上海科技大学
ShanghaiTech University

Lecture 15: CNNs –II Architectures

Yujiao Shi
SIST, ShanghaiTech
Spring, 2025

Outline

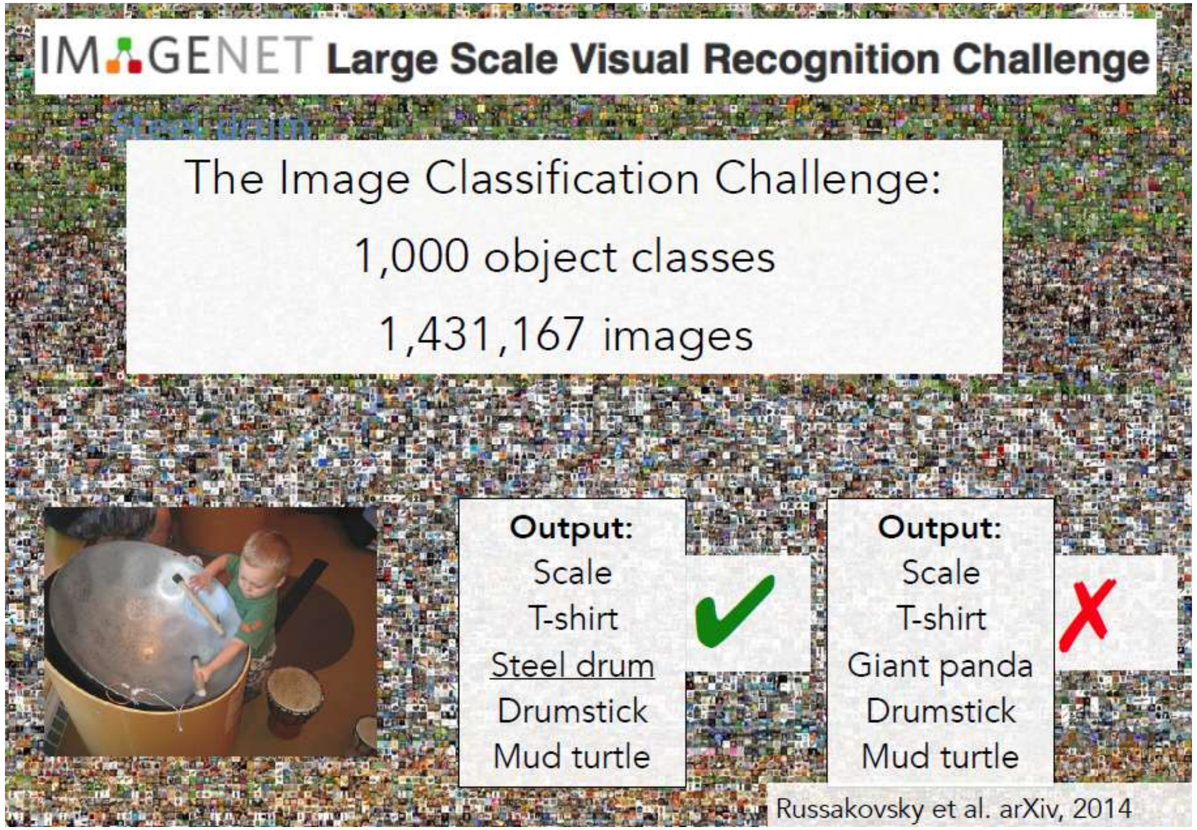
上海科技大学
ShanghaiTech University

- CNN architectures
  - Sequential structure: LeNet/AlexNet/VGGNet
  - Parallel branches: GoogLeNet
  - Residual structure: ResNet/DenseNet
  - Network Architecture Search

