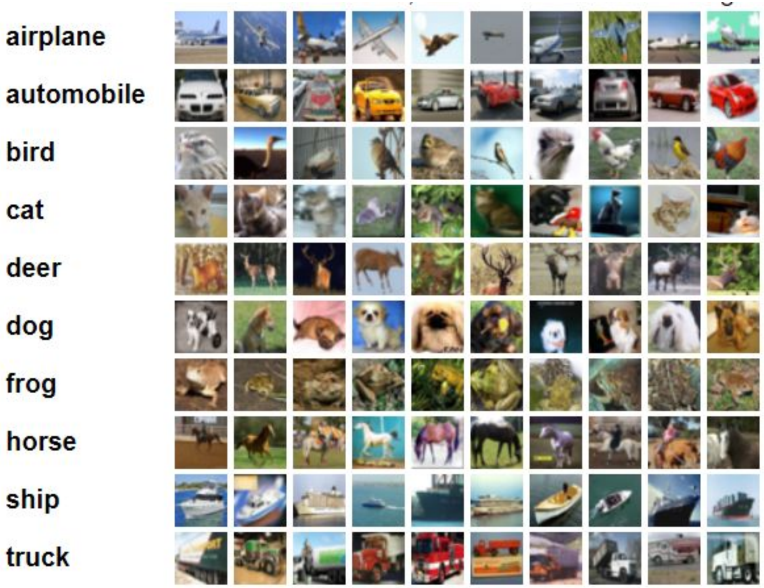
LeNet5: MNIST vs CIFAR10



MNIST

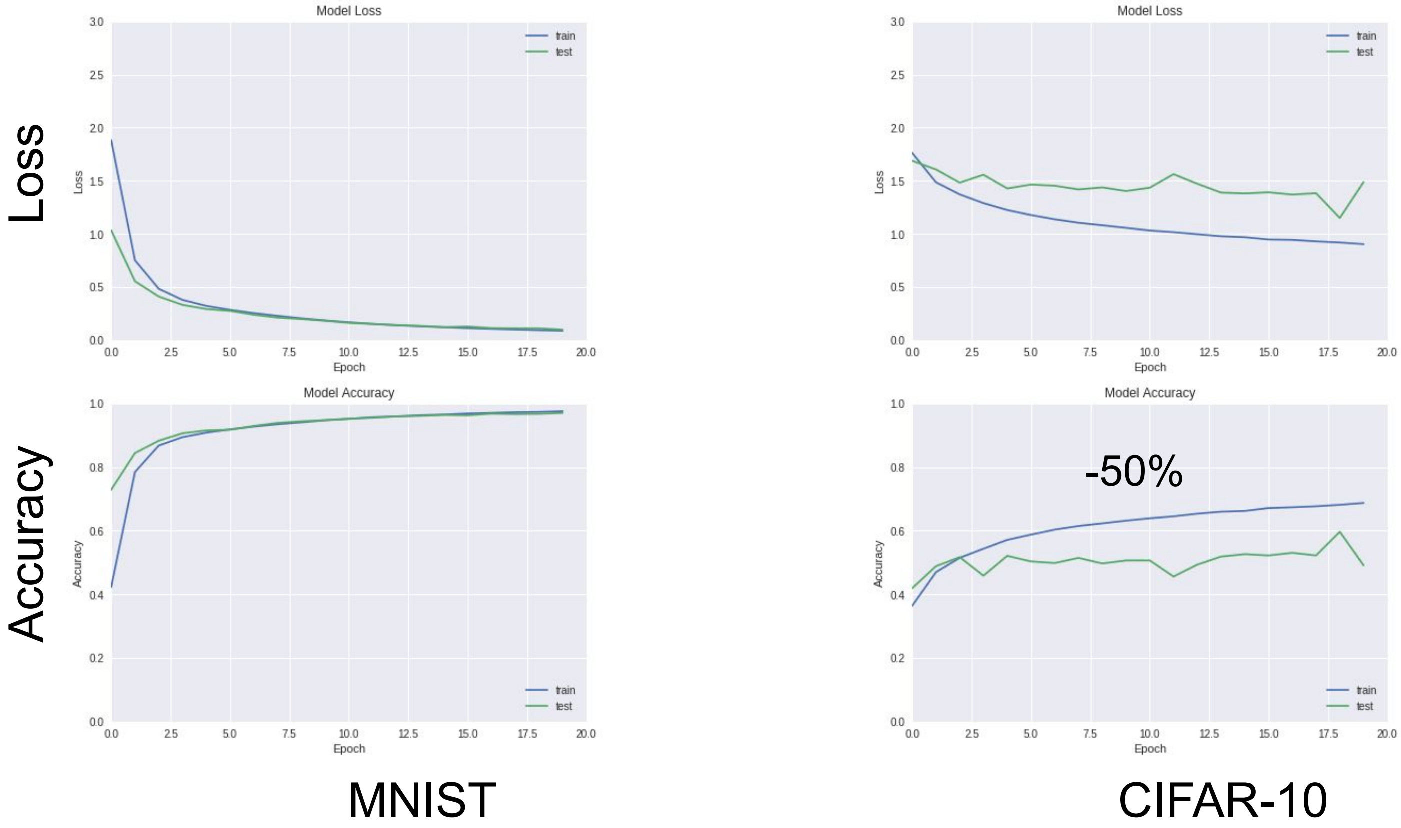
The CIFAR10 Datsets

- *CIFAR10* dataset (2009)
  - 50k train images, 10k test images, 10 classes, 32x32



https://www.cs.toronto.edu/~kriz/cifar.html

LeNet5: MNIST vs CIFAR10

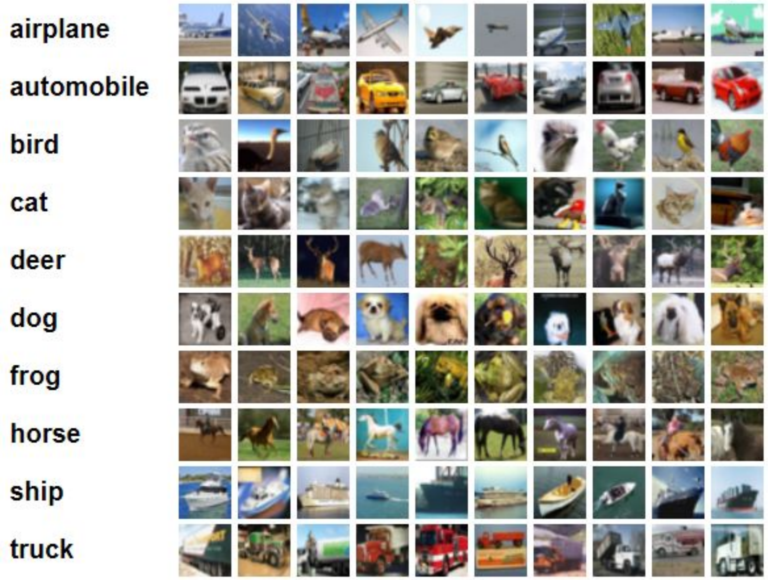


From shallow to deep networks

- More parameters (layers) -> more complex tasks
  - More annotated training samples to avoid *overfitting*

The CIFAR10 Datsets

- *CIFAR10* dataset (2009)
  - 50k train images, 10k test images, 10 classes, 32x32



https://www.cs.toronto.edu/~kriz/cifar.html

«Early» Datsets for Image Classification

- *CalTech101* dataset (2003)
  - ~10k images, 101 classes, variable size

Deng, Jia; Dong, Wei; Socher, Richard; Li, Li-Jia; Li, Kai; Fei-Fei, Li (2009), "ImageNet: A Large-Scale Hierarchical Image Database" (PDF), 2009 conference on Computer Vision and Pattern Recognition

«Early» Datsets for Image Classification

- *CalTech256* dataset (2007)
  - ~30k images, 256 classes, variable size

Fei-Fei, Li, Rob Fergus, and Pietro Perona. "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories." Computer vision and Image understanding 106, no. 1 (2007): 59-70.

ImageNet Large Scale Virsual Recognition Contest (ILSVRC)

- Originally presented in 2009 (3 M images, 5k classes)
- Since 2010 dataset for ILSVRC (1M images, 1k classes)

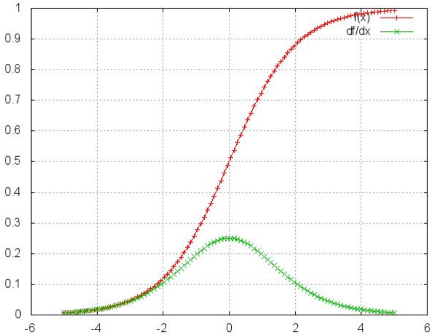
Deng, Jia; Dong, Wei; Socher, Richard; Li, Li-Jia; Li, Kai; Fei-Fei, Li (2009), "ImageNet: A Large-Scale Hierarchical Image Database" (PDF), 2009 conference on Computer Vision and Pattern Recognition

From shallow to deep networks

- More parameteres (layers) -> more complex tasks
  - More annotated training samples to avoid *overfitting*
  - Increased (training) complexity -> more compute

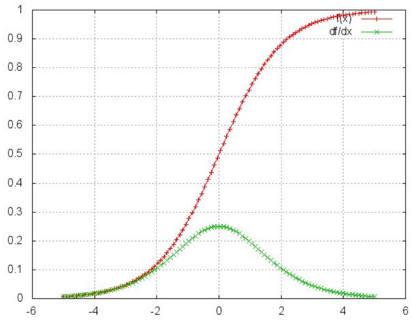
From shallow to deep networks

- More parameters (layers) -> more complex tasks
  - More annotated training samples to avoid *overfitting*
  - Increased (training) complexity
  - *Vanishing gradient* (with sigmoid activations)



