10707
Deep Learning

Russ Salakhutdinov

Machine Learning Department
rsalakhu@cs.cmu.edu

Convolutional Networks II

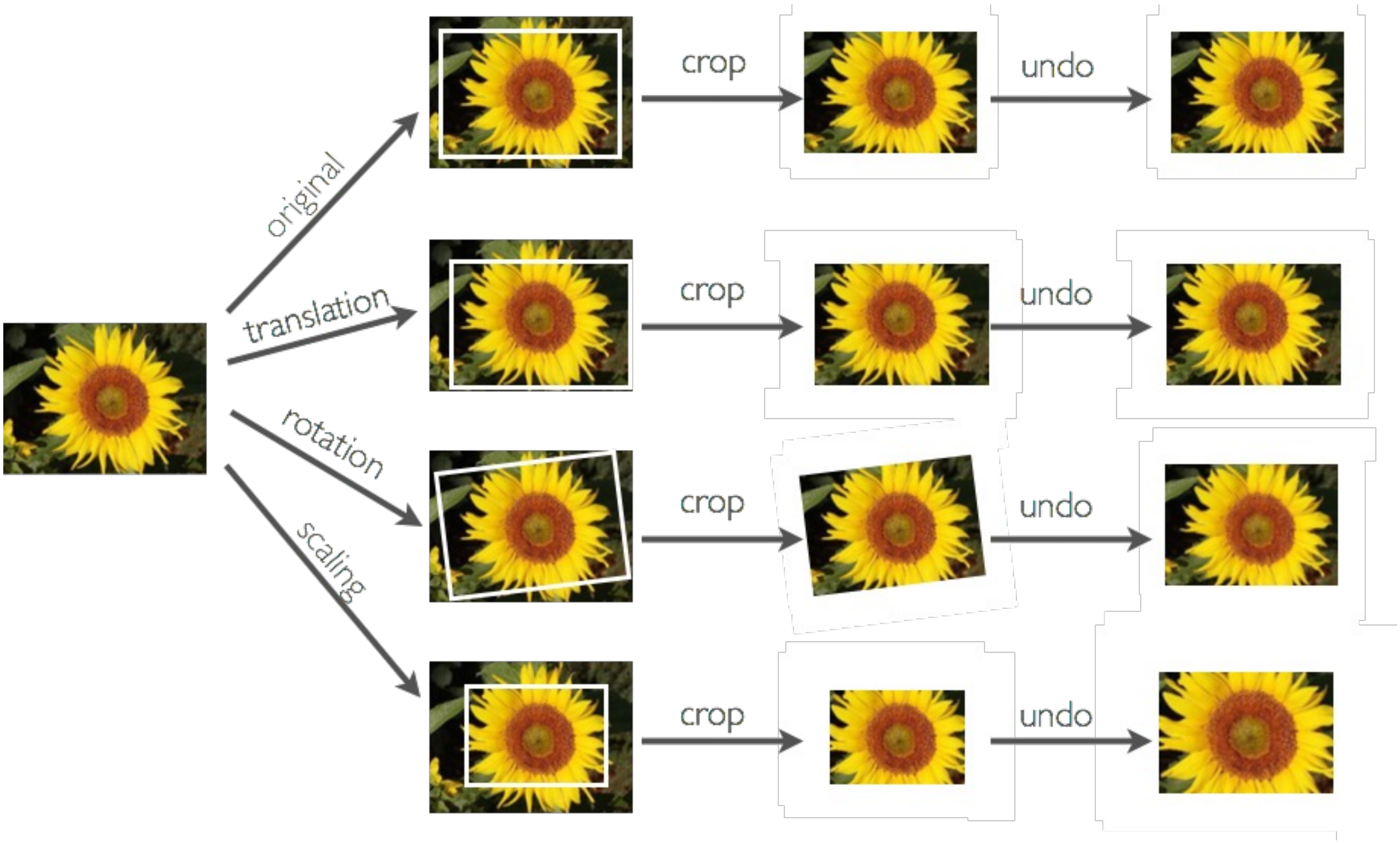
Invariance by Dataset Expansion

- Invariances built-in in convolutional network:
  - small translations: due to convolution and max pooling
  - small illumination changes: due to local contrast normalization

- It is not invariant to other important variations such as rotations and scale changes

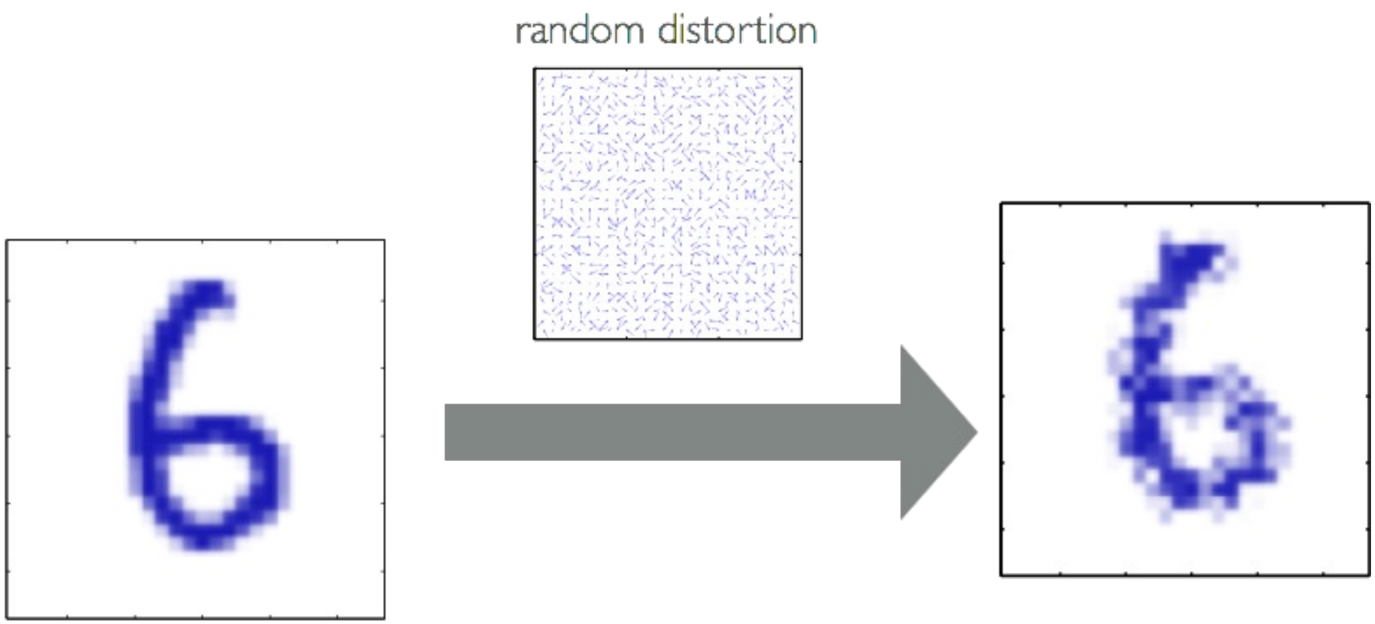
- However, it's easy to artificially generate data with such transformations
  - could use such data as additional training data
  - neural network can potentially learn to be invariant to such transformations