Acknowledgement: Zemel et al's CSC411 and Feifei Li et al's cs231n notes

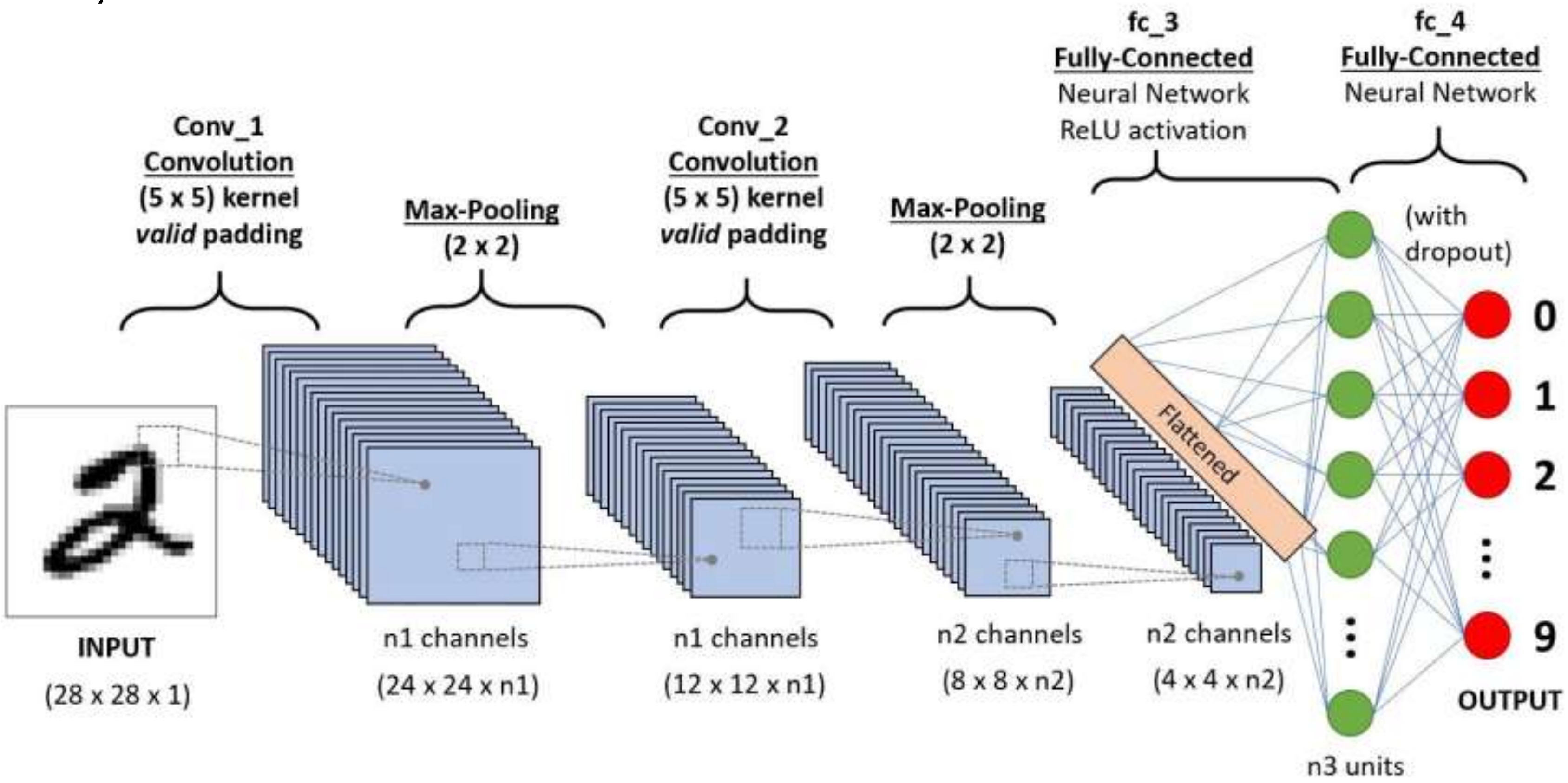
Background: Image/Object Classification

- Problem Setup
  - Input: Image
  - Output: Object class



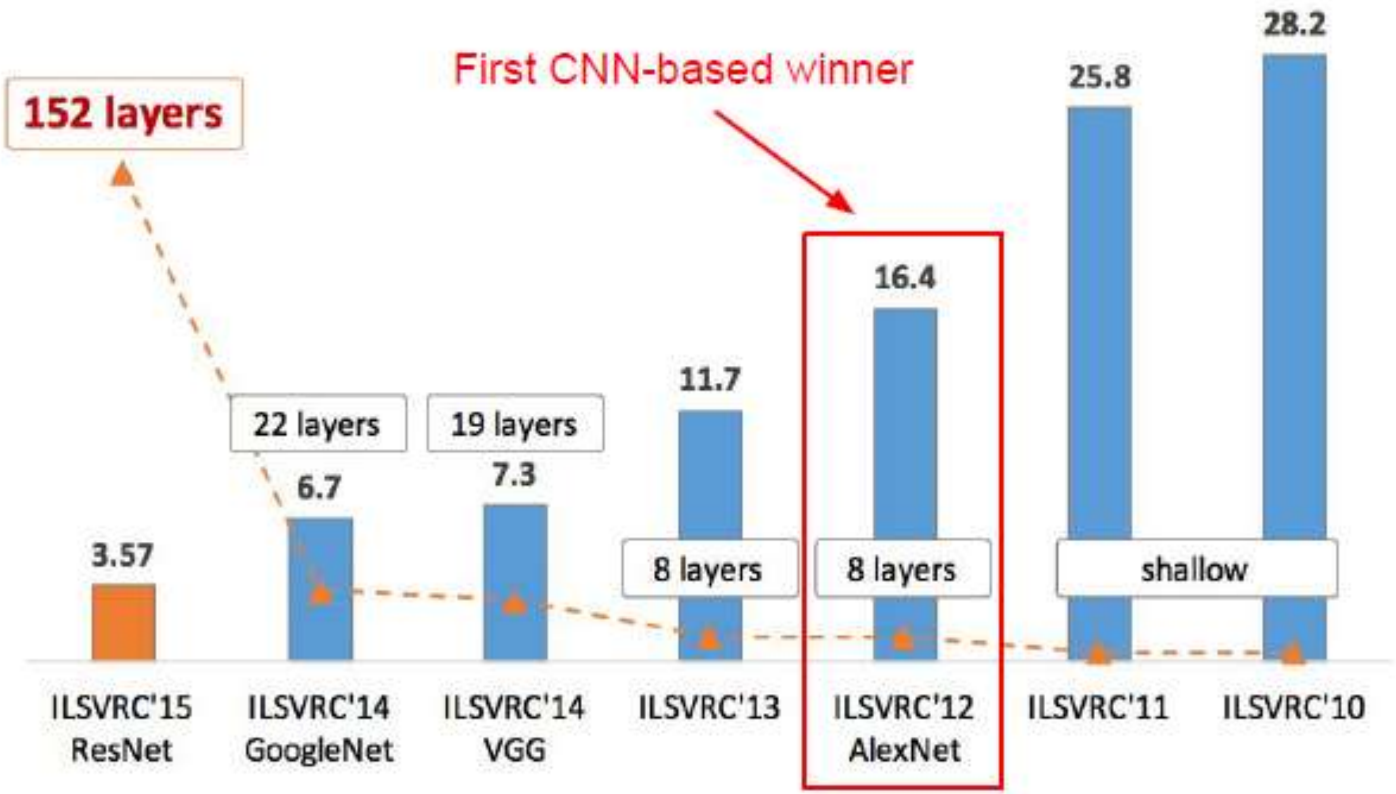
LeNet-5

- Handwritten digit recognition
- LeCun et al., 1998

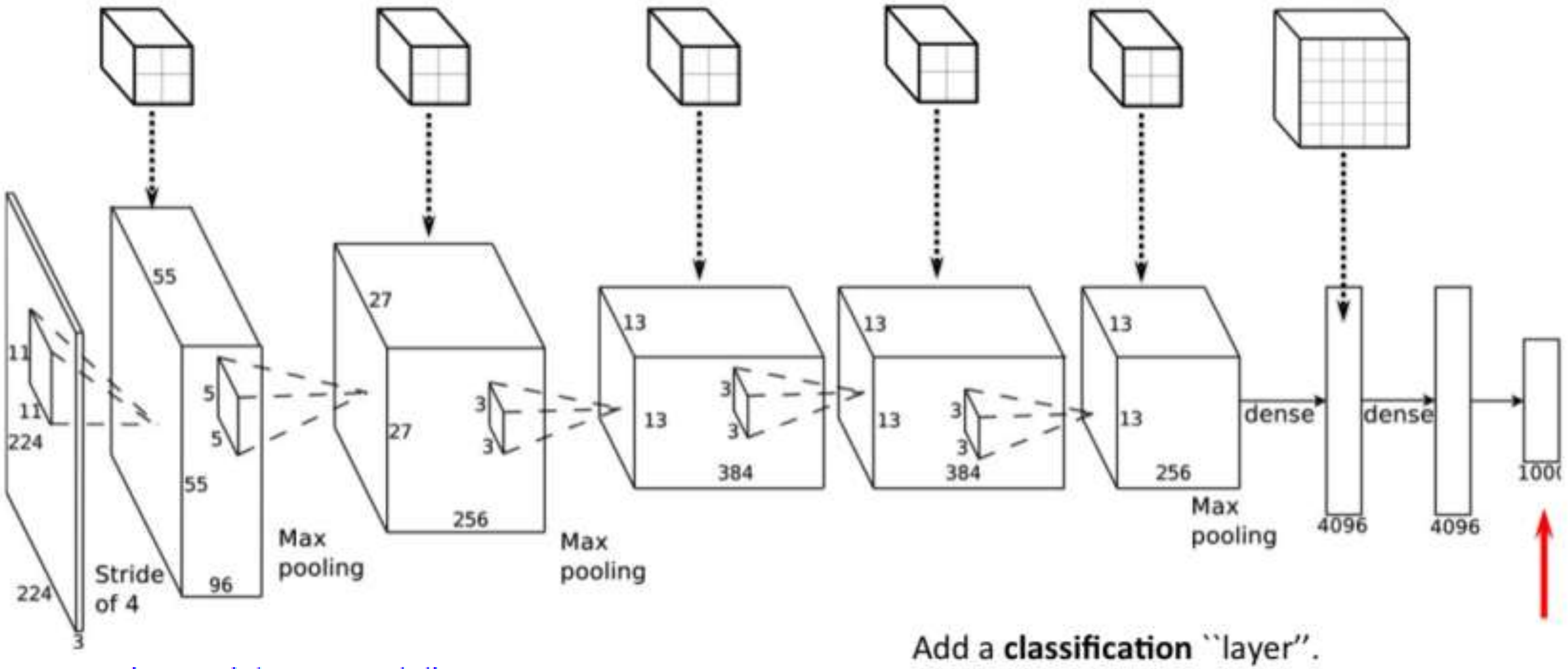


ImageNet (ILSVRC)

上海科技大学
ShanghaiTech University



AlexNet



- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble: 18.2% -> 15.4%

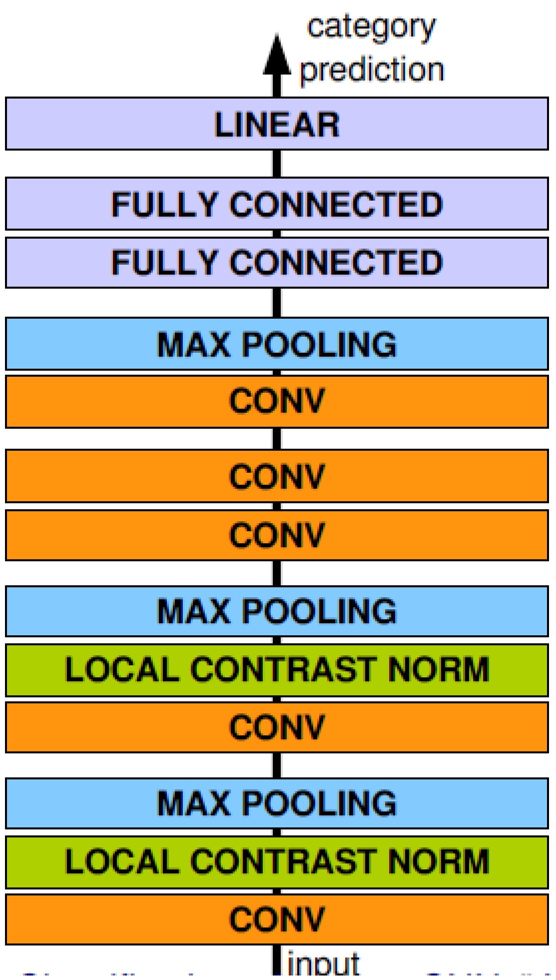
Add a **classification** ``layer''.

For an input image, the value in a particular dimension of this vector tells you the probability of the corresponding object class.

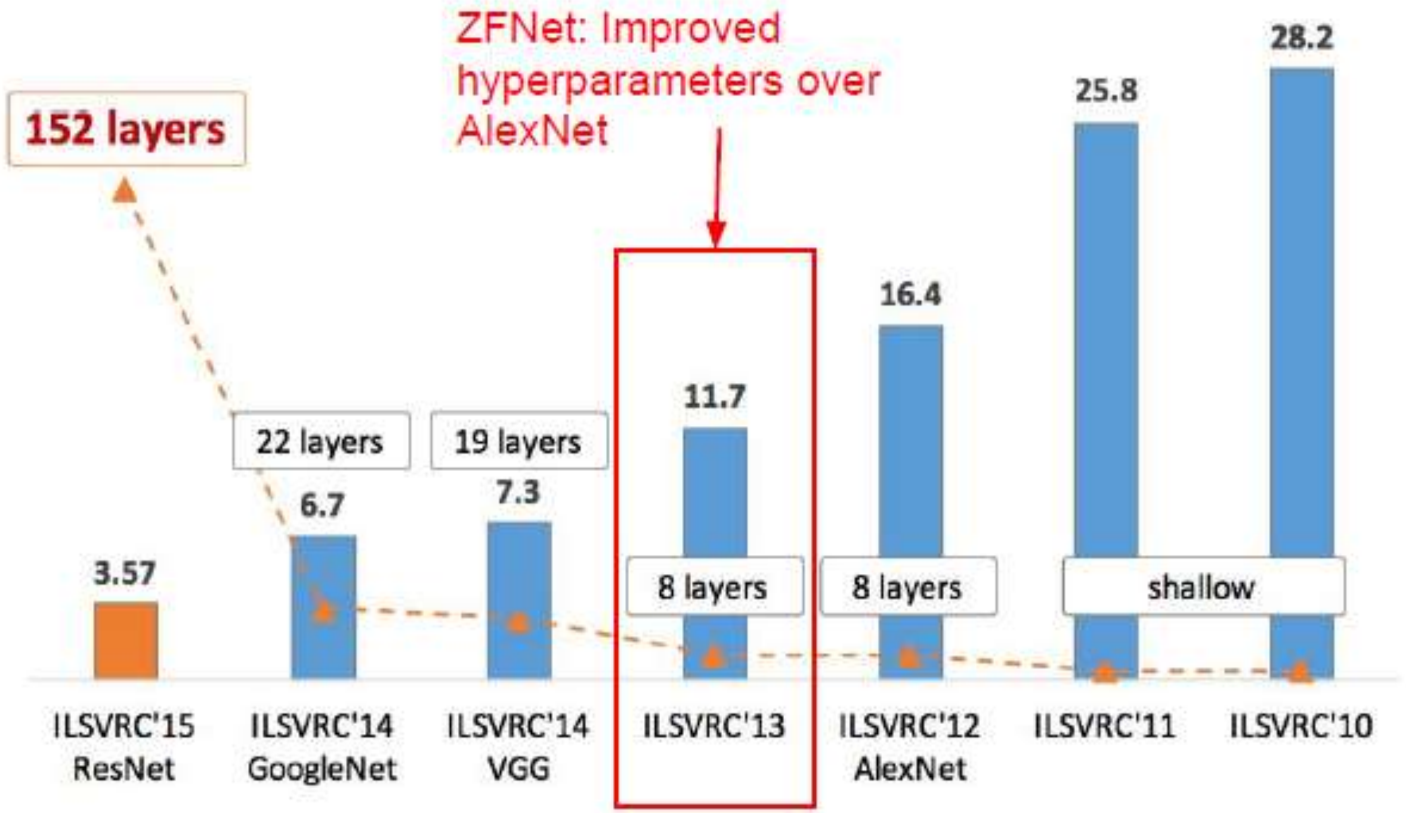
AlexNet

- Deeper network structure
  - More convolution layers
  - Local contrast normalization
  - ReLu instead of Tanh
  - Dropout as regularization

[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

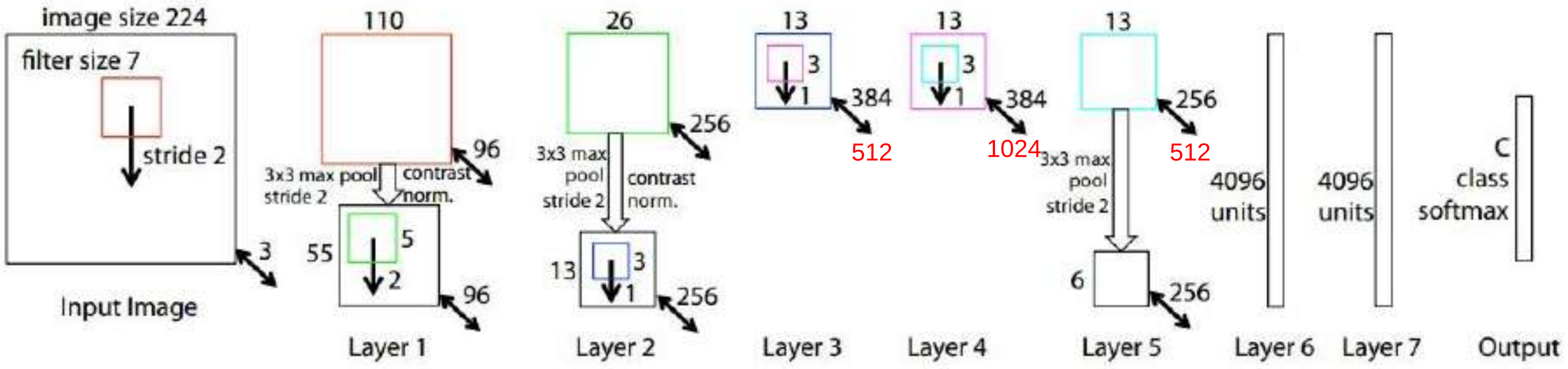


ImageNet (ILSVRC)



ZFNet

上海科技大学
ShanghaiTech University

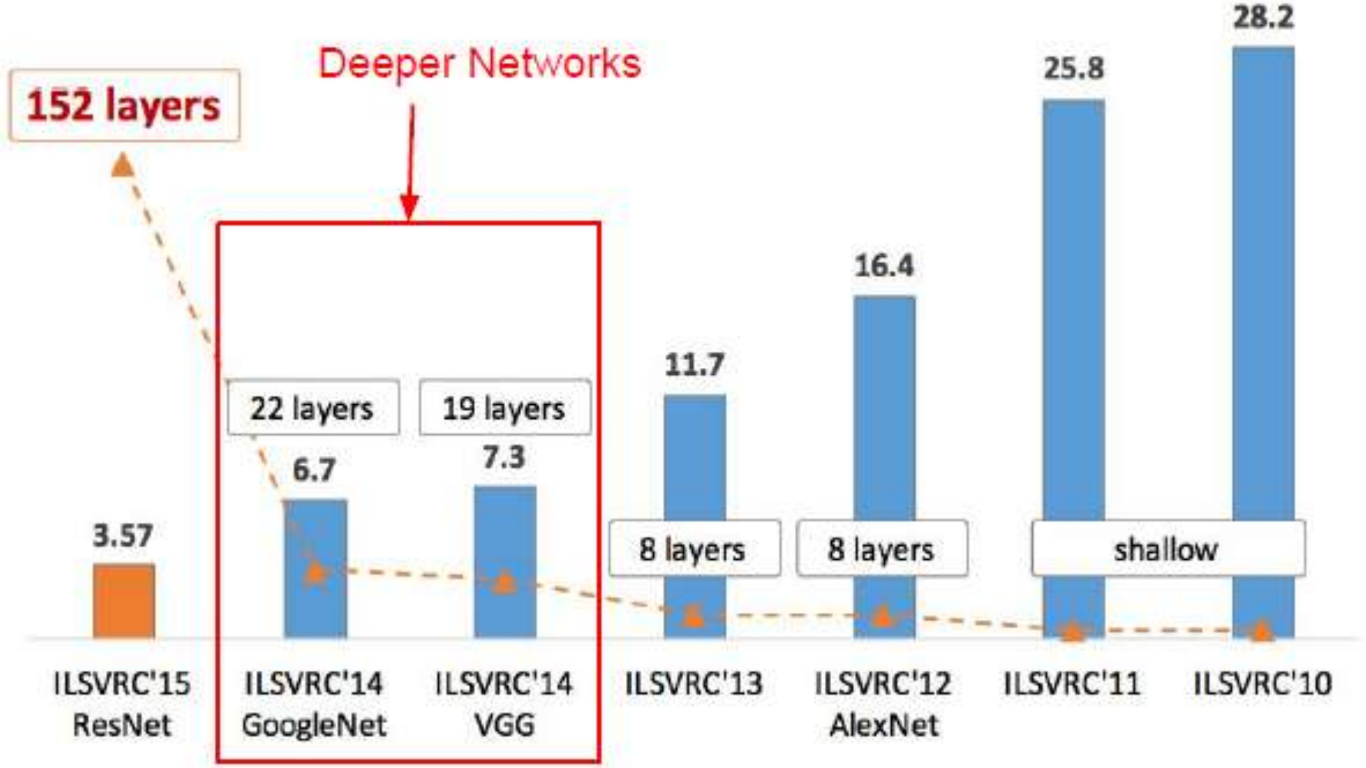


AlexNet but:
CONV1: change from (11x11 stride 4) to (7x7 stride 2)
CONV3,4,5: instead of 384, 384, 256 filters use 512, 1024, 512