Vanishing Gradient Problem

$$g'(z) = \frac{e^{-z}}{(1+e^{-z})^2}$$

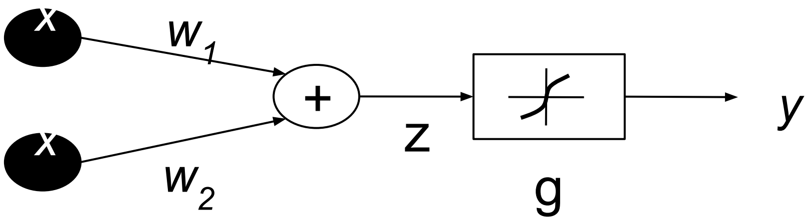


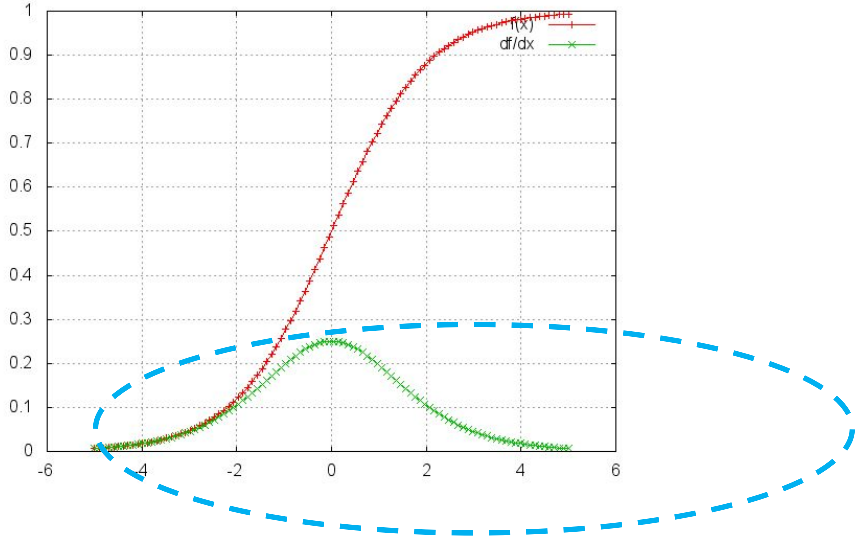
The Sigmoid Activation

- Compute gradient of E w.r.t. to each w_n via *chain rule*, e.g.:

$$\frac{\mathrm{dE}}{\mathrm{d}w_1} = \frac{\mathrm{d}E}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{dz}}\frac{\mathrm{dz}}{dw_1}$$

$$g(z) = \frac{1}{1+e^{-z}}$$



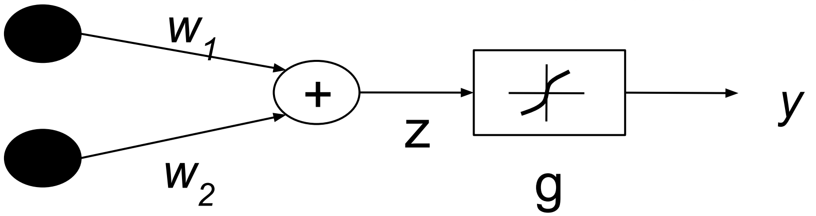


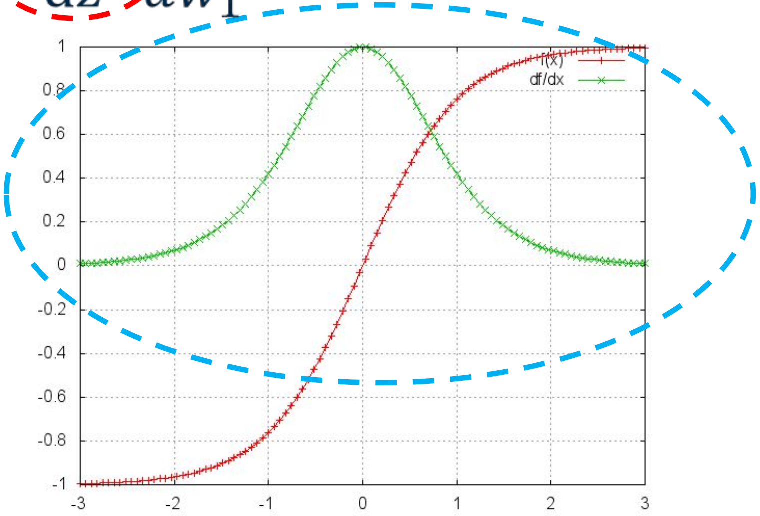
The Hyperbolic Tangent Activation

- Compute gradient of *E* w.r.t. to each w_n via *chain rule*, e.g.:

$$\frac{\mathrm{d}E}{\mathrm{d}w_1} = \frac{\mathrm{d}E}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{d}z}\frac{\mathrm{d}z}{dw_1}$$

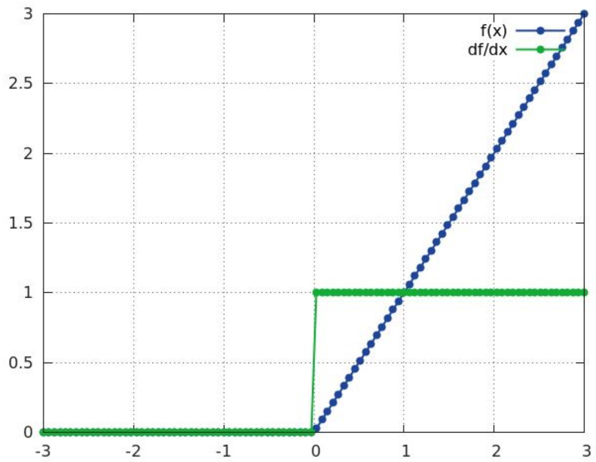
$$g(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$





The Rectified Liner Unit - ReLU

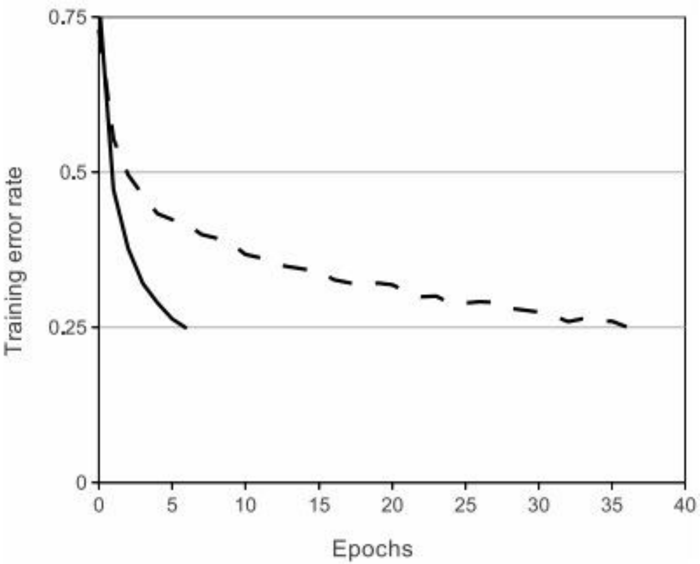
- *Rectified Linear Units* (*halfwave rectifier*) y = max (0,x)
  - Mitigates gradient vanishing problem
  - Easy to compute
  - Sparse activations
  - Biological plausibility (one-sided)



X. Glorot, A. Bordes, Y. Bengio. "Deep Sparse Rectifier Neural Networks." In Aistats, vol. 15, no. 106, p. 275. 2011.

The Rectified Liner Unit - ReLU

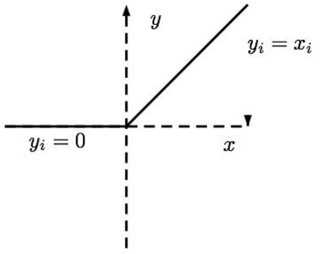
- CIFAR-10 training error a 4-layer ConvNet
- 6~7 times faster convergence



A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks."
In Advances in neural information processing systems, pp. 1097-1105. 2012.

The Leaky/Parametric ReLU

- ReLU may yield *dead neurons*
  - Gradient propagation problem
- Leaky ReLU

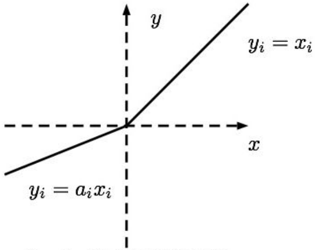


ReLU

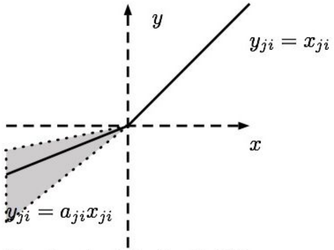
$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0.01x & \text{otherwise} \end{cases}$$

- Parametric ReLU

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ ax & \text{otherwise} \end{cases}$$



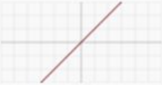
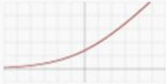
Leaky ReLU/PReLU



Randomized Leaky ReLU

X. Glorot, A. Bordes, Y. Bengio. "Deep Sparse Rectifier Neural Networks." In Aistats, vol. 15, no. 106, p. 275. 2011.

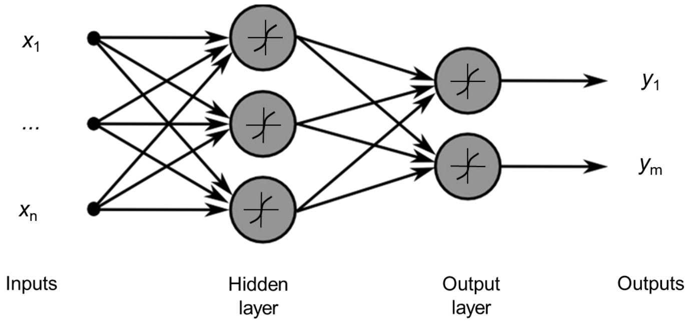
Activation Functions Summary

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

S.Sharma "*Activation Functions Explained*" https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Which Activation Function ?