Generating Additional Examples



Elastic Distortions

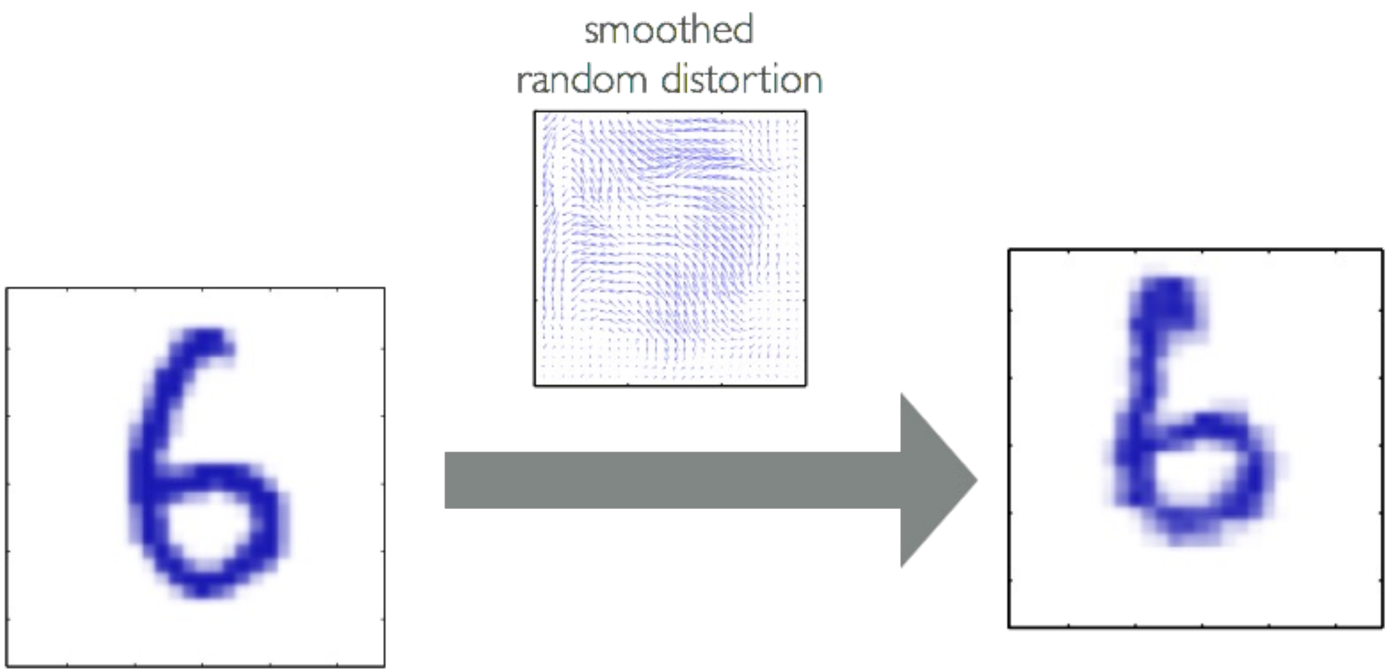
- Can add "elastic" deformations (useful in character recognition)
- We can do this by applying a "distortion field" to the image
  - a distortion field specifies where to displace each pixel value



Bishop's book

Elastic Distortions

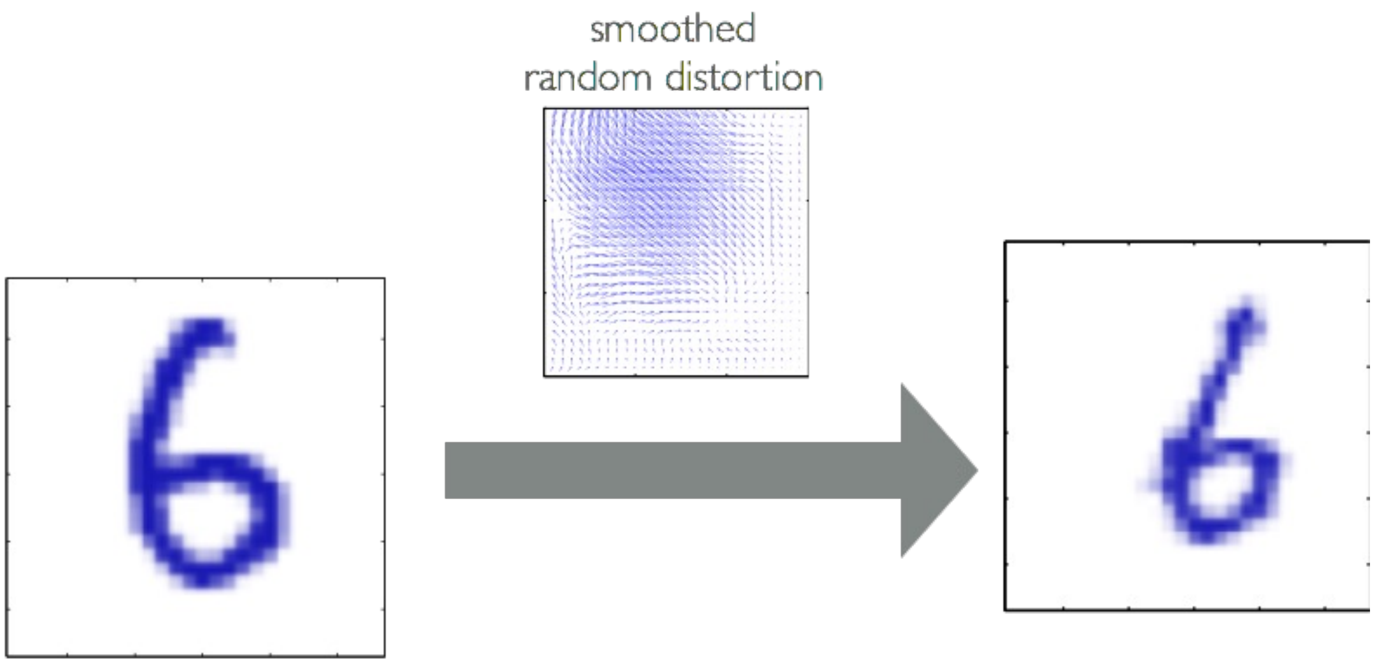
- Can add "elastic" deformations (useful in character recognition)
- We can do this by applying a "distortion field" to the image
  - ➢ a distortion field specifies where to displace each pixel value



Bishop's book

Elastic Distortions

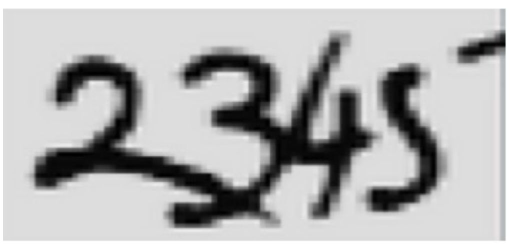
- Can add "elastic" deformations (useful in character recognition)
- We can do this by applying a "distortion field" to the image
  - ➢ a distortion field specifies where to displace each pixel value