ImageNet top 5 error: 16.4% -> 11.7%

ImageNet (ILSVRC)

上海科技大学
ShanghaiTech University



上海科技大学 ShanghaiTech University

Case Study: VGGNet

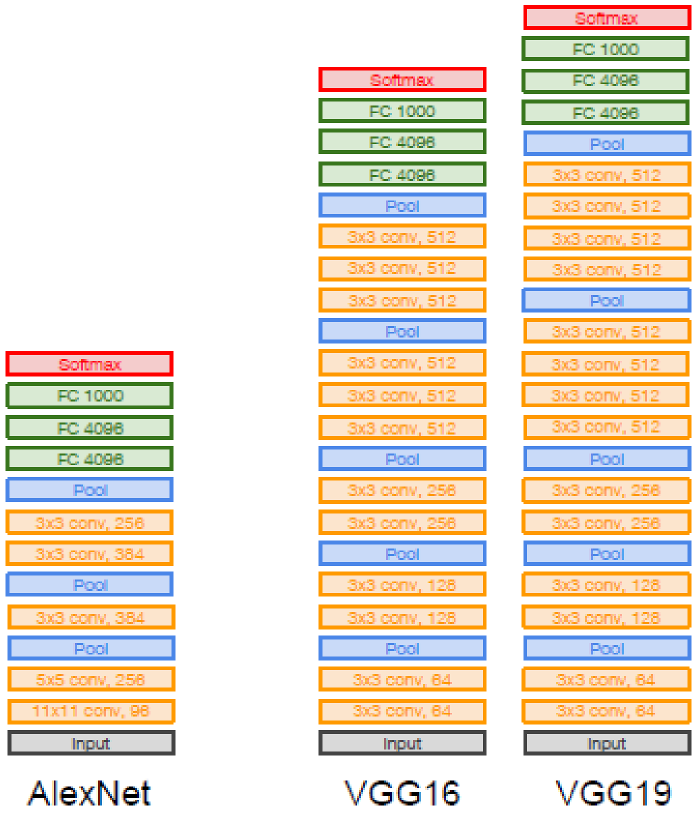
[Simonyan and Zisserman, 2014]

Small filters, Deeper networks

8 layers (AlexNet)
-> 16 - 19 layers (VGG16Net)

Only 3x3 CONV stride 1, pad 1
and 2x2 MAX POOL stride 2

11.7% top 5 error in ILSVRC'13
(ZFNet)
-> 7.3% top 5 error in ILSVRC'14



VGGNet

Case Study: VGGNet

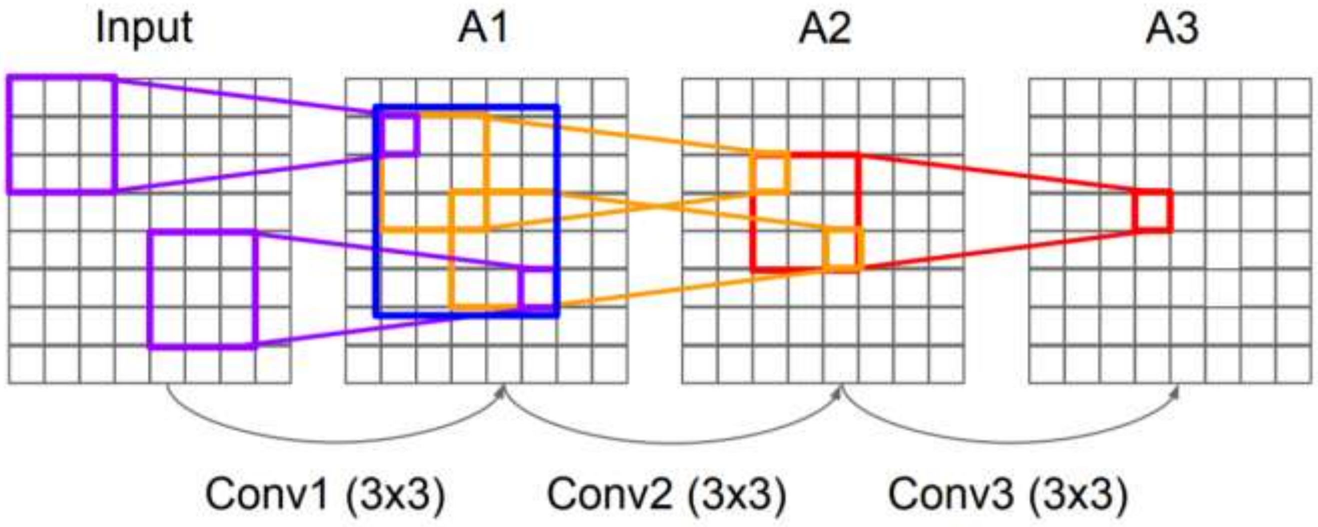
[Simonyan and Zisserman, 2014]

Q: Why use smaller filters? (3x3 conv)

Stack of three 3x3 conv (stride 1) layers has same **effective receptive field** as one 7x7 conv layer

But deeper, more non-linearities

And fewer parameters: 3 * (3^2C^2) vs. 7^2C^2 for C channels per layer



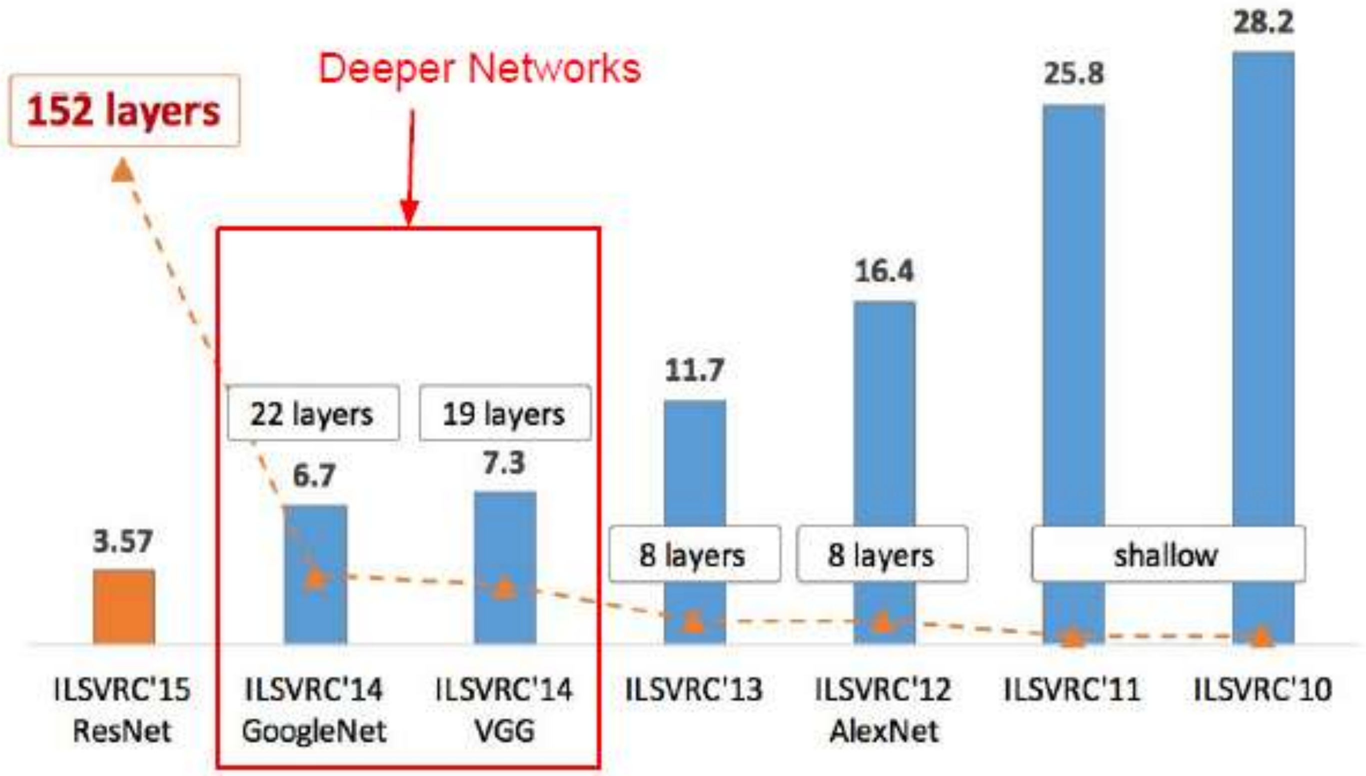
Outline

- CNN architectures
  - Sequential structure: LeNet/AlexNet/VGGNet
  - Parallel branches: GoogLeNet
  - Residual structure: ResNet/DenseNet
  - Network Architecture Search

Acknowledgement: Zemel et al's CSC411 and Feifei Li et al's cs231n notes

ImageNet (ILSVRC)

上海科技大学 ShanghaiTech University



GoogLeNet

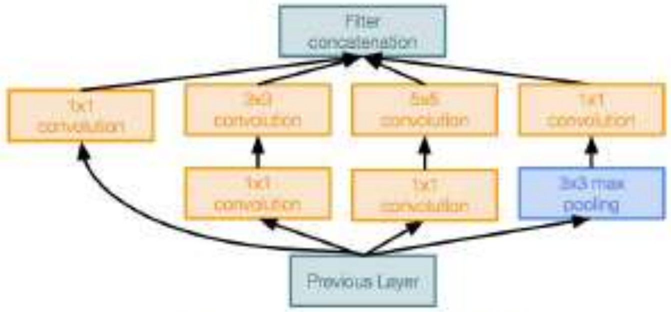
上海科技大学
ShanghaiTech University

Case Study: GoogLeNet

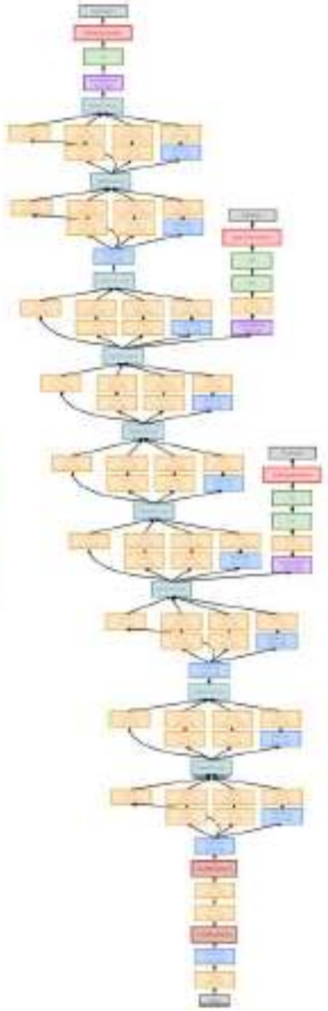
[Szegedy et al., 2014]

Deeper networks, with computational efficiency

- 22 layers
- Efficient “Inception” module
- No FC layers
- Only 5 million parameters!
  12x less than AlexNet
- ILSVRC’14 classification winner
  (6.7% top 5 error)