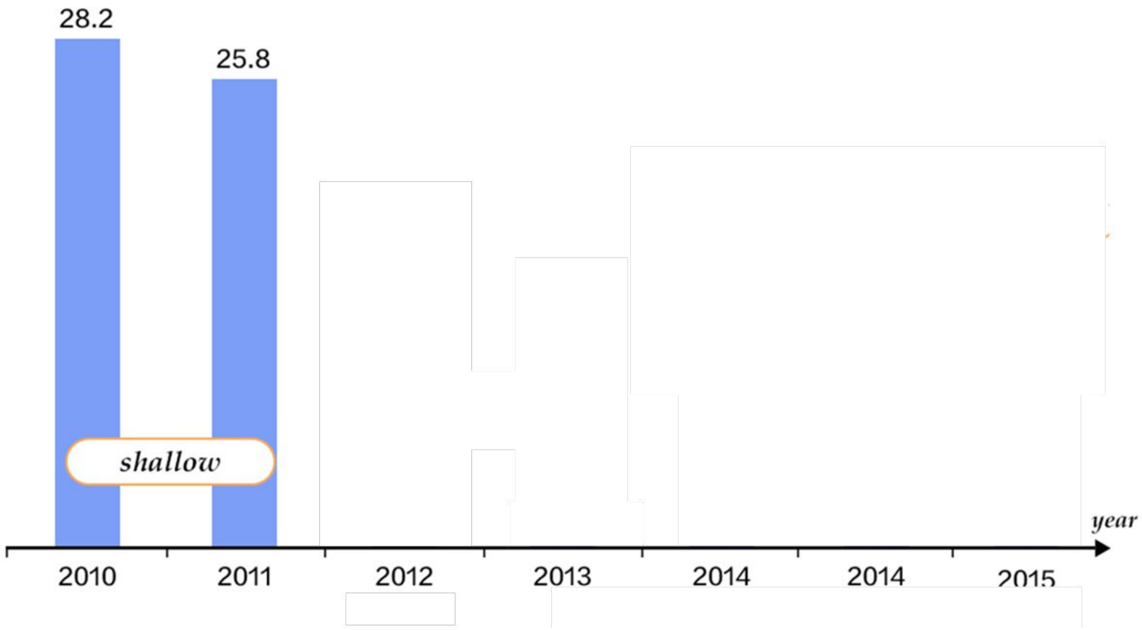
- Output layer
  - Classification
    - Sigmoid (binary) or softmax (multiclass)
  - Bounded zero-mean regression
    - Hyperbolic tangent
  - Unbounded regression
    - Linear
- Hidden layer
  - ReLU-like



CNN Architectures

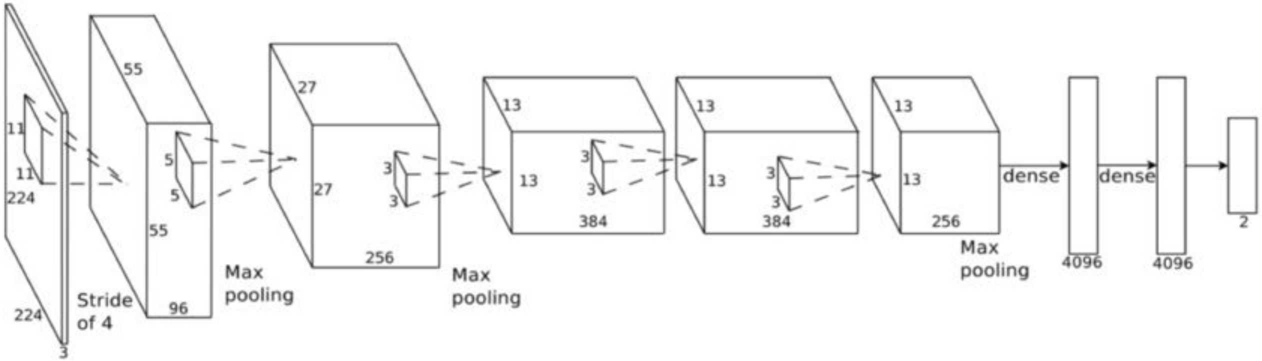
ImageNet Challenge Before the *Deep* Era

- 2010: SIFT descriptors + SVN (NEC)
- 2011: SIFT descriptors, Fisher Vectors, SVM (XRCE)



The *AlexNet* Architecture (2012)

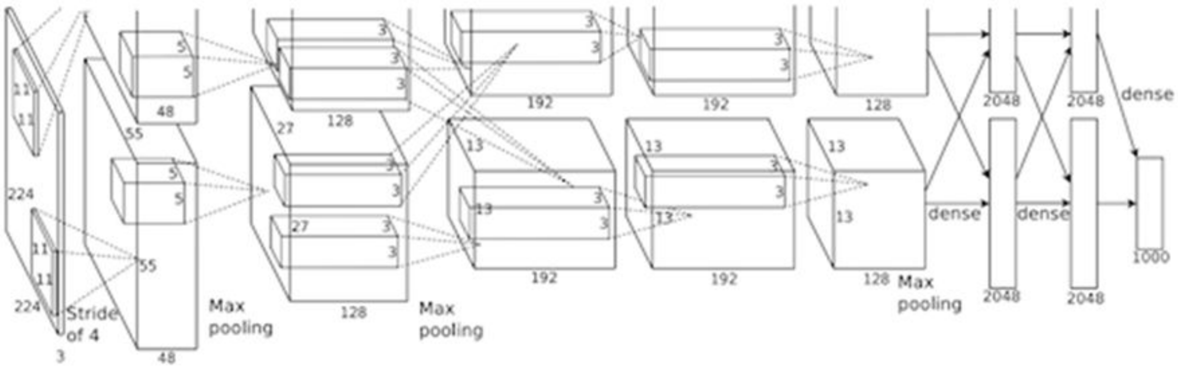
- First «*deep*» convolutional network
  - 5 convolutional layers, 3 fully connected layers
  - 62.3M parameters (conv layers 6% but take 95% of time)



A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks."
In Advances in neural information processing systems, pp. 1097-1105. 2012.

The *AlexNet* Architecture (2012)

- Trained over two *GTX580* GPUs (2GB memory each)
  - Split convolutions to different GPUs
  - Distribute the fully connected layers to different GPUs
  - Trained on 2 x GTX 580 for 5~6 days (90 epochs)



A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.

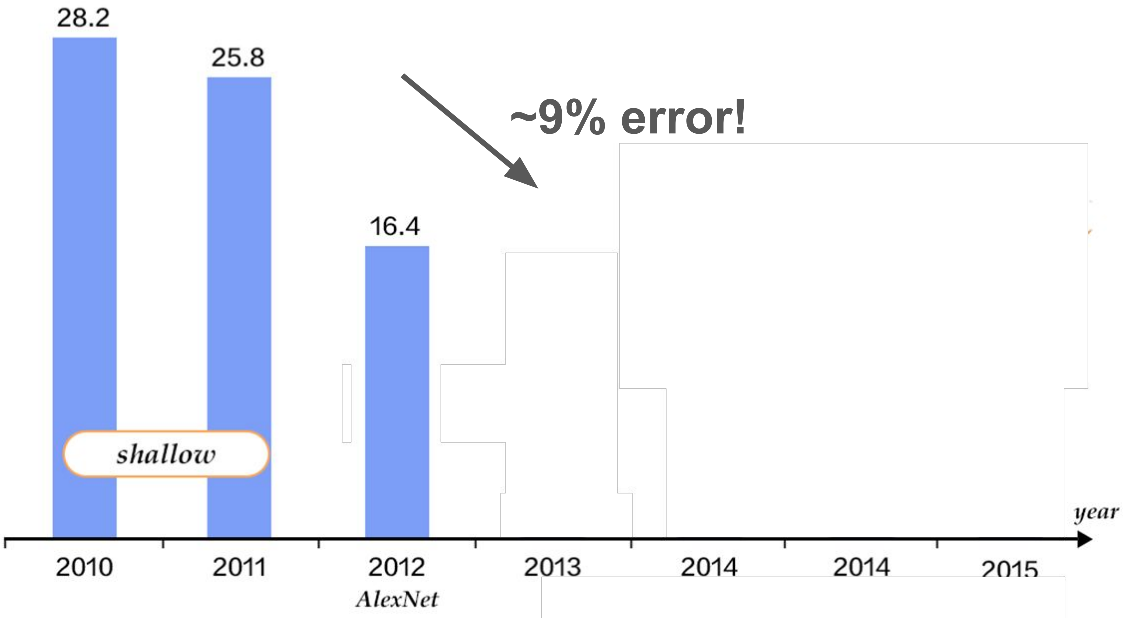
The *AlexNet* Architecture (2012)

- (Main) differences w.r.t. *LeNet5*
  - Deeper than LeNet5 (5 Conv w.r.t. 3)
  - ReLU activations in place of sigmoids
  - Dropout before FC layers (+ L2 regularization)
  - Batch size 128 images
  - Data augmentation
  - LR divide by 10 when validation error settles

A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.