Bishop's book

Conv Nets: Examples

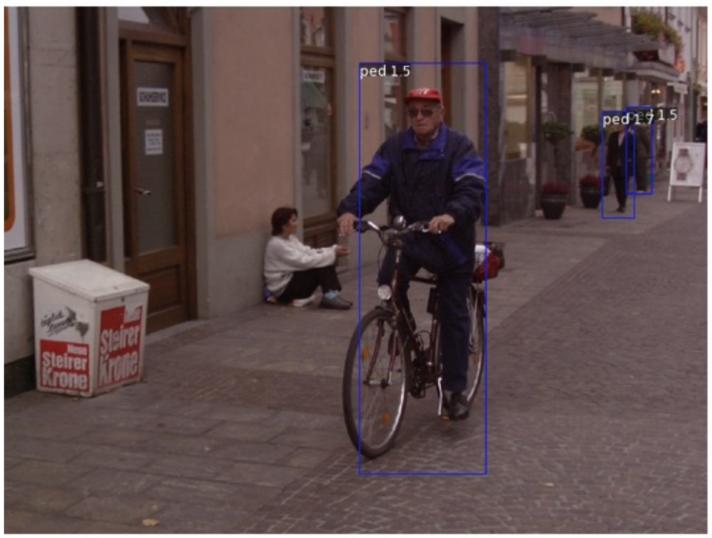
- Optical Character Recognition, House Number and Traffic Sign classification

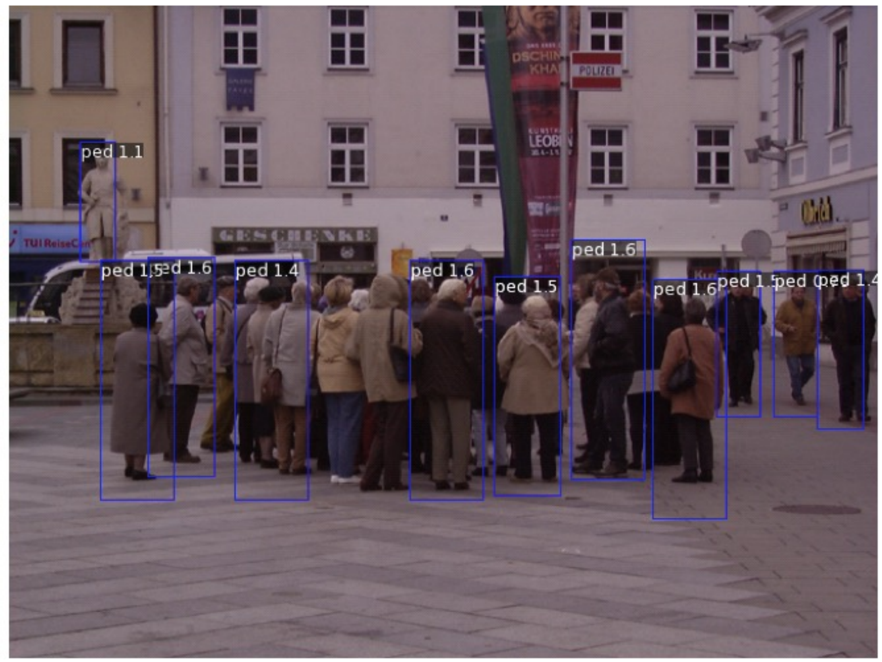


Ciresan et al. “MCDNN for image classification” CVPR 2012
Wan et al. “Regularization of neural networks using dropconnect” ICML 2013
Goodfellow et al. “Multi-digit nuber recognition from StreetView...” ICLR 2014
Jaderberg et al. “Synthetic data and ANN for natural scene text recognition” arXiv 2014

Conv Nets: Examples

- Pedestrian detection

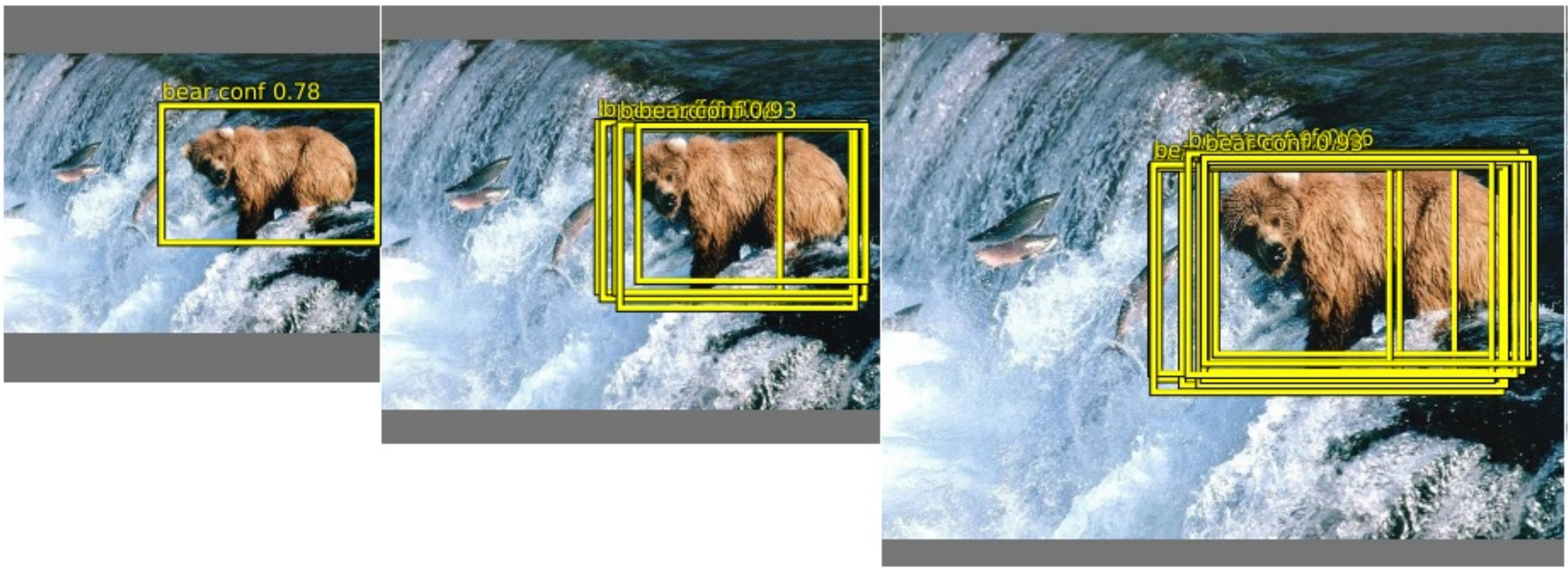




Sermanet et al. “Pedestrian detection with unsupervised multi-stage..” CVPR 2013

Conv Nets: Examples

- Object Detection



Sermanet et al. “OverFeat: Integrated recognition, localization” arxiv 2013
Girshick et al. “Rich feature hierarchies for accurate object detection” arxiv 2013
Szegedy et al. “DNN for object detection” NIPS 2013