Inception module

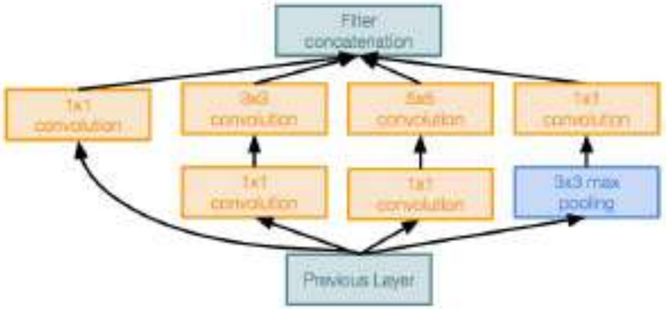


上海科技大学
ShanghaiTech University

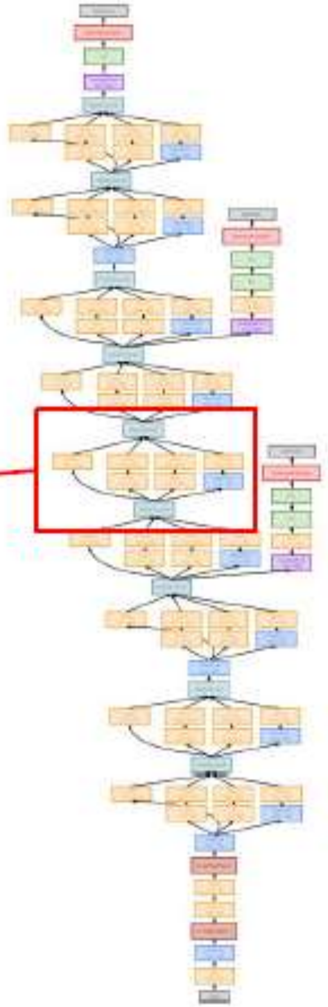
Case Study: GoogLeNet

[Szegedy et al., 2014]

"Inception module": design a good local network topology (network within a network) and then stack these modules on top of each other



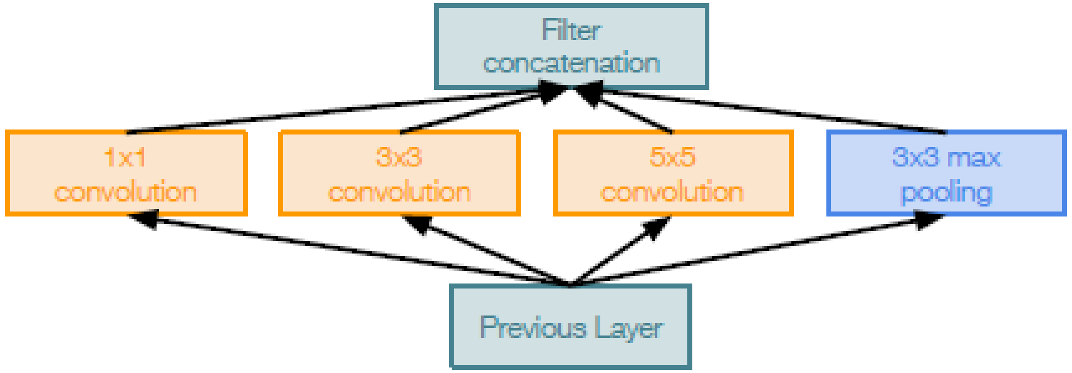
Inception module



GoogLeNet

上海科技大学
ShanghaiTech University

- Inception Module



Naive Inception module

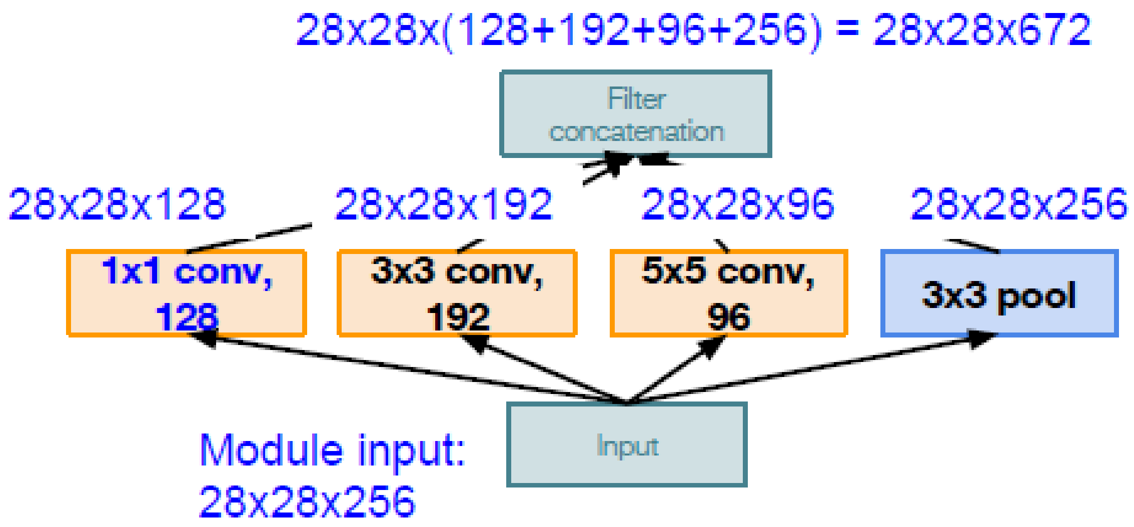
Apply parallel filter operations on the input from previous layer:

- Multiple receptive field sizes for convolution (1x1, 3x3, 5x5)
- Pooling operation (3x3)

Concatenate all filter outputs together depth-wise

GoogLeNet

- Inception Module



Naive Inception module

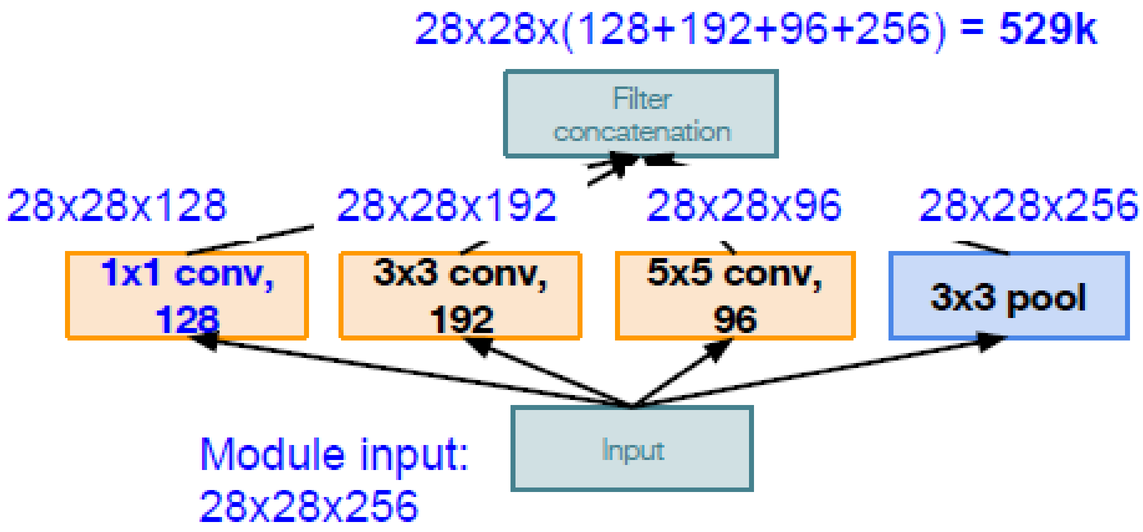
Conv Ops:
[1x1 conv, 128] 28x28x128x1x1x256
[3x3 conv, 192] 28x28x192x3x3x256
[5x5 conv, 96] 28x28x96x5x5x256
Total: 854M ops

Very expensive compute

Pooling layer also preserves feature depth, which means total depth after concatenation can only grow at every layer!

GoogLeNet

- Inception Module

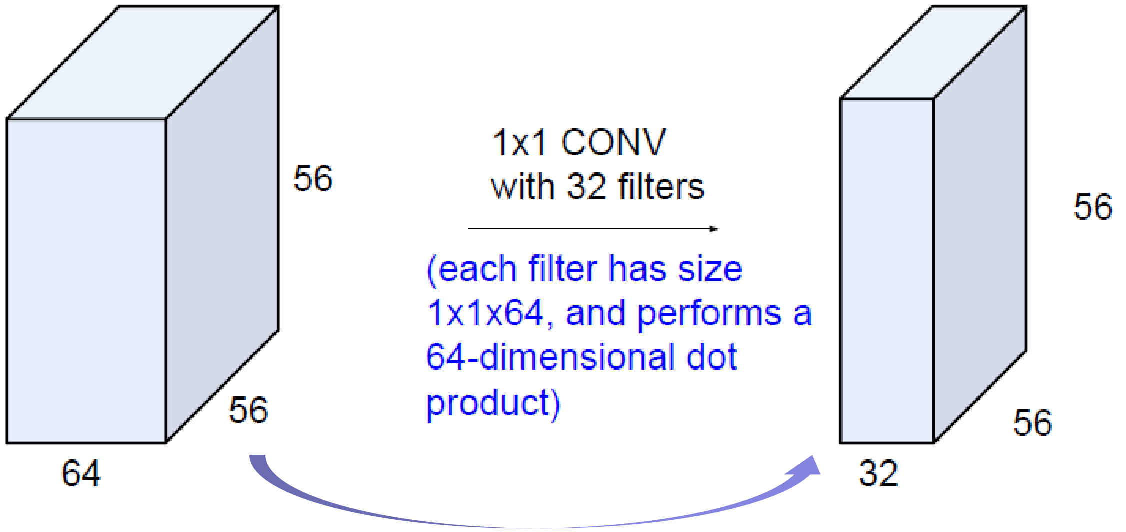


Naive Inception module

Solution: “bottleneck” layers that use 1x1 convolutions to reduce feature depth

GoogLeNet

- Bottleneck layer

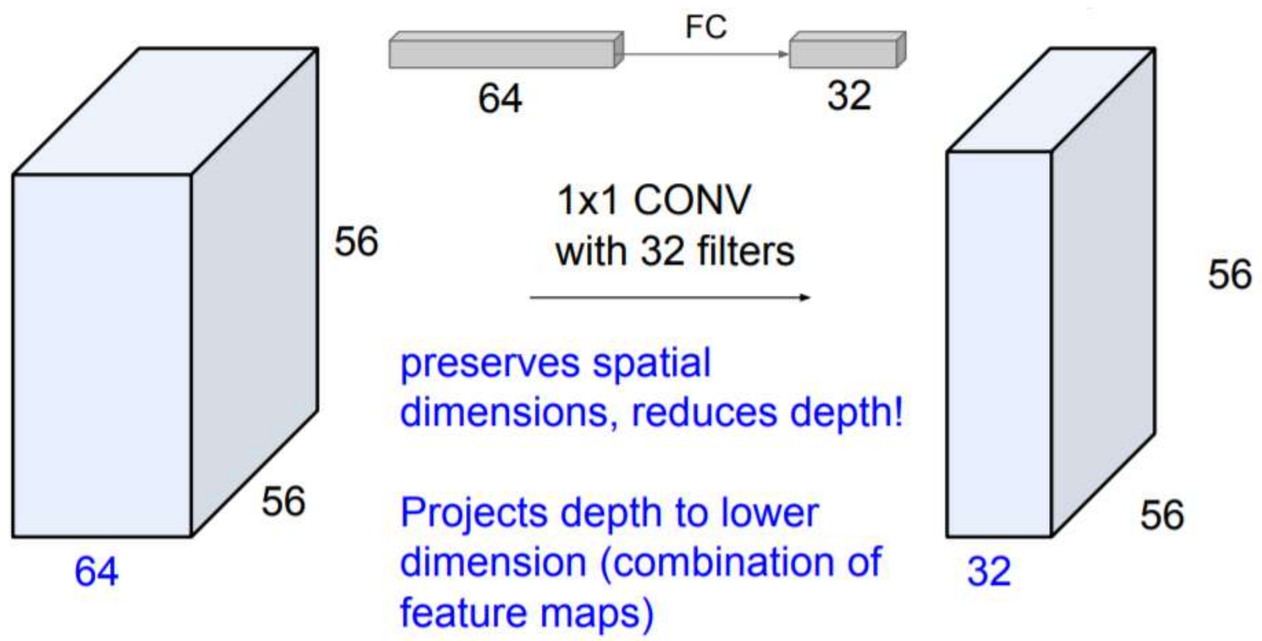


preserves spatial
dimensions, reduces depth!