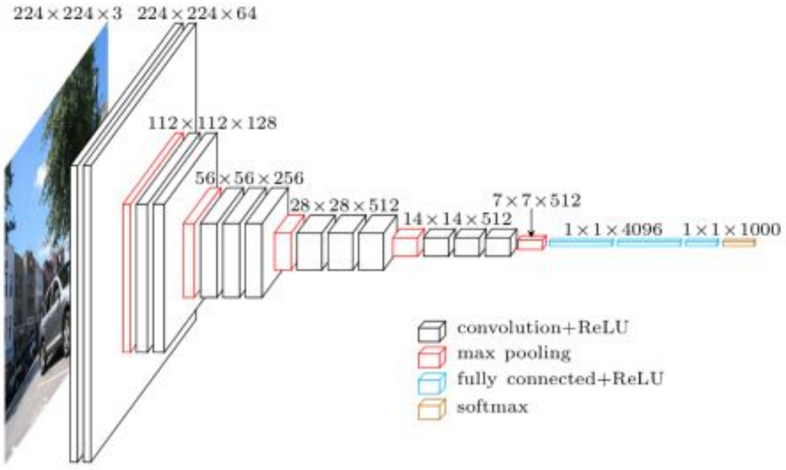
The *AlexNet* Architecture (2012)

- 2012 ILSVRC winner with top-5 error rate 16.4% (vs. 26.2%)
  - Problem: very large 11x11 filters in first conv layer



VGGNet (2014)

- Up to 19 convolutional layers, 3 fully connected layers
- Key idea: 3x3 filters everywhere



K. Simonyan, A. Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

VGGNet (2014)

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					

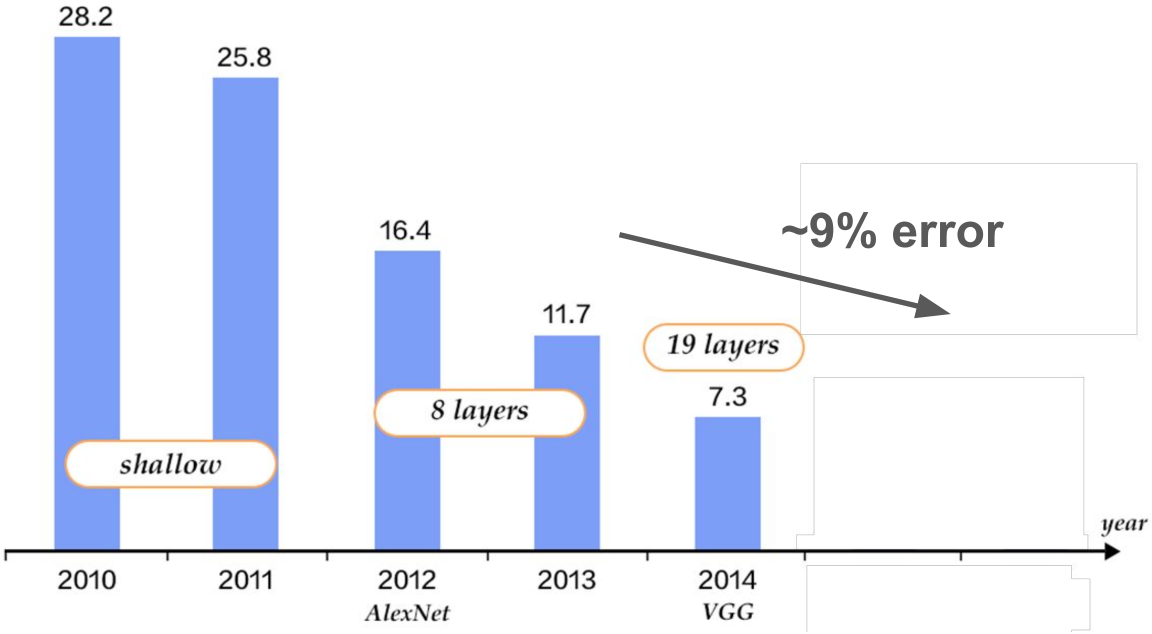
VGG-16: 138M parameters

Large, but simple architecture

K. Simonyan, A. Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

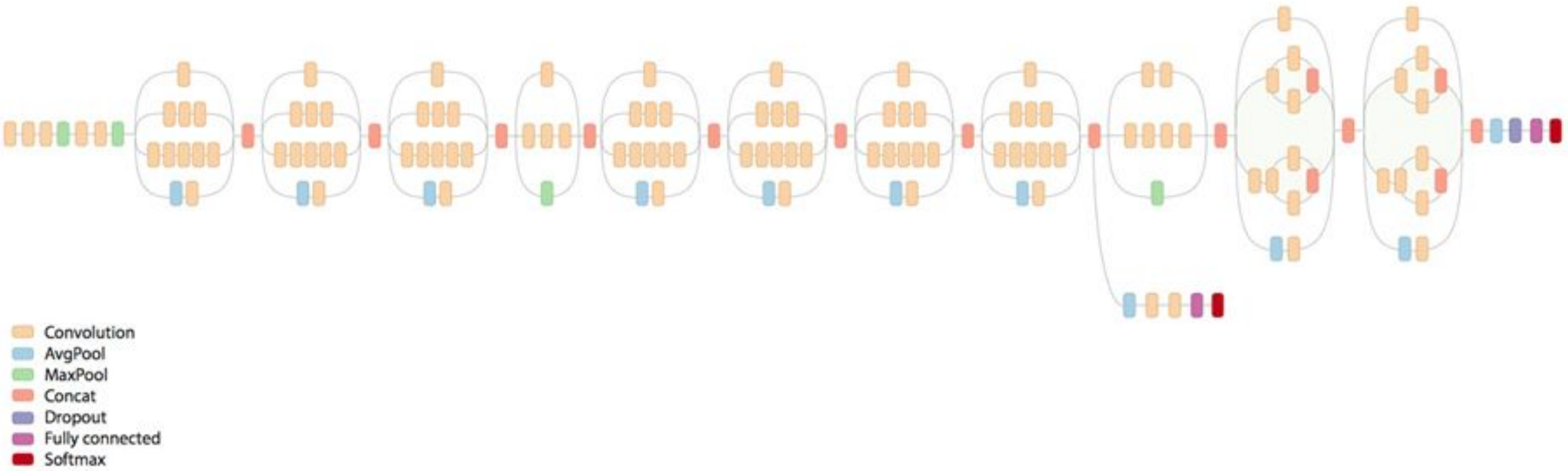
VGGNet (2014)

- 2014 ILSVRC top-5 runner with error rate 7,3%



GoogLeNet (2015)

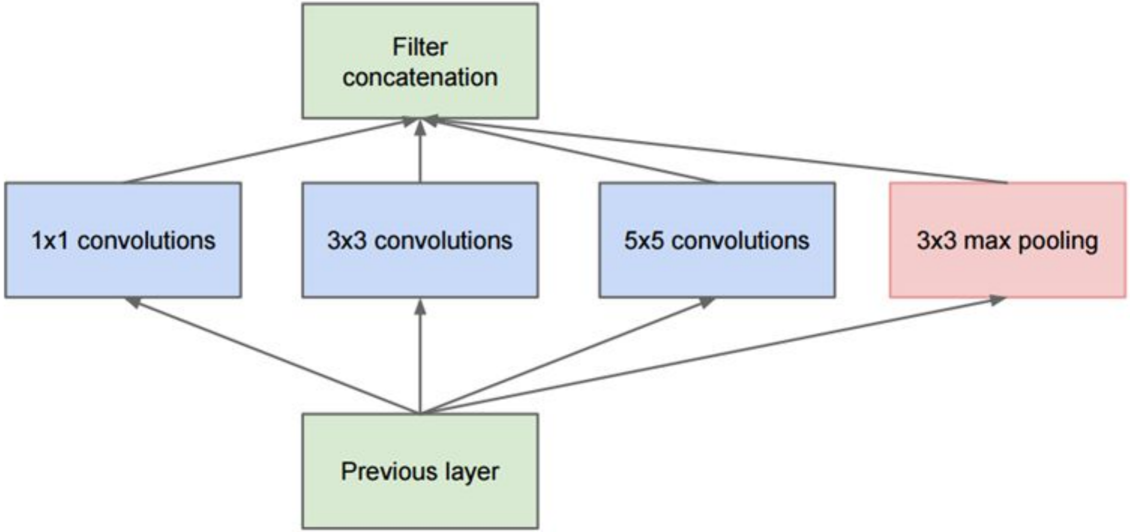
- Big IT firm (Google) wins ILSVRC
- Non-strictly sequential data processing (*Inception* module)



Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

GoogLeNet (2015)