ImageNet Dataset

- 1.2 million images, 1000 classes

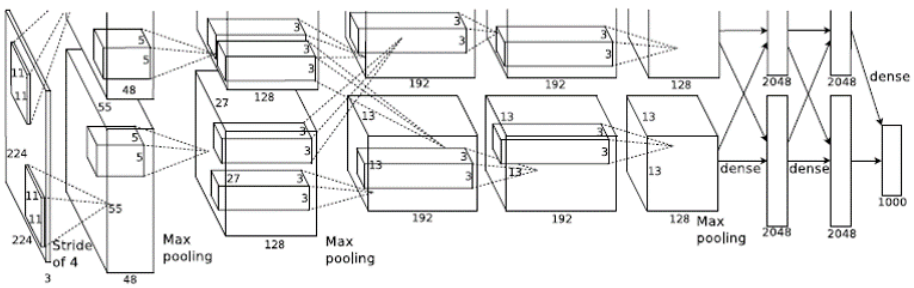
Examples of Hammer

Deng et al. “Imagenet: a large scale hierarchical image database” CVPR 2009

Important Breakthroughs

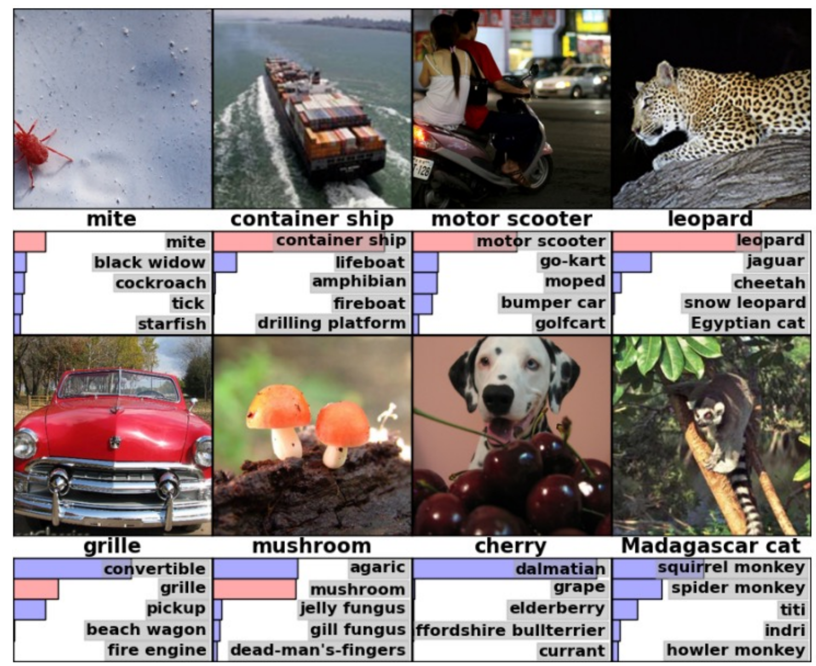
- Deep Convolutional Nets for Vision (Supervised)

  Krizhevsky, A., Sutskever, I. and Hinton, G. E., ImageNet Classification with Deep Convolutional Neural Networks, NIPS, 2012.



IMAGENET

1.2 million training images
1000 classes



Architecture

- How can we select the right architecture:
  - ➢ Manual tuning of features is now replaced with the manual tuning of architechtures

- Depth
- Width
- Parameter count

How to Choose Architecture

- Many hyper-parameters:
  - Number of layers, number of feature maps

- Cross Validation

- Grid Search (need lots of GPUs)

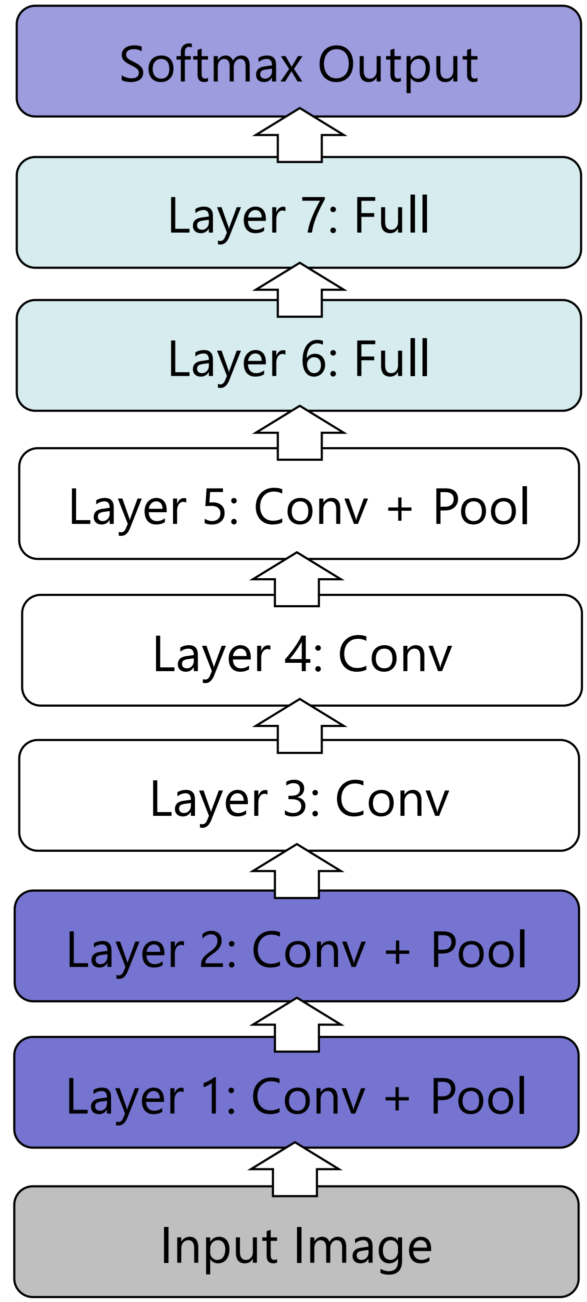
- Smarter Strategies
  - Random search
  - Bayesian Optimization

AlexNet

- 8 layers total

- Trained on Imagenet dataset [Deng et al. CVPR'09]