Projects depth to lower
dimension (combination of
feature maps)

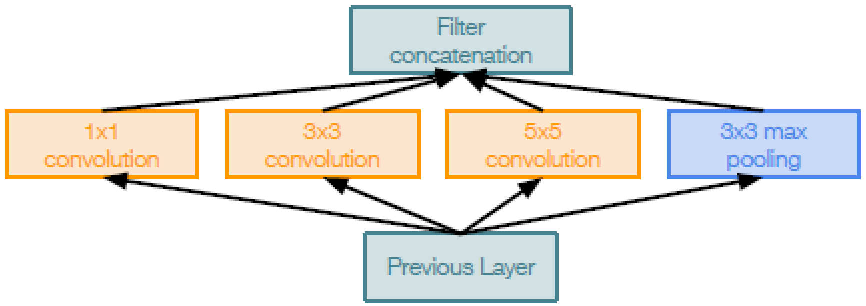
GoogLeNet

- 1x1 Convolutions
  - Alternatively, interpret it as applying the same FC layer on each input pixel



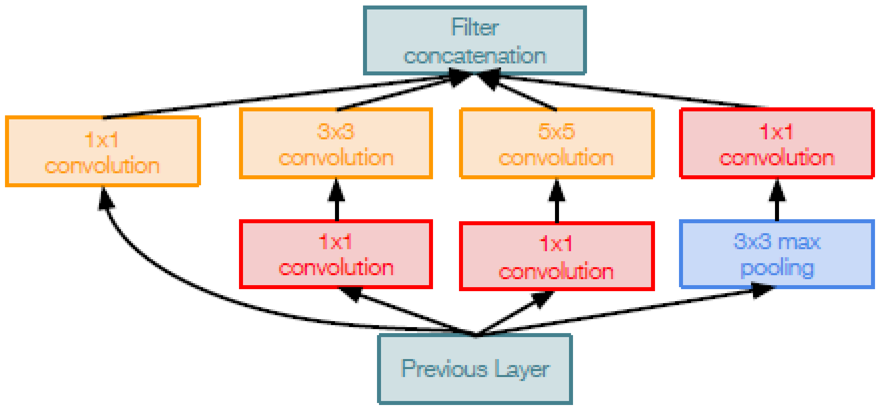
GoogLeNet

- Inception Module



Naive Inception module

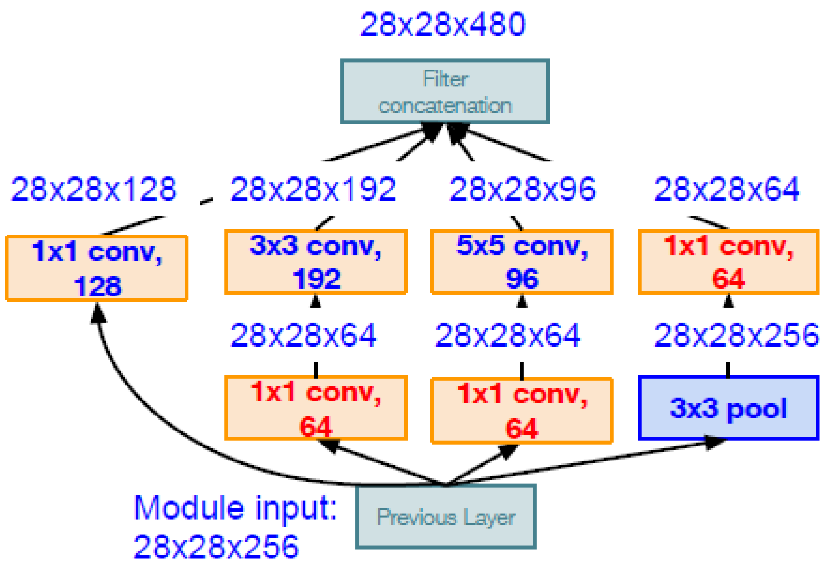
1x1 conv "bottleneck"
layers



Inception module with dimension reduction

GoogLeNet

- Inception Module



Inception module with dimension reduction

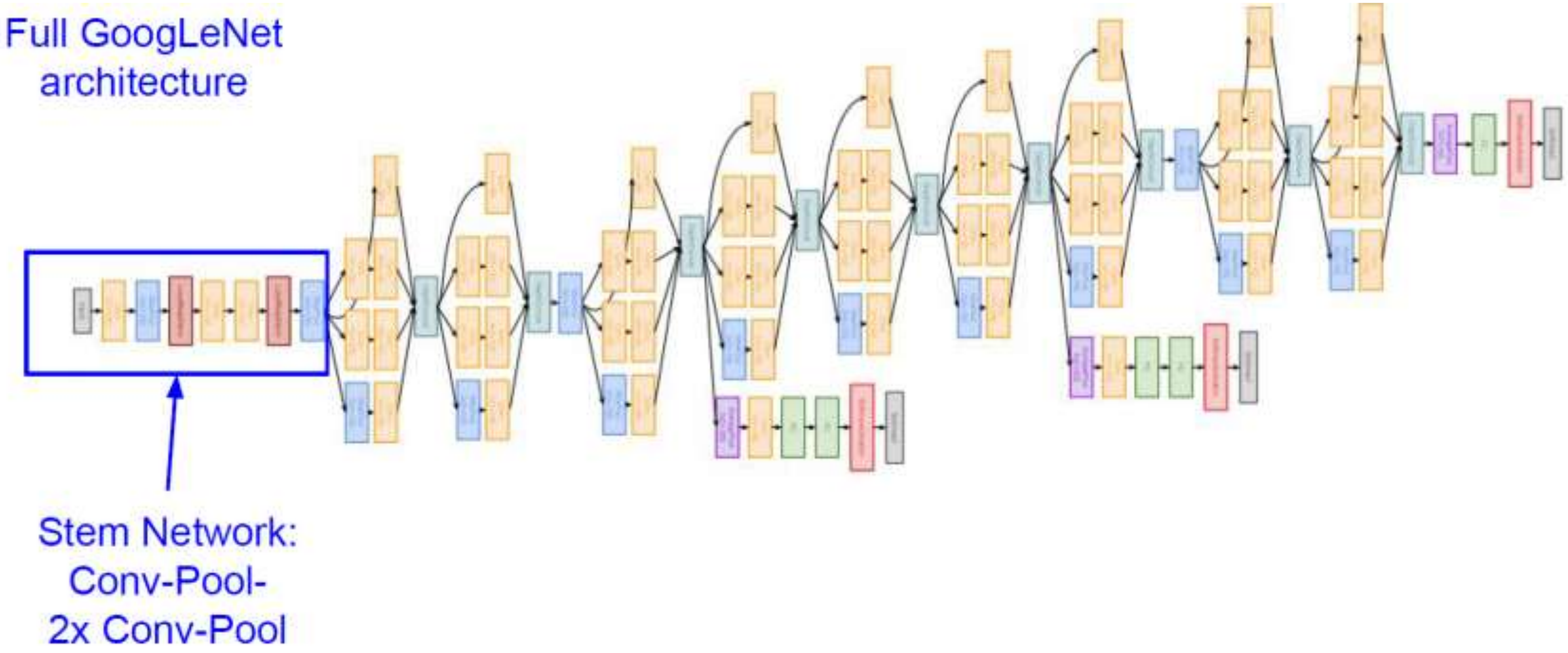
Conv Ops:
[1x1 conv, 64] 28x28x64x1x1x256
[1x1 conv, 64] 28x28x64x1x1x256
[1x1 conv, 128] 28x28x128x1x1x256
[3x3 conv, 192] 28x28x192x3x3x64
[5x5 conv, 96] 28x28x96x5x5x64
[1x1 conv, 64] 28x28x64x1x1x256
Total: 358M ops

Compared to 854M ops for naive version
Bottleneck can also reduce depth after pooling layer