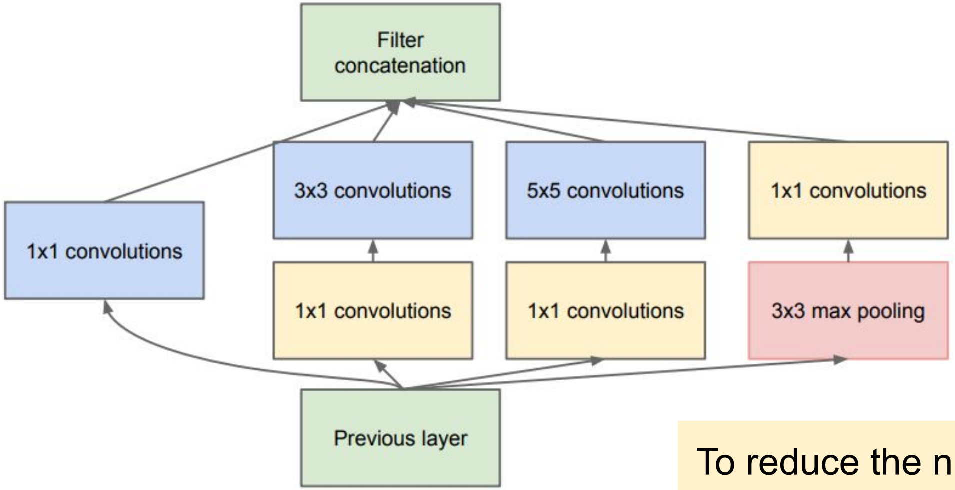
- Inception module simplified view
  - Key idea: do convolutions and pooling in parallel



Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

GoogLeNet (2015)

- Inception module simplified view
  - Key idea: do convolutions and pooling in parallel



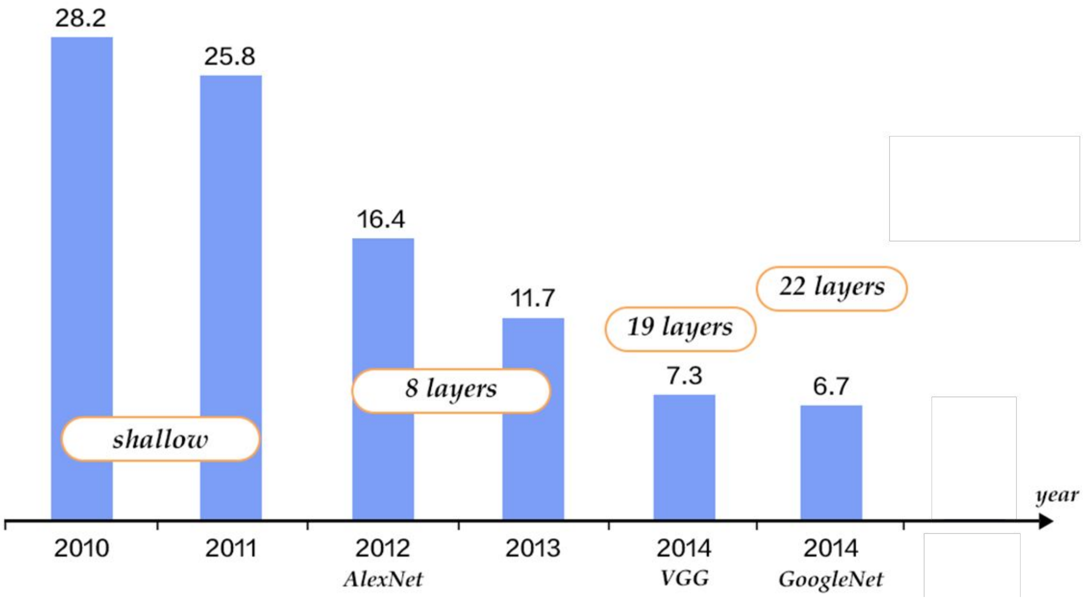
To reduce the number of input channels

(b) Inception module with dimension reductions

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

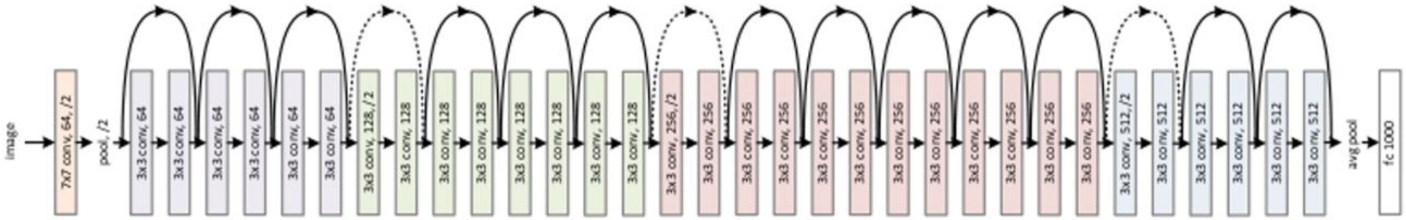
GoogLeNet (2015)

- 2014 ILSVRC winner with top-5 error rate 6.7%



ResNet (2015-present)

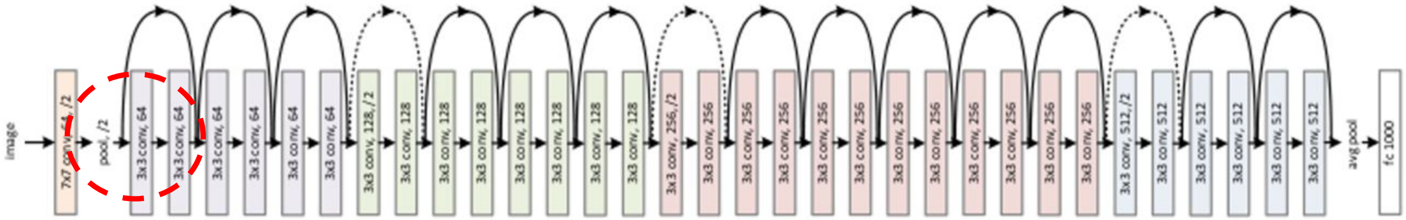
- 2015 ILSVRC winner with top-5 error rate 3.57%
  - 18, **34**, 50, 101,151 layers



He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

ResNet (2015-present)

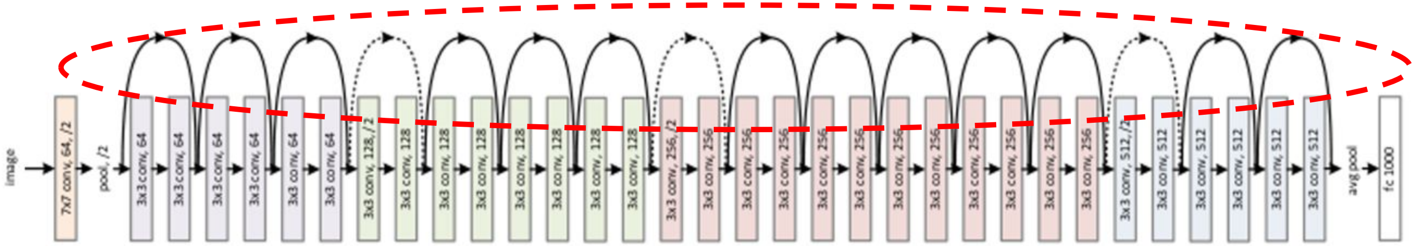
- 2015 ILSVRC winner with top-5 error rate 3.57%
  - 18, **34**, 50, 101,151 layers
  - (Almost) *pool*-less (2*px* convolution stride)



He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

ResNet (2015-present)

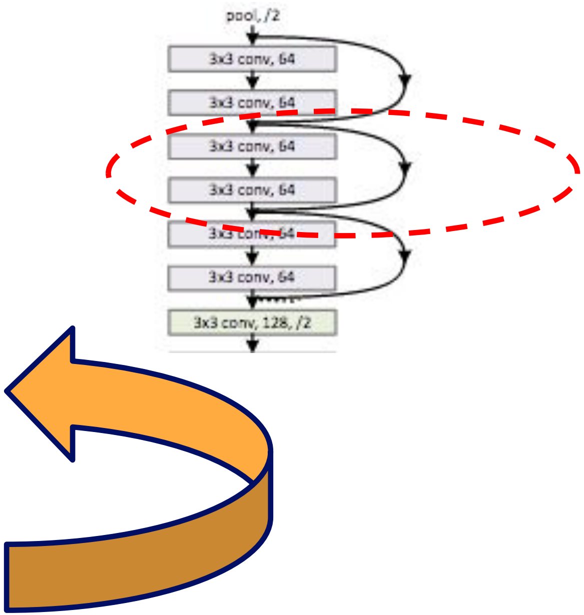
- 2015 ILSVRC winner with top-5 error rate 3.57%
  - 18, **34**, 50, 101,151 layers
  - (Almost) *pool*-less (2*px* convolution stride)
  - Relies on skip connections

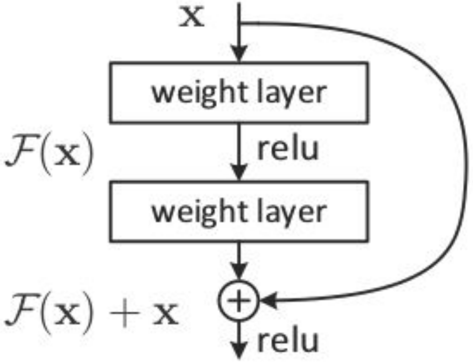


He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

ResNet (2015)

- Relies on skip/shortcut connections
  - Gradient backprop easier





Understanding and Implementing Architectures of ResNet
https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-cf51669e1624

The *ResNet* Architecture (2015)

- ResNet-152: 60M parameters
  - ReLU activations
  - Batch size 256 images
  - No dropout, but L2 regularization
  - Batch normalization
  - SDG with momentum
  - Learning rate 0.1, divided by 10 at validation plateau

A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.