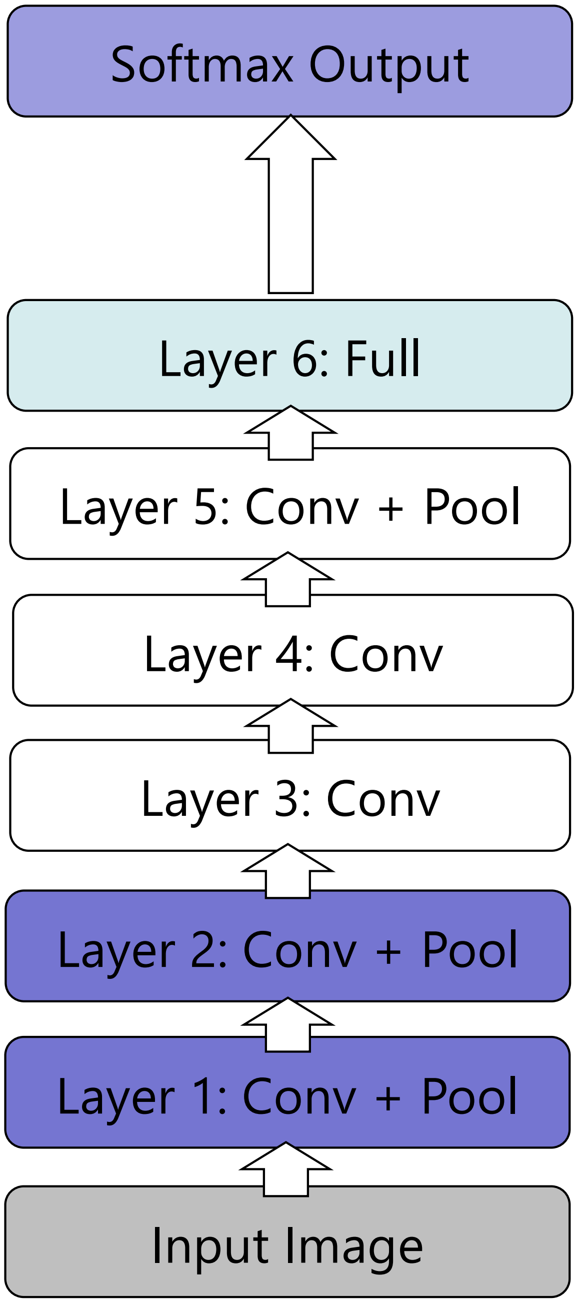
- 18.2% top-5 error



[From Rob Fergus' CIFAR 2016 tutorial]

AlexNet

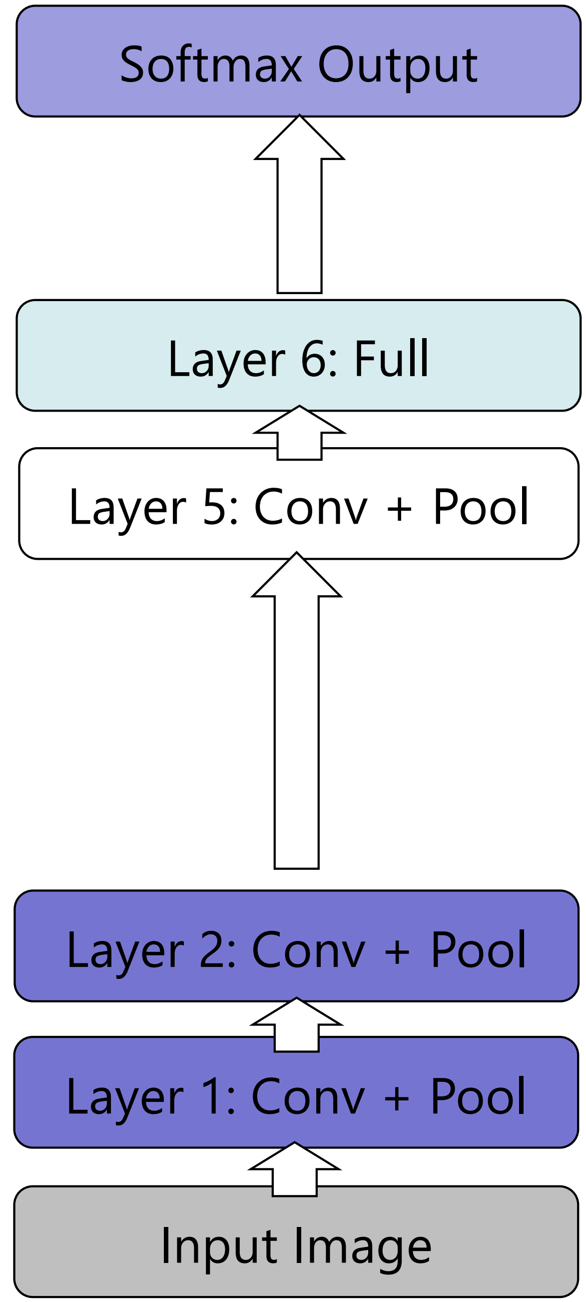
- Remove top fully connected layer 7
- Drop ~16 million parameters
- Only 1.1% drop in performance!



[From Rob Fergus' CIFAR 2016 tutorial]

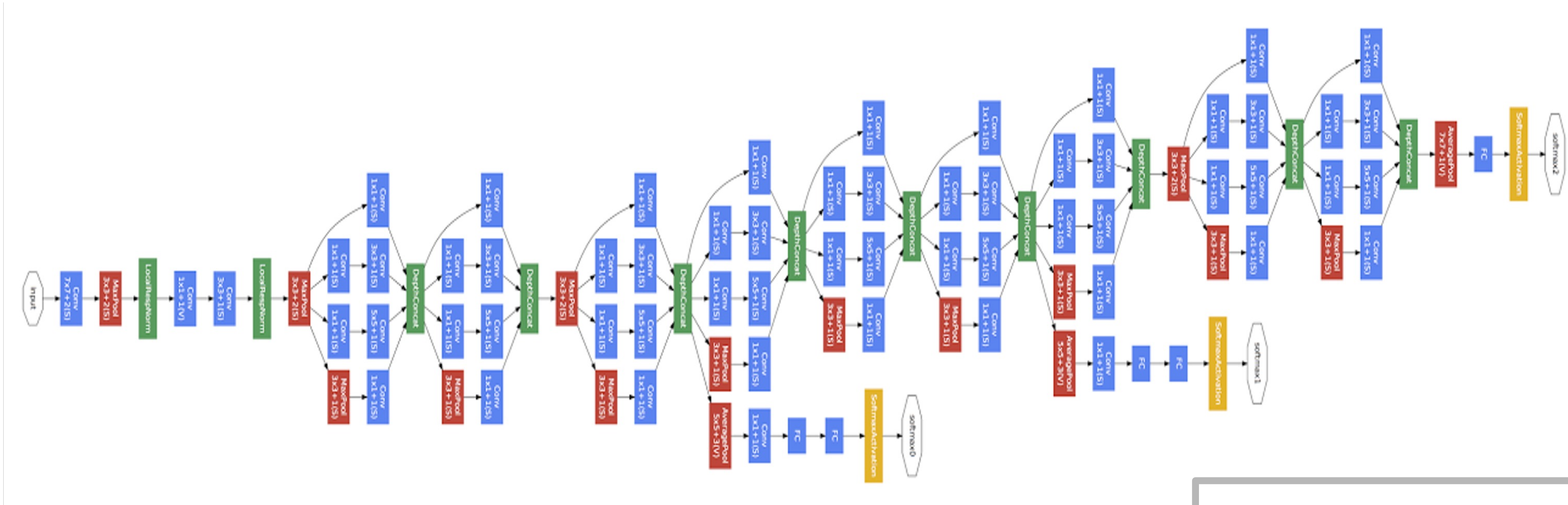
AlexNet

- Let us remove upper feature extractor layers and fully connected:
  - Layers 3,4, 6 and 7
- Drop ~50 million parameters
- **33.5 drop in performance!**
- Depth of the network is the key.



[From Rob Fergus' CIFAR 2016 tutorial]

GoogLeNet



- 24 layer model that uses so-called inception module.