GoogLeNet

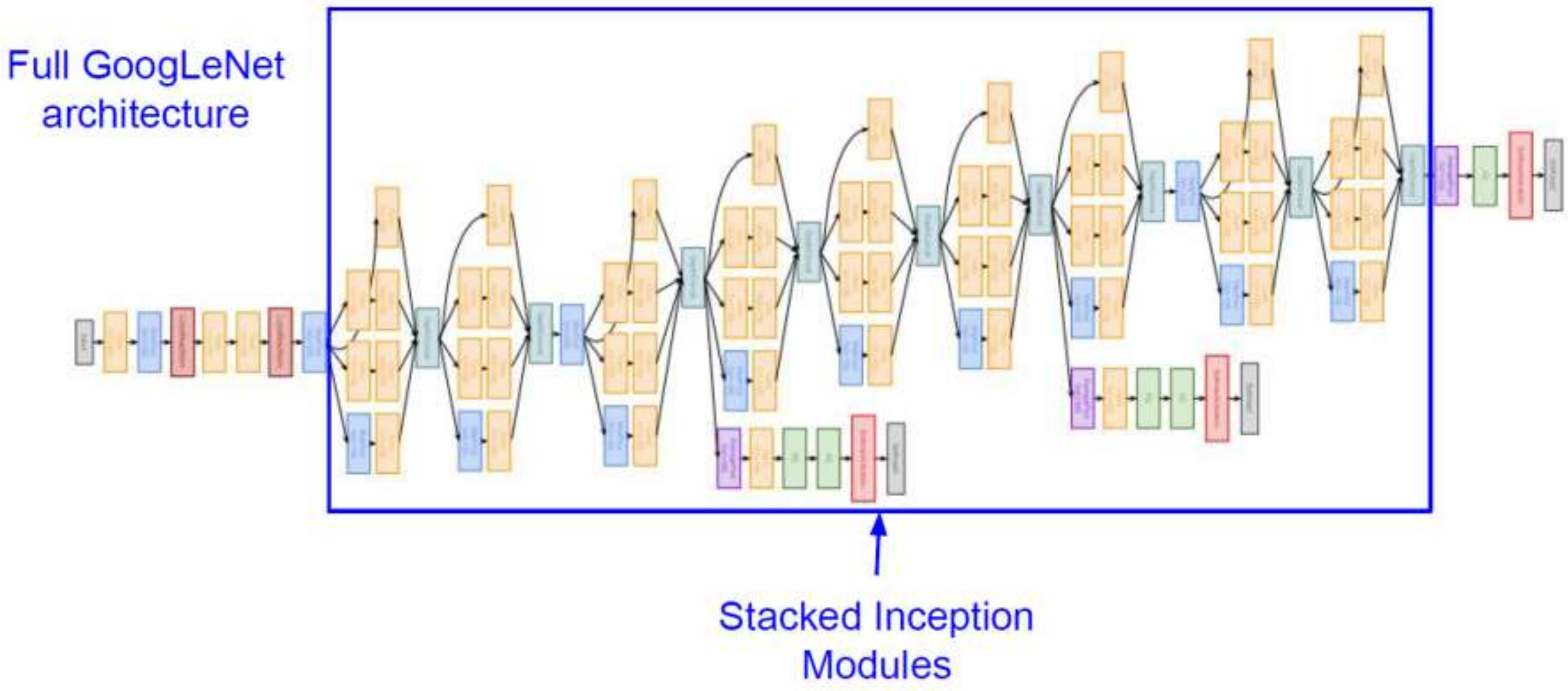
上海科技大学
ShanghaiTech University

- Overall network structure



GoogLeNet

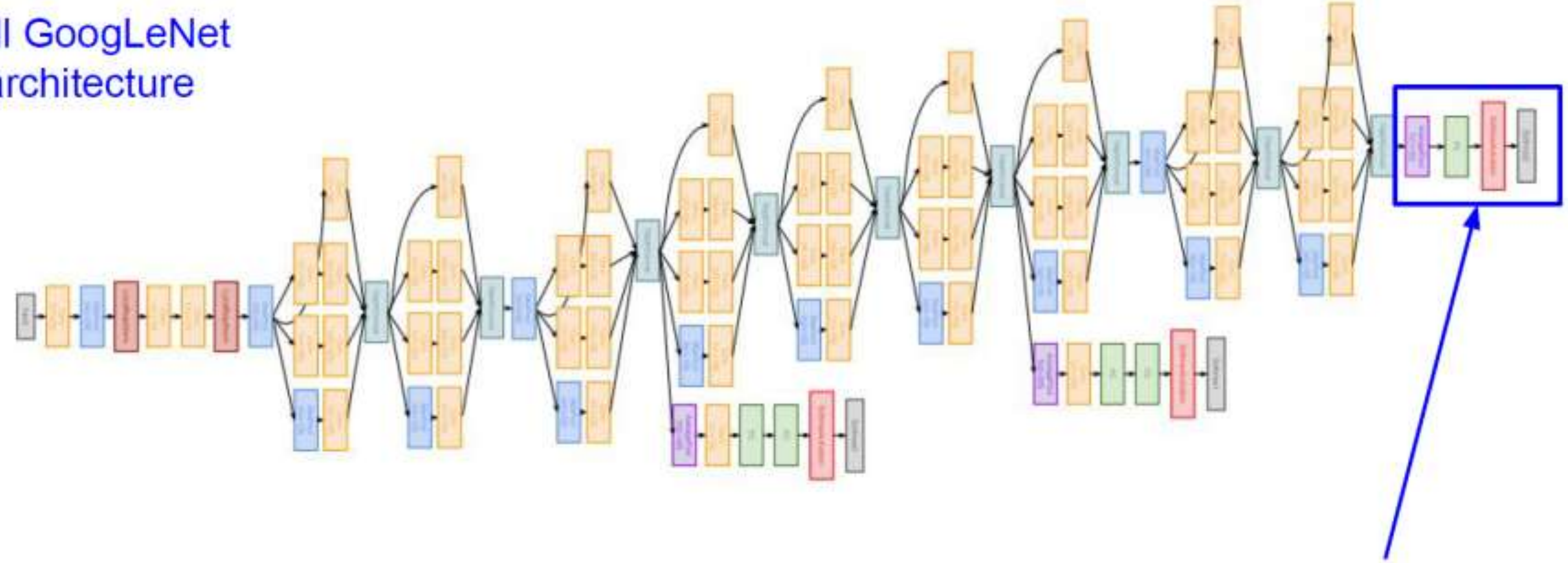
- Overall network structure



GoogLeNet

- Overall network structure

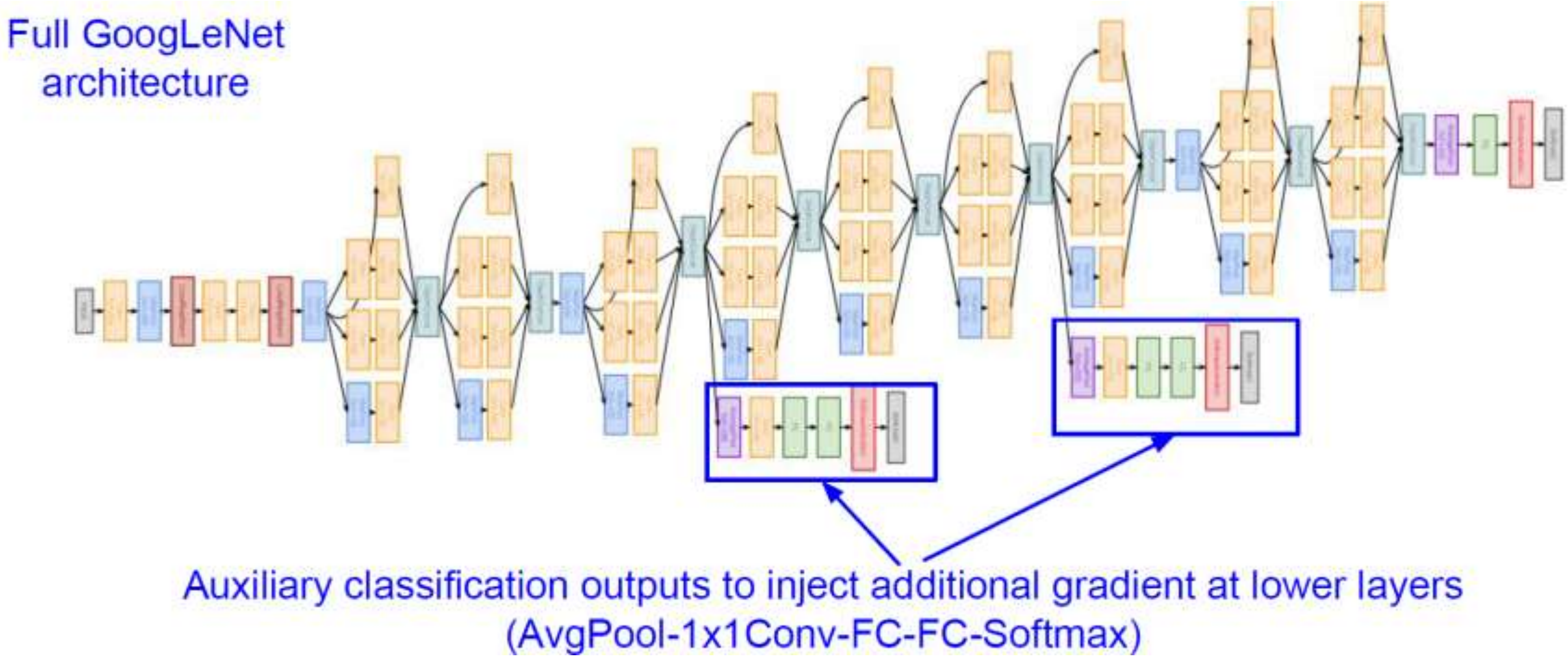
Full GoogLeNet architecture



Classifier output
(removed expensive FC layers!)

GoogLeNet

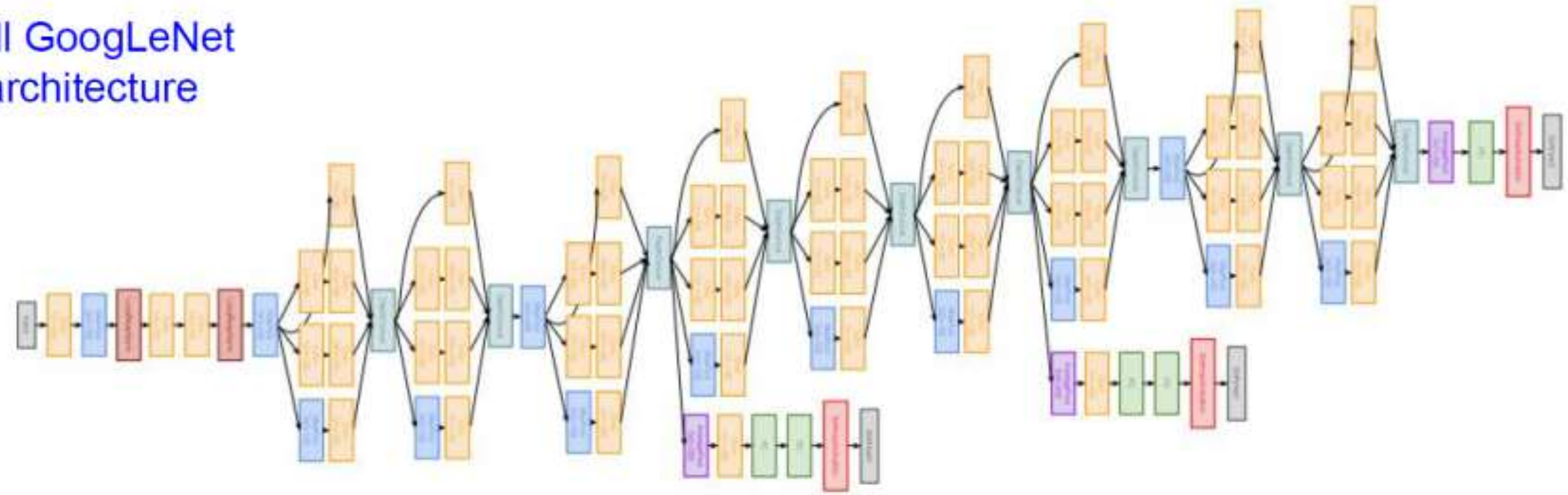
- Overall network structure



GoogLeNet

- Overall network structure

Full GoogLeNet architecture



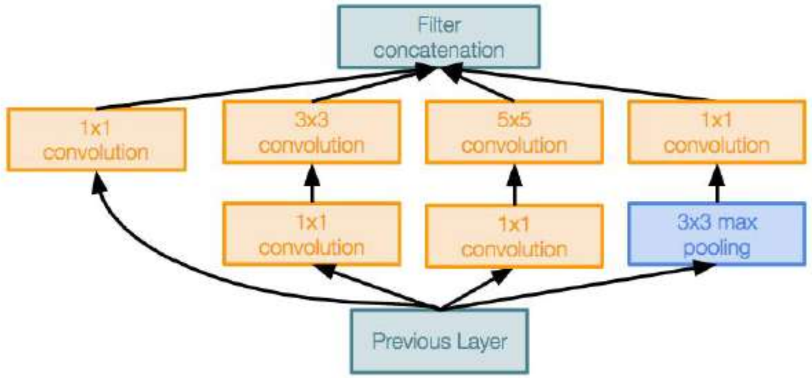
22 total layers with weights (including each parallel layer in an Inception module)

GoogLeNet

- Summary

Deeper networks, with computational efficiency

- 22 layers
- Efficient "Inception" module
- No FC layers
- 12x less params than AlexNet
- ILSVRC'14 classification winner (6.7% top 5 error)



Inception module

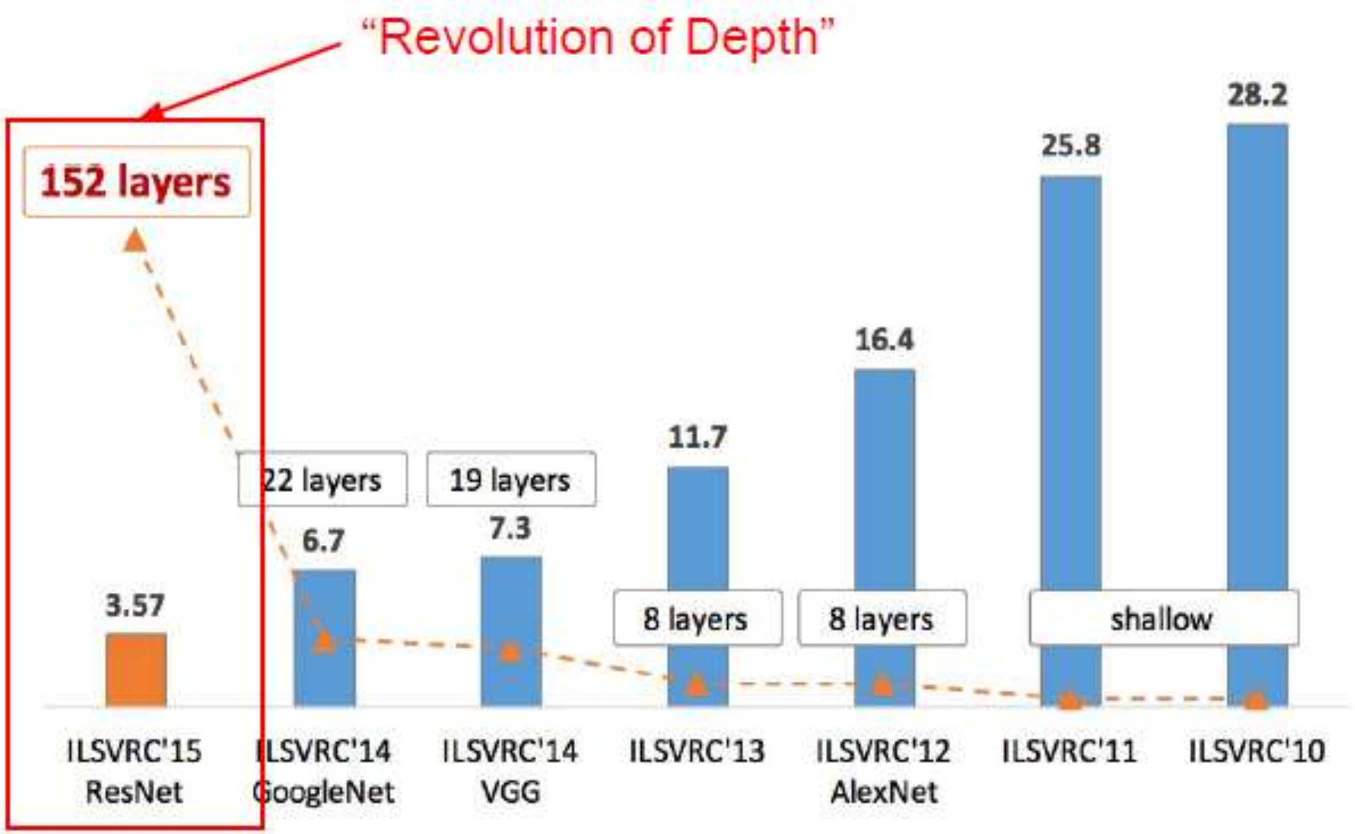
Outline

- CNN architectures
  - Sequential structure: LeNet/AlexNet/VGGNet
  - Parallel branches: GoogLeNet
  - Residual structure: ResNet/DenseNet
  - Network Architecture Search