ResNet (2015)

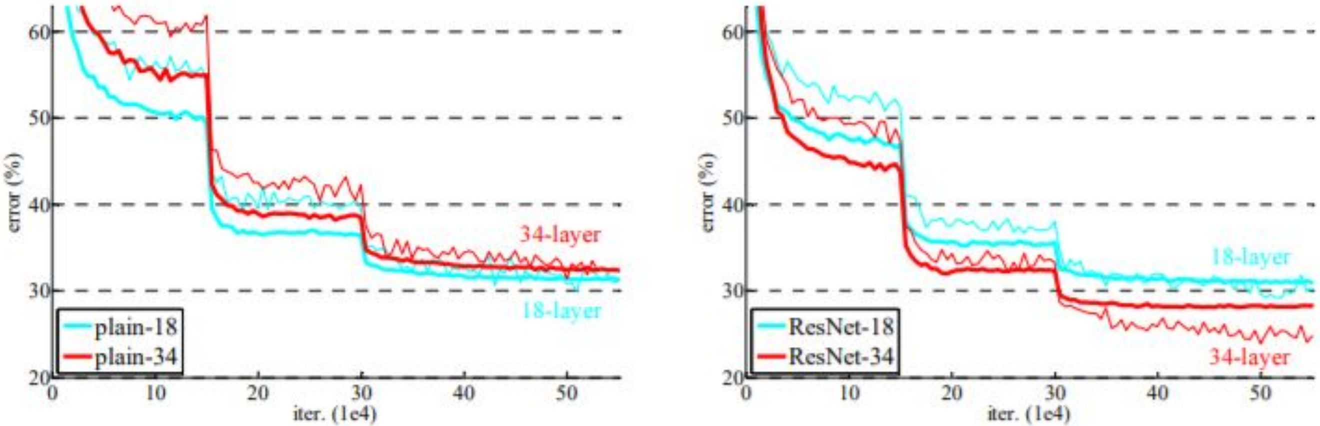
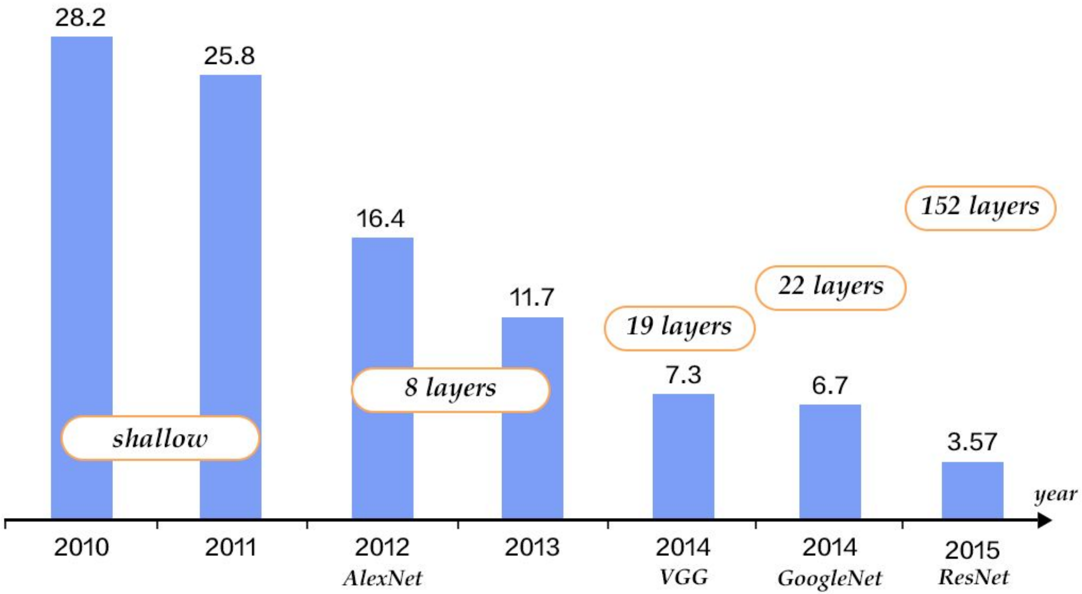


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

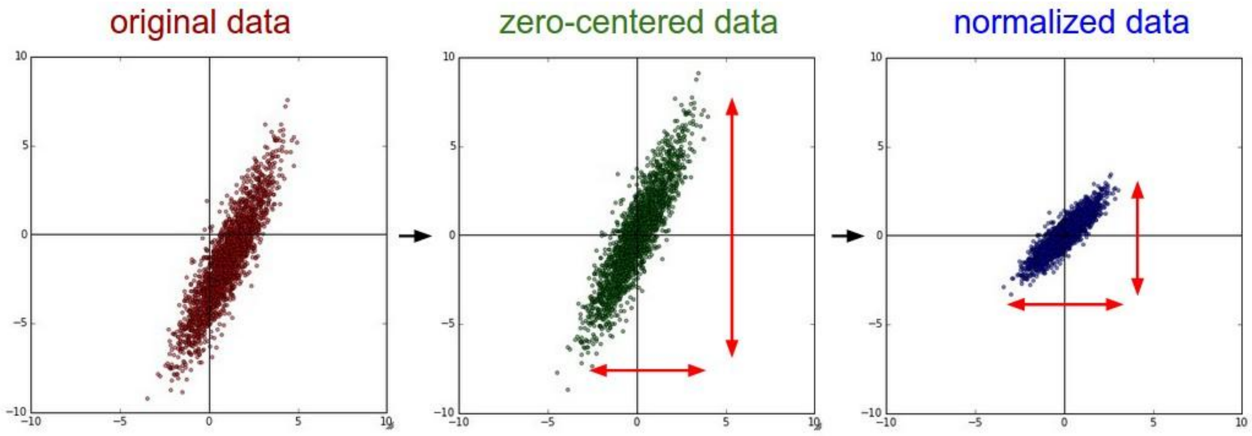
ILSVRC – Deeper and Better



Recap: Input Normalization

- Input mean and stdev normalized -> μ=0, σ=1

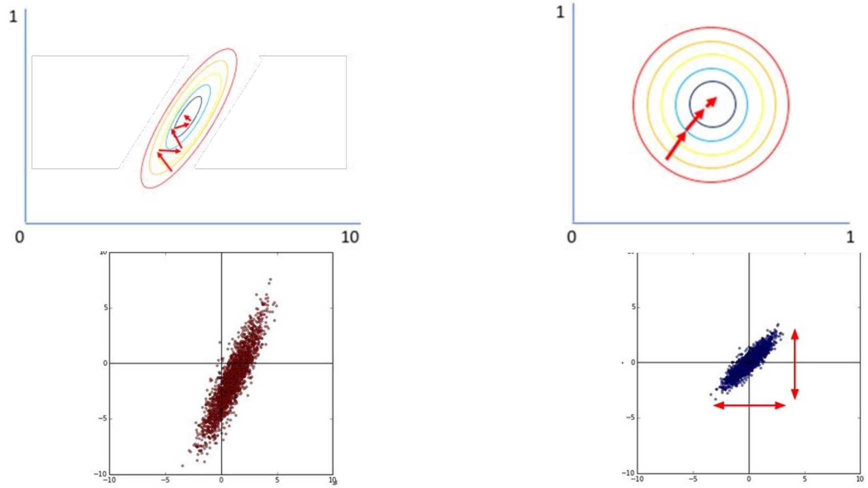
$$x_{i,j} = x_i - \mu(x_{i,j}) / \sigma(x_{i,j})$$



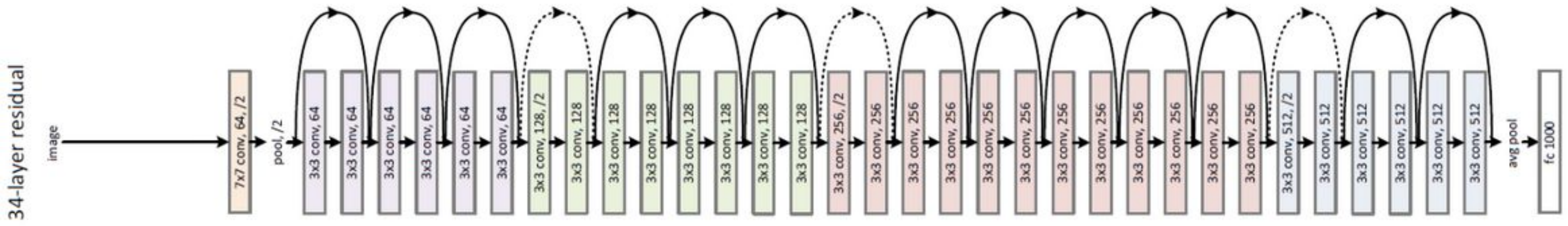
http://cs231n.github.io/neural-networks-2/

Recap: Input Normalization

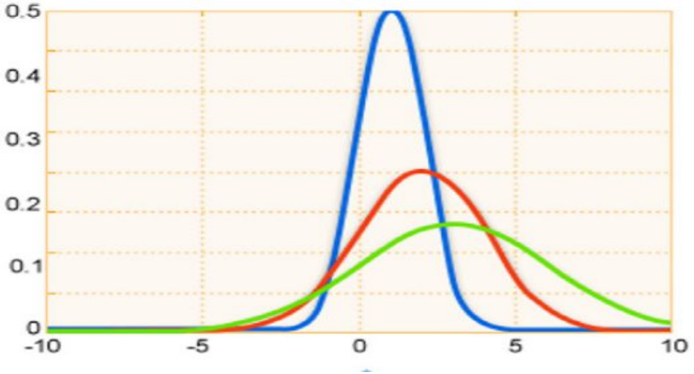
- All w_i updated according to same step-size/LR η
- Assumption: dE/dw_i comparable for all w_i
    - Otherwise, we would need separate η_i (complex problem)