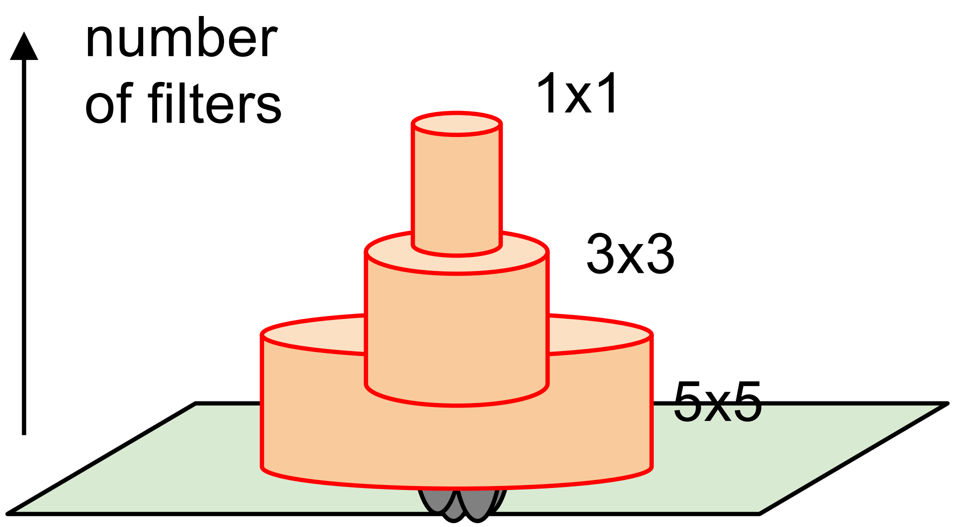
Convolution
Pooling
Softmax
Other

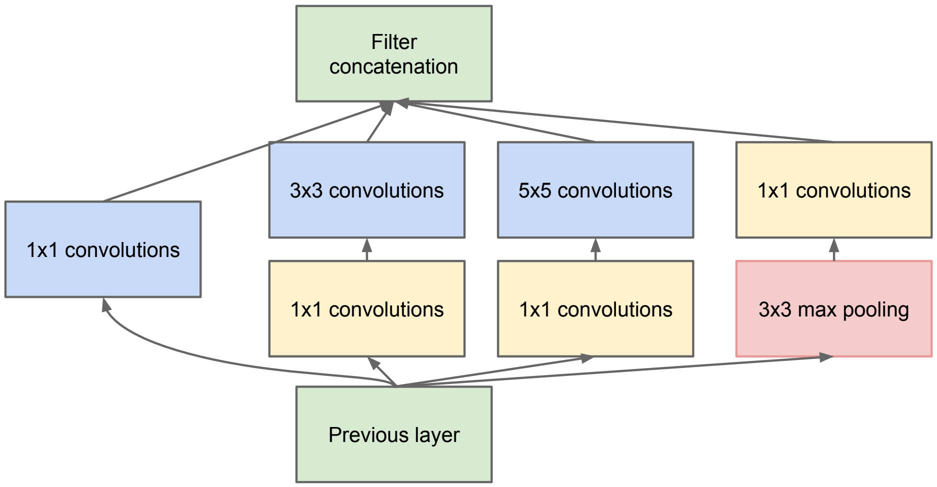
[Going Deep with Convolutions, Szegedy et al., arXiv:1409.4842, 2014]

GoogLeNet

- GoogLeNet inception module:

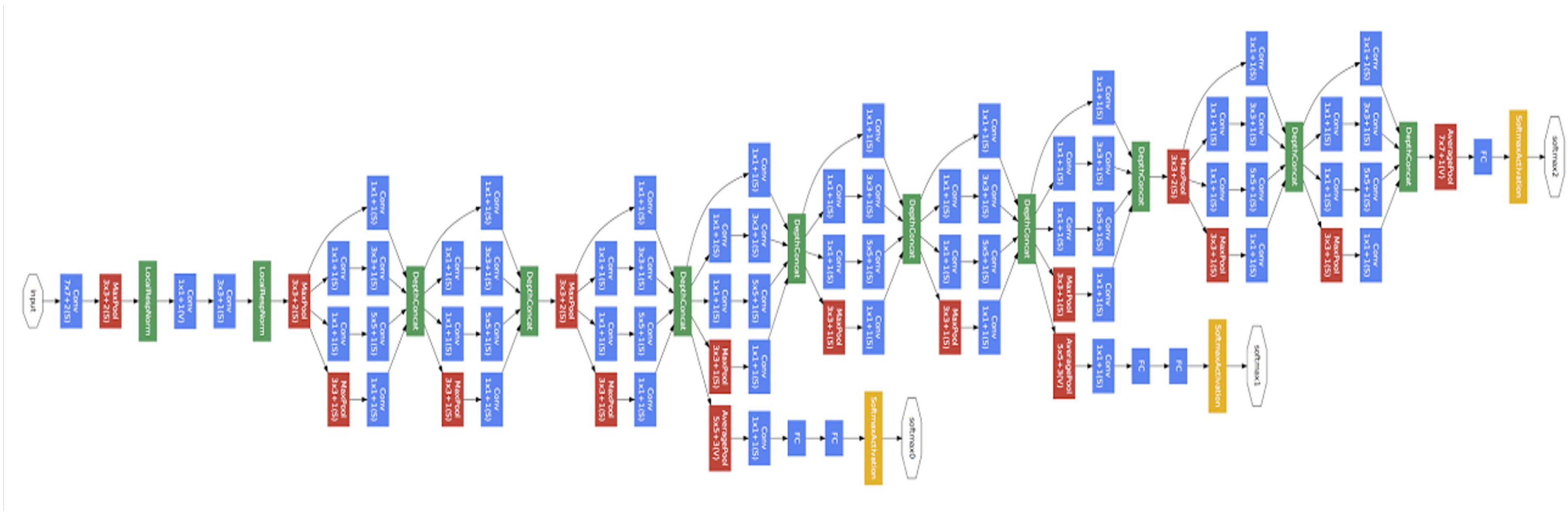
  - Multiple filter scales at each layer
  - Dimensionality reduction to keep computational requirements down





[Going Deep with Convolutions, Szegedy et al., arXiv:1409.4842, 2014]

GoogLeNet

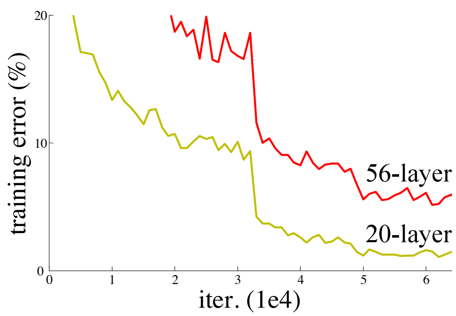


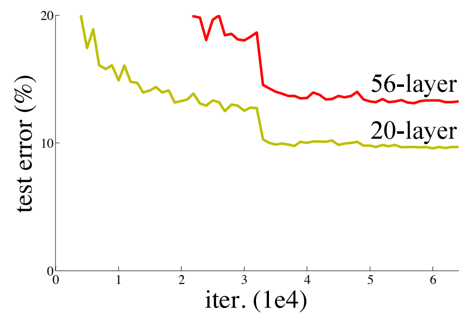
- Width of inception modules ranges from 256 filters (in early modules) to 1024 in top inception modules.
- Can remove fully connected layers on top completely
- Number of parameters is reduced to 5 million
- 6.7% top-5 validation error on Imagnet