Acknowledgement: Zemel et al's CSC411 and Feifei Li et al's cs231n notes

ImageNet (ILSVRC)

上海科技大学 ShanghaiTech University



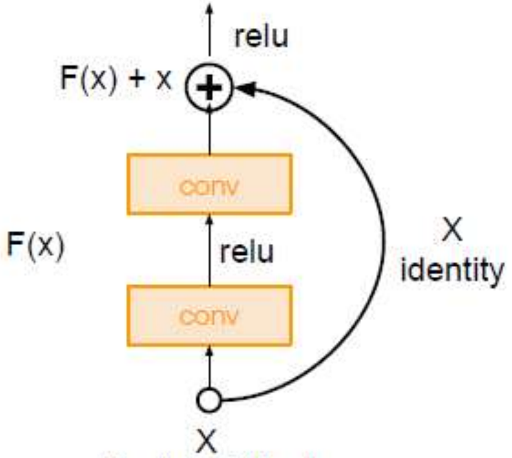
ResNet

Case Study: ResNet

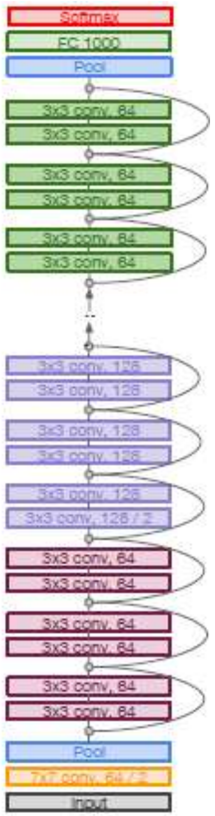
[He et al., 2015]

Very deep networks using residual connections

- 152-layer model for ImageNet
- ILSVRC'15 classification winner (3.57% top 5 error)
- Swept all classification and detection competitions in ILSVRC'15 and COCO'15!

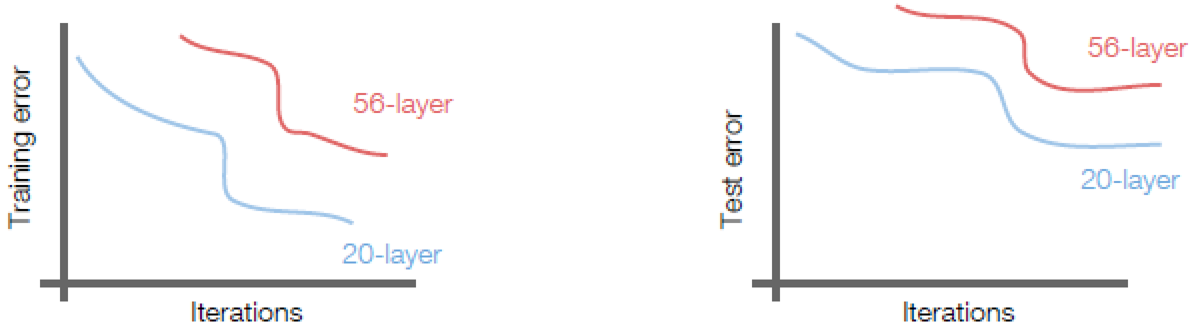


Residual block



ResNet

- What happens when stacking deeper plain conv layers?



56-layer model performs worse on both training and test error
-> The deeper model performs worse, but it's not caused by overfitting!

ResNet

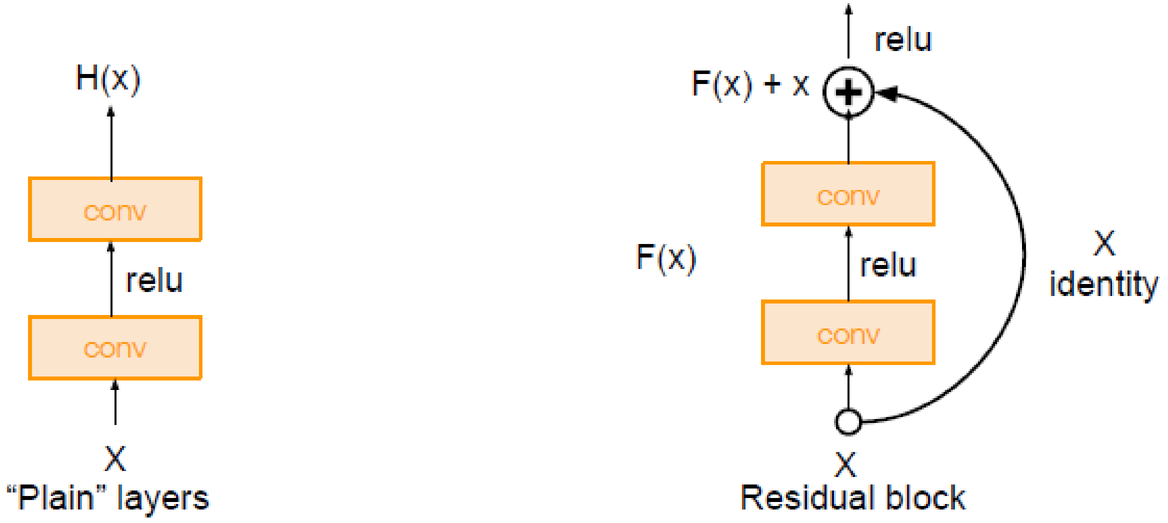
- Hypothesis:
  - The problem is an optimization problem, deeper models are harder to optimize

The deeper model should be able to perform at least as well as the shallower model.

A solution by construction is copying the learned layers from the shallower model and setting additional layers to identity mapping.

ResNet

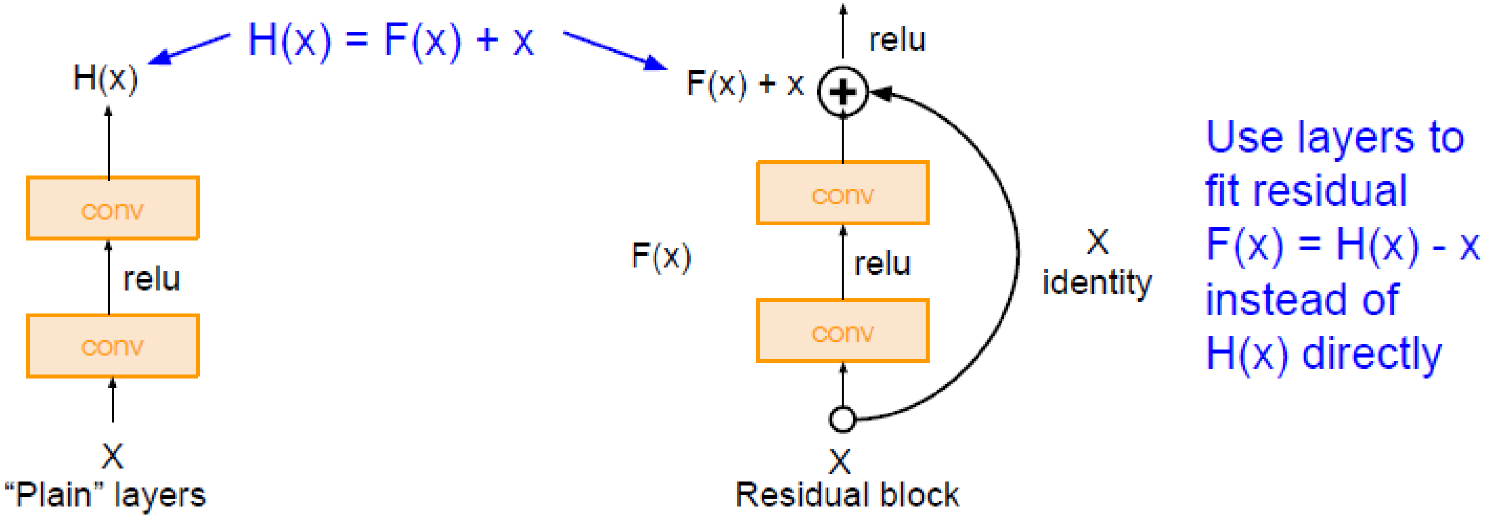
- Solution:
  - Use network layers to fit a residual mapping



He et al "Deep Residual Learning for Image Recognition", CVPR 2016

ResNet

- Solution:
  - Use network layers to fit a residual mapping



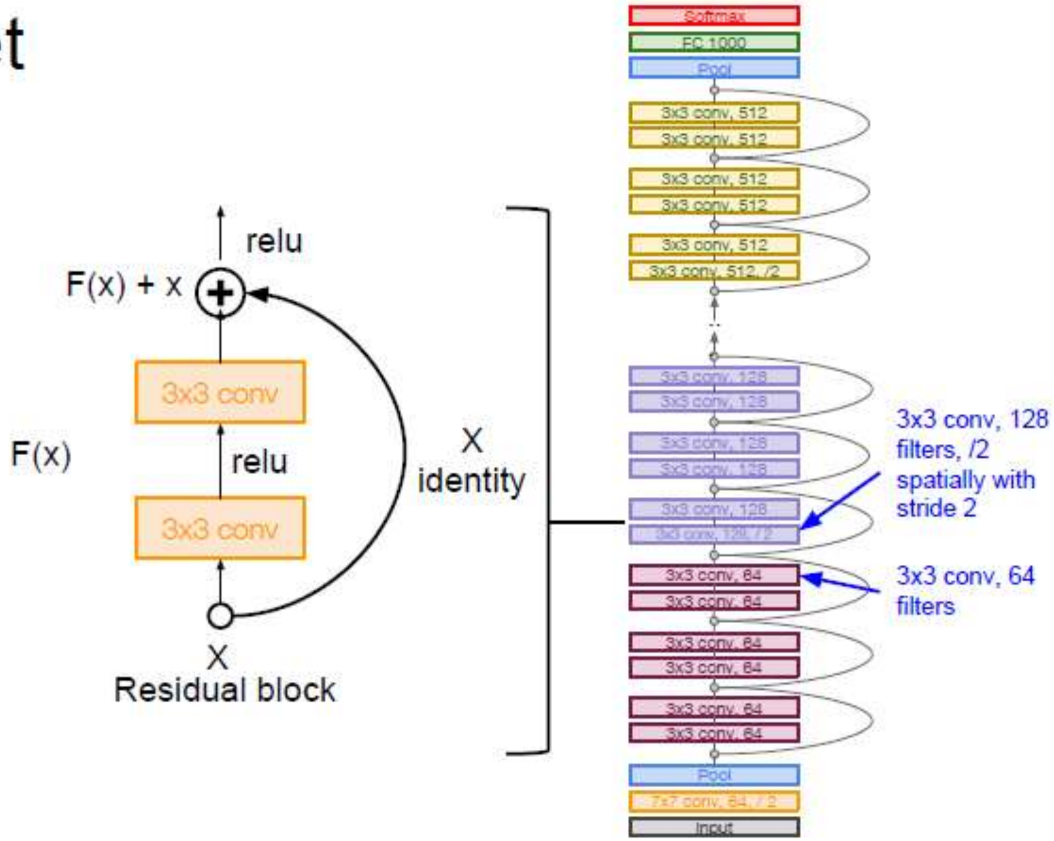
ResNet

Case Study: ResNet

[He et al., 2015]

Full ResNet architecture:

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (/2 in each dimension)