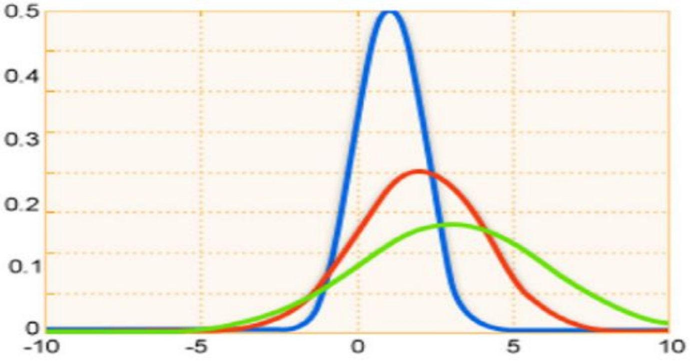
Layers: Batch-normalization

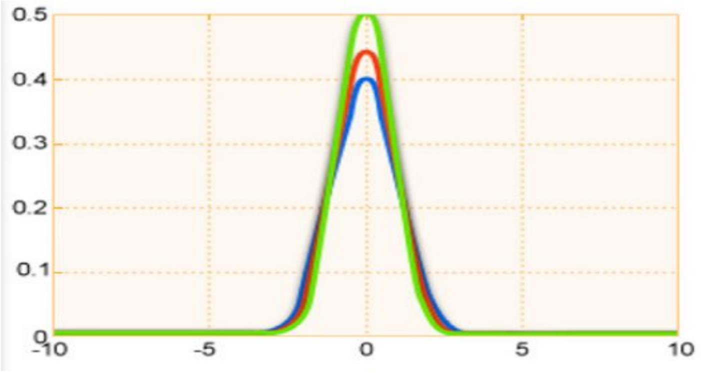


- Problem: how to estimate μ, σ of hidden layers inputs?



Layers: Batch-normalization





Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

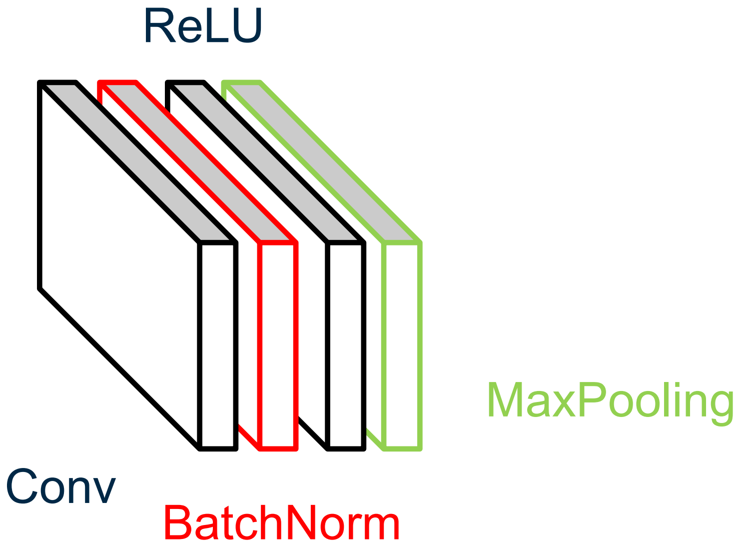
$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift
Sergey Ioffe, Christian Szegedy

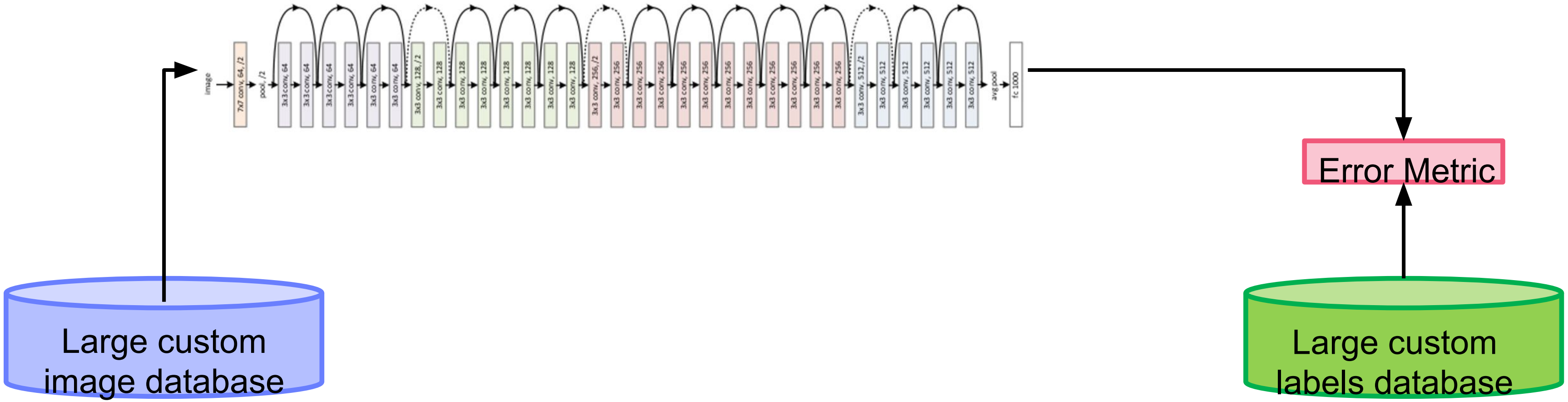
Convolve-ReLU-Pool-BatchNorm



S. Ioffe, C. Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." arXiv preprint arXiv:1502.03167 (2015).

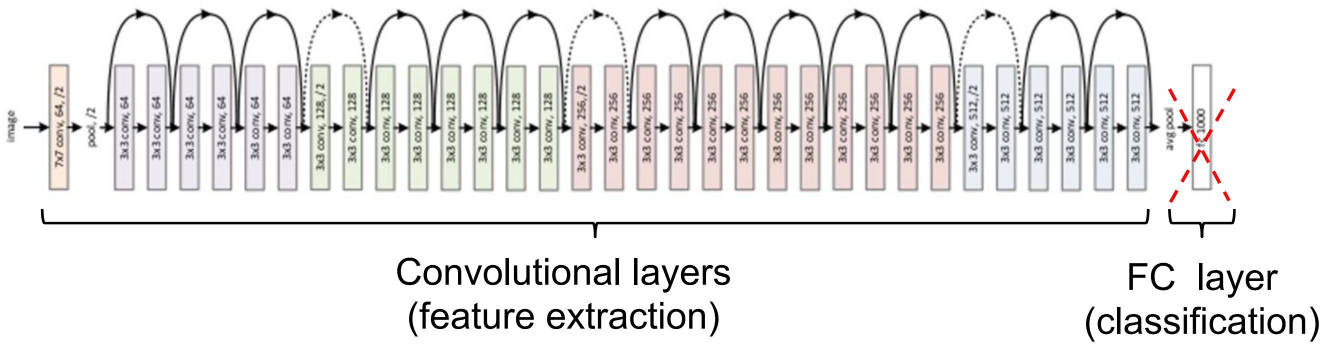
Training from Scratch

- Train *ResNet* to recognize K custom objects classes
  - Long training time
  - Must collect and label **many** train samples



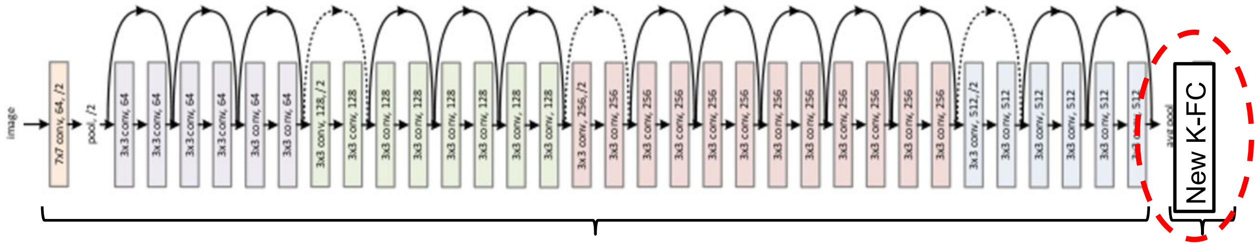
Transfer Learning

- Take ResNet pretrained on *ImageNet*



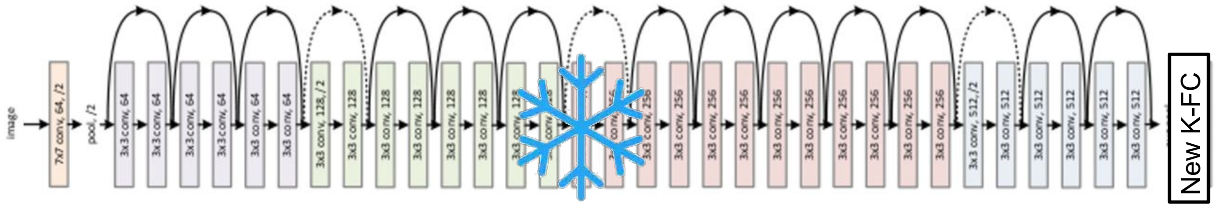
Transfer Learning

- Take ResNet pretrained on *ImageNet*
- Replace FC layer(s) with ad-hoc K-units FC layer