[Going Deep with Convolutions, Szegedy et al., arXiv:1409.4842, 2014]

Residual Networks

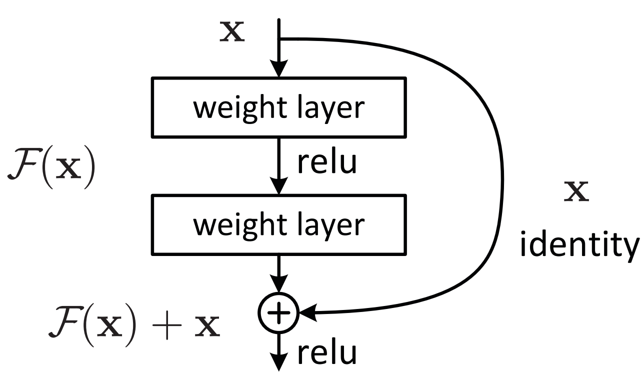
Really, really deep convnets do not train well, E.g. CIFAR10:





Key idea: introduce "pass through" into each layer

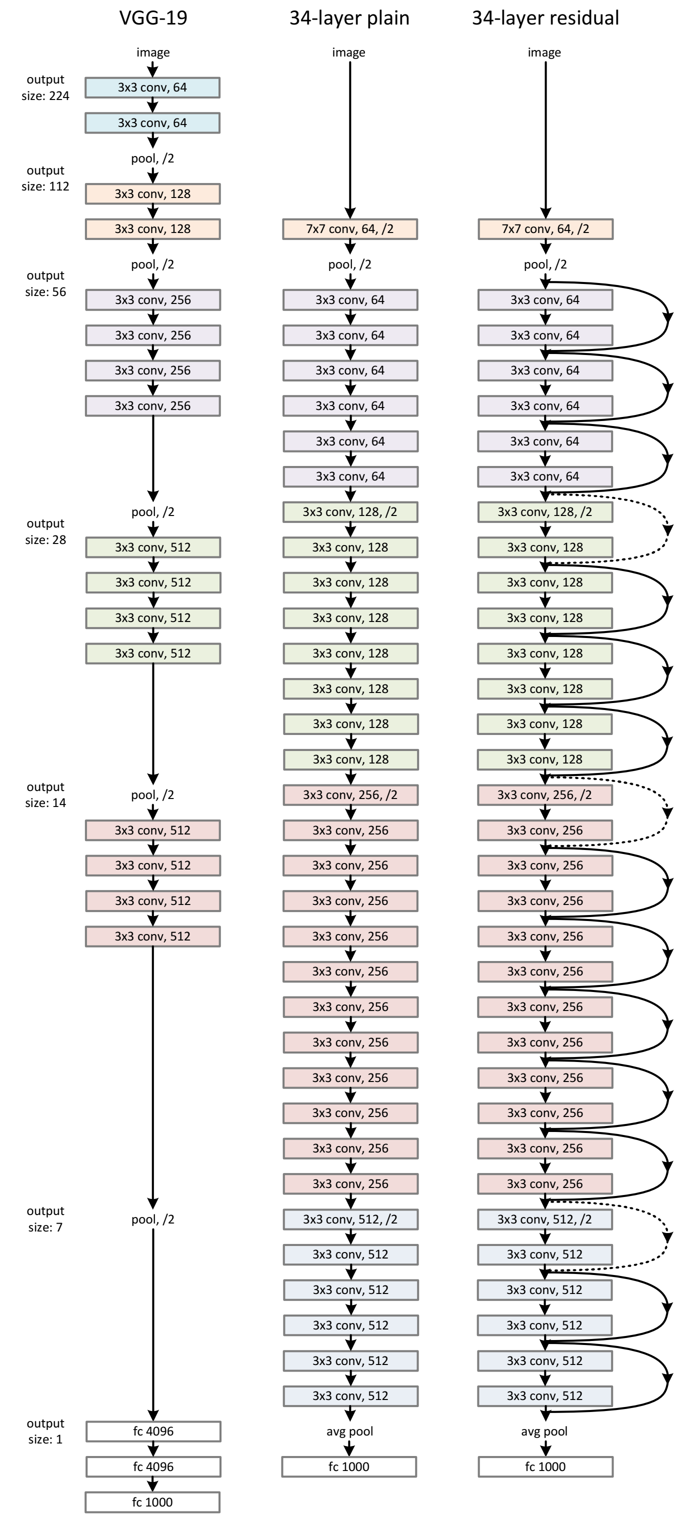
Thus only residual now needs to be learned



method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).

With ensembling, 3.57% top-5 test error on ImageNet



[He, Zhang, Ren, Sun, CVPR 2016]

Choosing the Architecture

- Task dependent
- Cross-validation
- [Convolution → pooling]* + fully connected layer
- The more data: the more layers and the more kernels
  - Look at the number of parameters at each layer
  - Look at the number of flops at each layer
- Computational resources

[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

Optimization Tricks

- SGD with momentum, batch-normalization, and dropout usually works very well

- Pick learning rate by running on a subset of the data
  - Start with large learning rate & divide by 2 until loss does not diverge
  - Decay learning rate by a factor of ~100 or more by the end of training

- Use ReLU nonlinearity

- Initialize parameters so that each feature across layers has similar variance. Avoid units in saturation.

[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

Improving Generalization

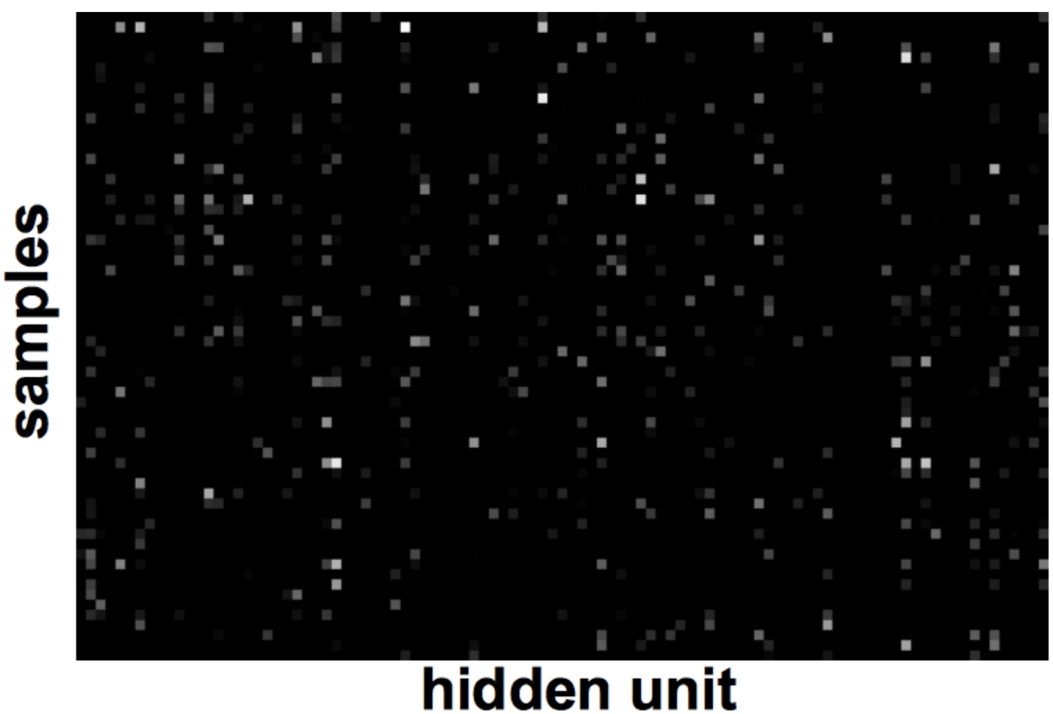
- Weight sharing (greatly reduce the number of parameters)
- Data augmentation (e.g., jittering, noise injection, etc.)
- Dropout
- Weight decay (L2, L1)
- Sparsity in the hidden units
- Multi-task (unsupervised learning)