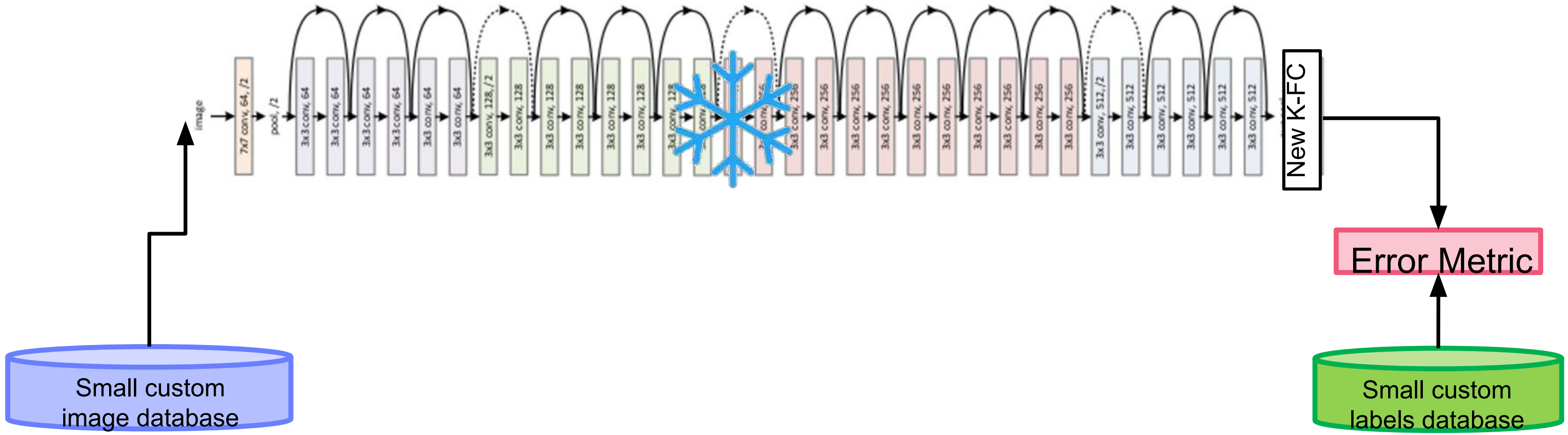
Transfer Learning

- Take ResNet pretrained on *ImageNet*
- Replace FC layer(s) with ad-hoc K-units FC layer
- *Freeze* (early) convolutional layers (η=0 or close to)



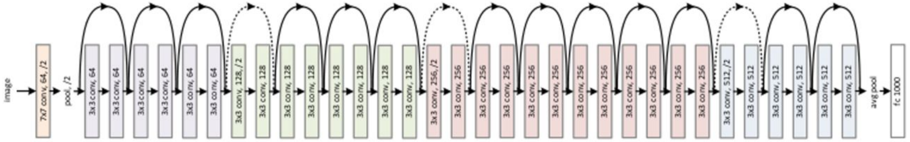
Transfer Learning

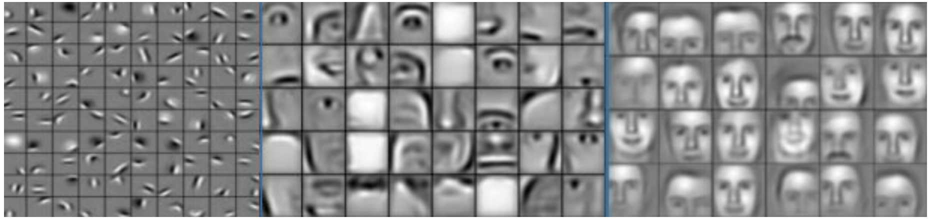
- Take ResNet pretrained on *ImageNet*
- Replace FC layer(s) with ad-hoc K-units FC layer
- *Freeze* (early) convolutional layers (η=0 or close to)



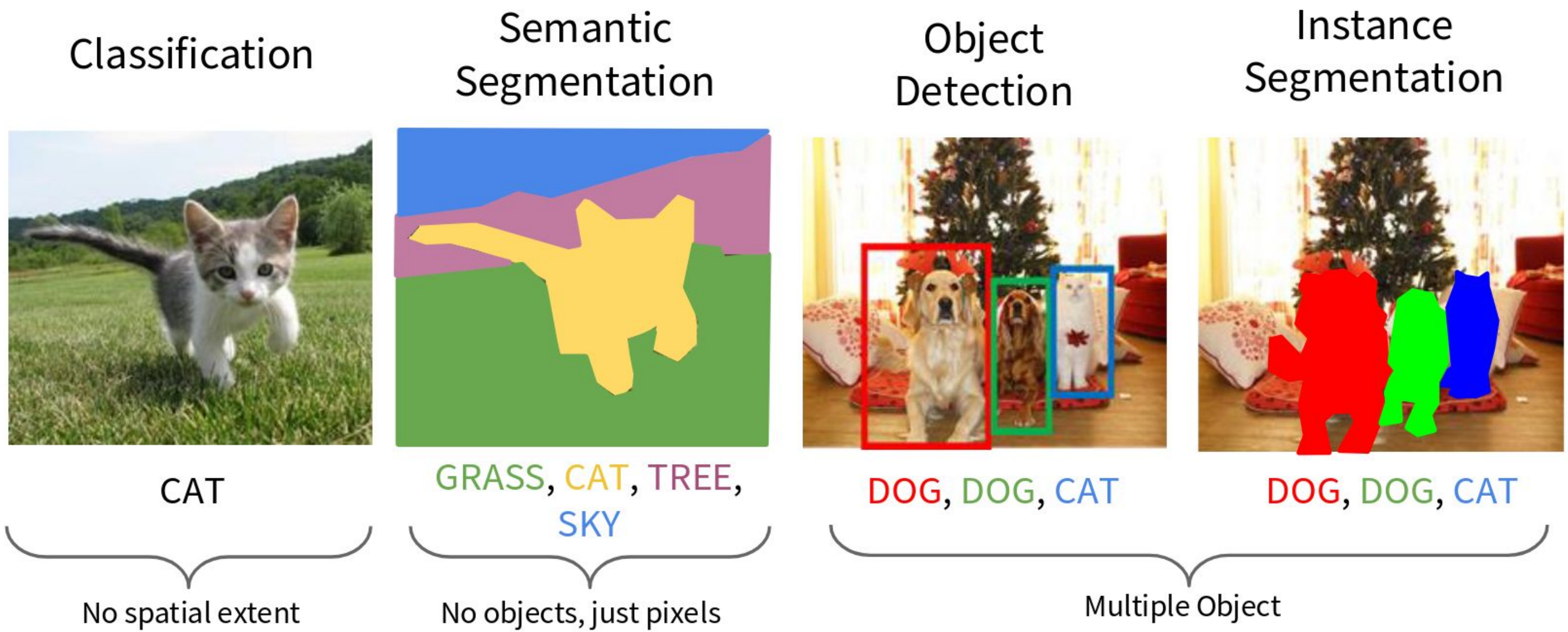
Transfer Learning – Why it Works ?

- Early conv. layers more difficult to train (faint error gradients)
  - Very low level filters (edges, etc.)
    - «Reusing» pre-learned feature detectors





CNNs for Computer Vision Tasks



[Li et al., CS231n, Course Slide from Lecture 9]

References (1)

Y. LeCun, L. Bottou, Y. Bengio and P. Haffner. “*Gradient-Based Learning Applied to Document Recognition*”, Proceedings of the IEEE 86, no. 11 (1998): 2278-2324.

D. Williams, G. Hinton. "*Learning representations by back-propagating errors*." Nature 323, no. 6088 (1986): 533-538.

A. Krizhevsky, I. Sutskever, G. E. Hinton. "*Imagenet classification with deep convolutional neural networks*." In Advances in neural information processing systems, pp. 1097-1105. 2012.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. "*Going deeper with convolutions*." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-9. 2015.

K. Simonyan, A. Zisserman. "*Very deep convolutional networks for large-scale image recog*nition." arXiv preprint arXiv:1409.1556 (2014).

P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun. "*Overfeat: Integrated recognition, localization and detection using convolutional networks*." arXiv preprint arXiv:1312.6229 (2013).

S. Srinivas, R. K. Sarvadevabhatla, K. R. Mopuri, N. Prabhu, S. Kruthiventi, R. V. Babu. "*A Taxonomy of Deep Convolutional Neural Nets for Computer Vision*." arXiv preprint arXiv:1601.06615 (2016).

G.S. Hsu, J.C. Chen, Y.Z. Chung. "*Application-oriented license plate recognition*." IEEE Transactions on Vehicular technology 62, no. 2 (2013): 552-561.

H. Li, C. Shen. "*Reading Car License Plates Using Deep Convolutional Neural Networks and LSTMs*." arXiv preprint arXiv:1601.05610 (2016).

X. Glorot, A. Bordes, Y. Bengio. "Deep Sparse Rectifier Neural Networks." In *Aistats*, vol. 15, no. 106, p. 275. 2011.

References (2)

Jürgen Schmidhuber. "Deep learning in neural networks: An overview." *Neural networks* 61 (2015): 85-117.

C. M. Bishop. “Neural networks for pattern recognition” Oxford university press, 1995.

Jürgen Schmidhuber. “Deep learning in neural networks: An overview”, Neural Networks Volume 61, pp. 85–117 (2015)

The Stanford UFLDL Deep Learning Tuttorial - http://ufldl.stanford.edu/tutorial/

I. Goodfellow, Y. Bengio, A. Courville, “Deep Learning”
http://www.deeplearningbook.org/

F. Chollet Deep Learning with Python:
https://tanthiamhuat.files.wordpress.com/2018/03/deeplearningwithpython.pdf

Questions ?