[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

Visualization

• Check gradients numerically by finite differences

• Visualize features (feature maps need to be uncorrelated) and have high variance



• Good training: hidden units are sparse across samples

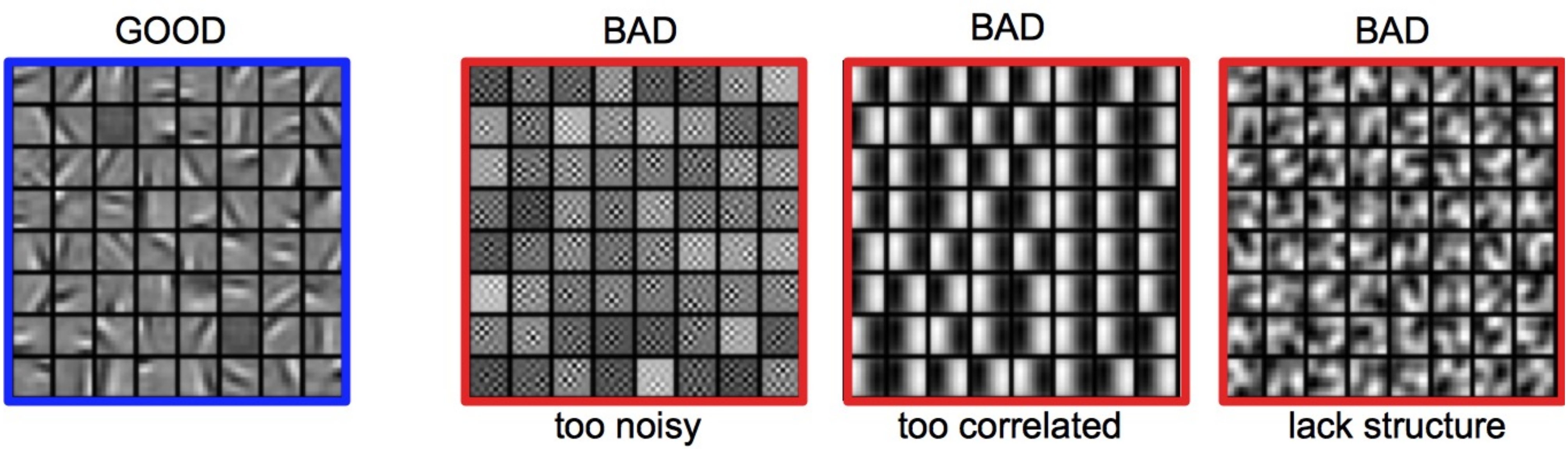
[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

Visualization

- Check gradients numerically by finite differences

- Visualize features (feature maps need to be uncorrelated) and have high variance

- Visualize parameters: learned features should exhibit structure and should be uncorrelated and are uncorrelated

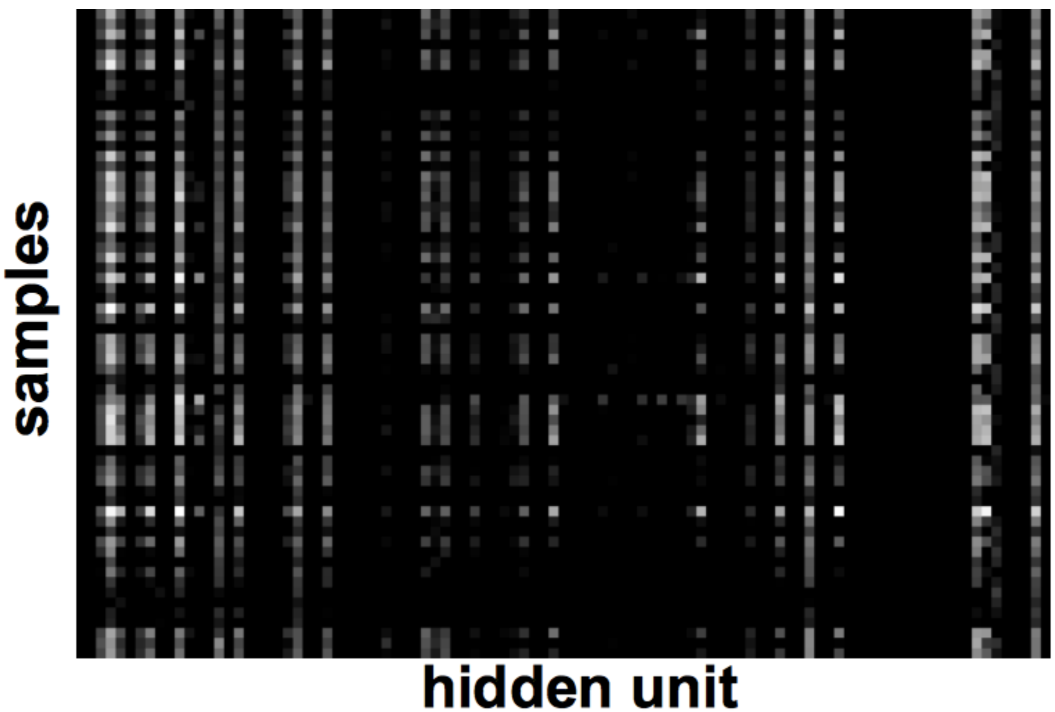


[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

Visualization

- Check gradients numerically by finite differences

- Visualize features (feature maps need to be uncorrelated) and have high variance



- Bad training: many hidden units ignore the input and/or exhibit strong correlations

[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

Visualization

- Check gradients numerically by finite differences

- Visualize features (feature maps need to be uncorrelated) and have high variance

- Visualize parameters: learned features should exhibit structure and should be uncorrelated and are uncorrelated

- Measure error on both training and validation set

- Test on a small subset of the data and check the error $\rightarrow 0$.

[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]

When it does not work

- Training diverges:
  - Learning rate may be too large → decrease learning rate
  - BPROP is buggy → numerical gradient checking
- Parameters collapse / loss is minimized but accuracy is low
  - Check loss function: Is it appropriate for the task you want to solve?
  - Does it have degenerate solutions?
- Network is underperforming
  - Compute flops and nr. params. → if too small, make net larger
  - Visualize hidden units/params → fix optimization
- Network is too slow
  - GPU,distrib. framework, make net smaller

[From Marc'Aurelio Ranzato, CVPR 2014 tutorial]