ResNet

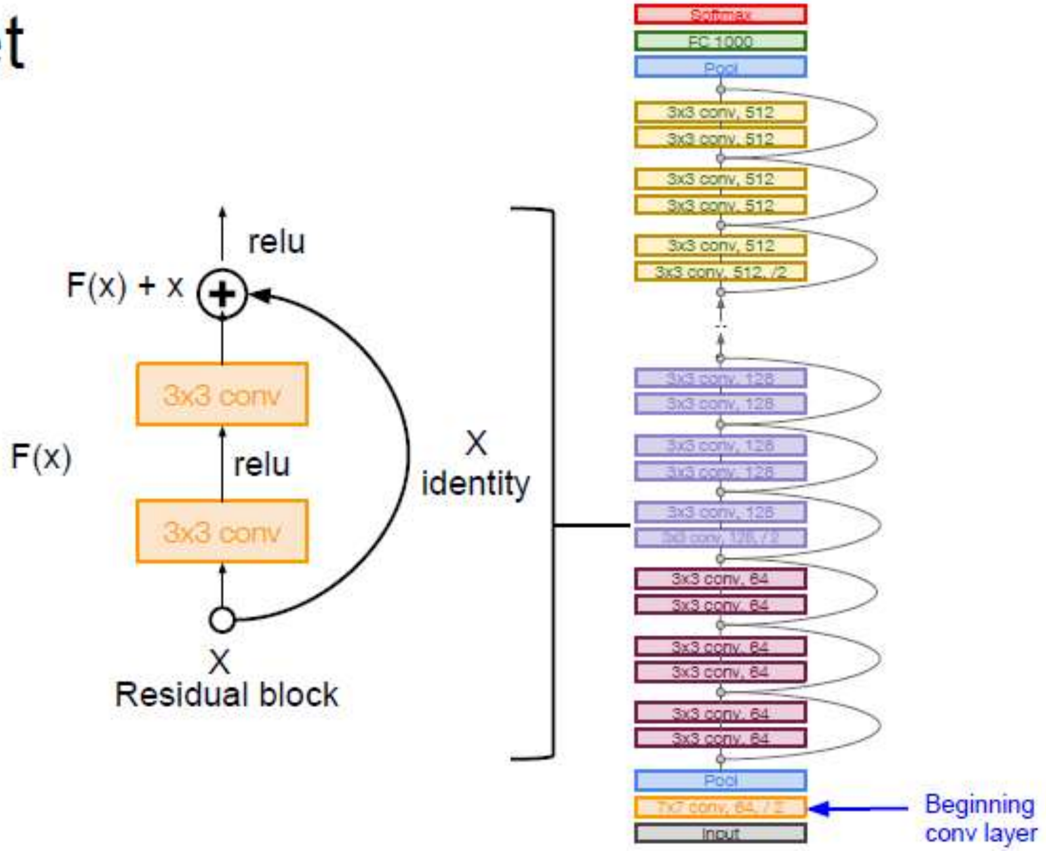
上海科技大学 ShanghaiTech University

Case Study: ResNet

[He et al., 2015]

Full ResNet architecture:

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (/2 in each dimension)
- Additional conv layer at the beginning



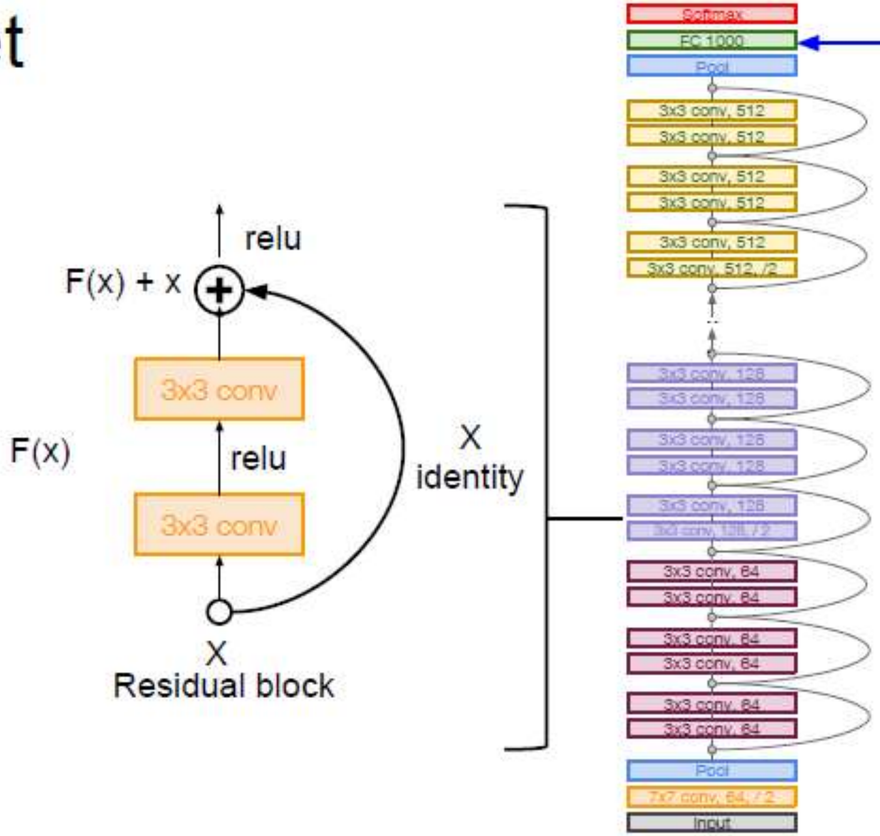
ResNet

上海科技大学
ShanghaiTech University

Case Study: ResNet

[He et al., 2015]

Full ResNet architecture:
- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (/2 in each dimension)
- Additional conv layer at the beginning
- No FC layers at the end (only FC 1000 to output classes)



No FC layers besides FC 1000 to output classes

Global average pooling layer after last conv layer

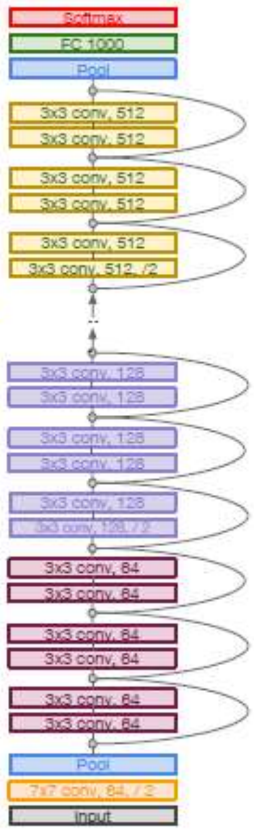
ResNet

上海科技大学
ShanghaiTech University

Case Study: ResNet

[He et al., 2015]

Total depths of 34, 50, 101, or 152 layers for ImageNet



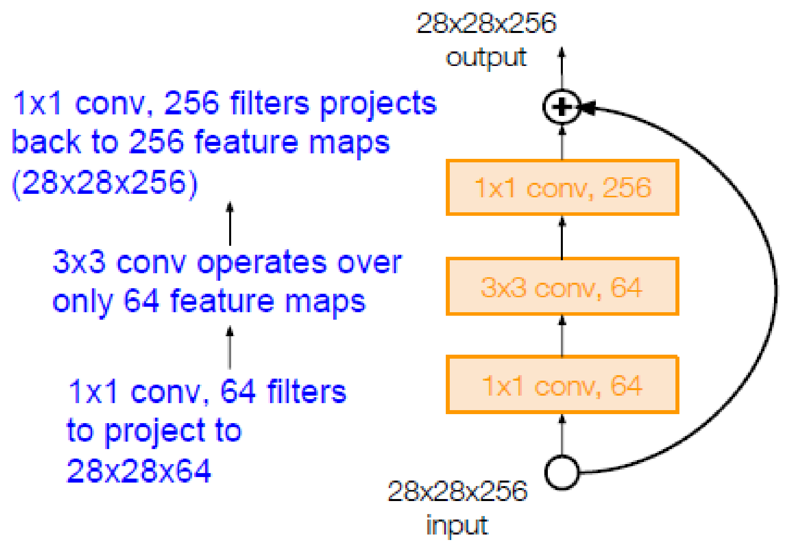
ResNet

上海科技大学 ShanghaiTech University

Case Study: ResNet

[He et al., 2015]

For deeper networks (ResNet-50+), use "bottleneck" layer to improve efficiency (similar to GoogLeNet)



ResNet

上海科技大学 ShanghaiTech University

- Training details

Training ResNet in practice:

- Batch Normalization after every CONV layer
- Xavier/2 initialization from He et al.
- SGD + Momentum (0.9)
- Learning rate: 0.1, divided by 10 when validation error plateaus
- Mini-batch size 256
- Weight decay of 1e-5
- No dropout used

ResNet

上海科技大学
ShanghaiTech University

- Results

Experimental Results
- Able to train very deep networks without degrading (152 layers on ImageNet, 1202 on Cifar)
- Deeper networks now achieve lowing training error as expected
- Swept 1st place in all ILSVRC and COCO 2015 competitions

MSRA @ ILSVRC & COCO 2015 Competitions

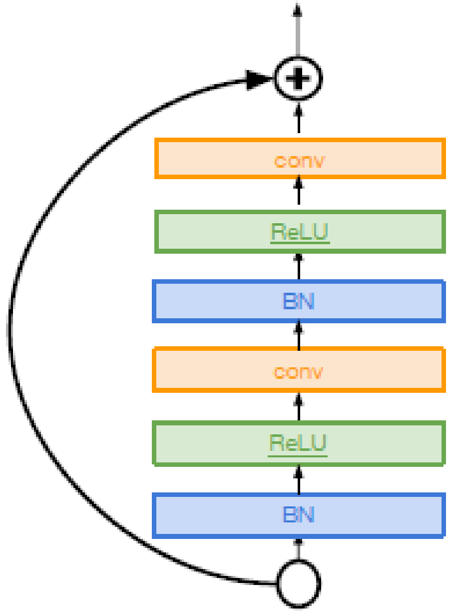
- **1st places in all five main tracks**
  - ImageNet Classification: *"Ultra-deep"* (quote Yann) 152-layer nets
  - ImageNet Detection: 16% better than 2nd
  - ImageNet Localization: 27% better than 2nd
  - COCO Detection: 11% better than 2nd
  - COCO Segmentation: 12% better than 2nd

ILSVRC 2015 classification winner (3.6% top 5 error) -- better than "human performance"! (Russakovsky 2014)

Other: Identity Mappings in ResNet

上海科技大学 ShanghaiTech University

- Improved ResNet block design from creators of ResNet
- Creates a more direct path for propagating information throughout network (moves activation to residual mapping pathway)
- Gives better performance

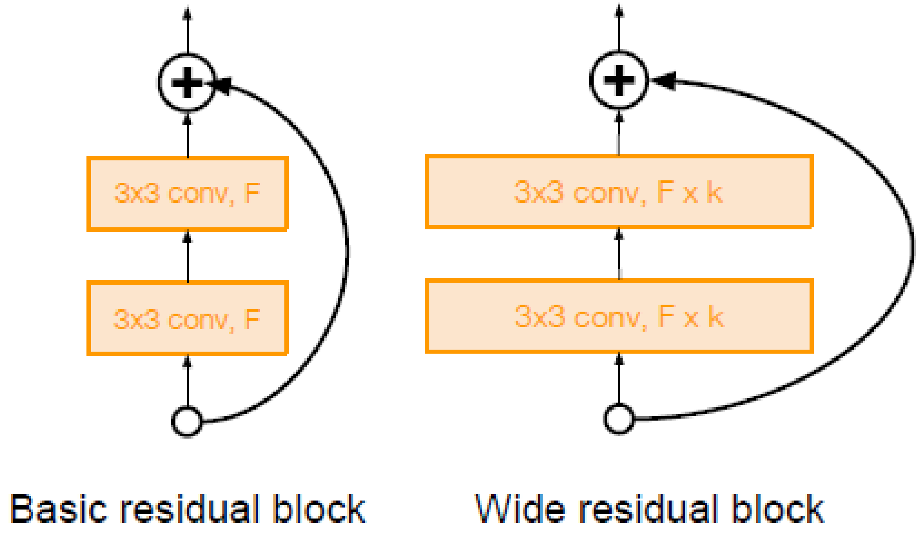


Other: Wide ResNets

Wide Residual Networks

[Zagoruyko et al. 2016]

- Argues that residuals are the important factor, not depth
- User wider residual blocks (F x k filters instead of F filters in each layer)
- 50-layer wide ResNet outperforms 152-layer original ResNet
- Increasing width instead of depth more computationally efficient (parallelizable)

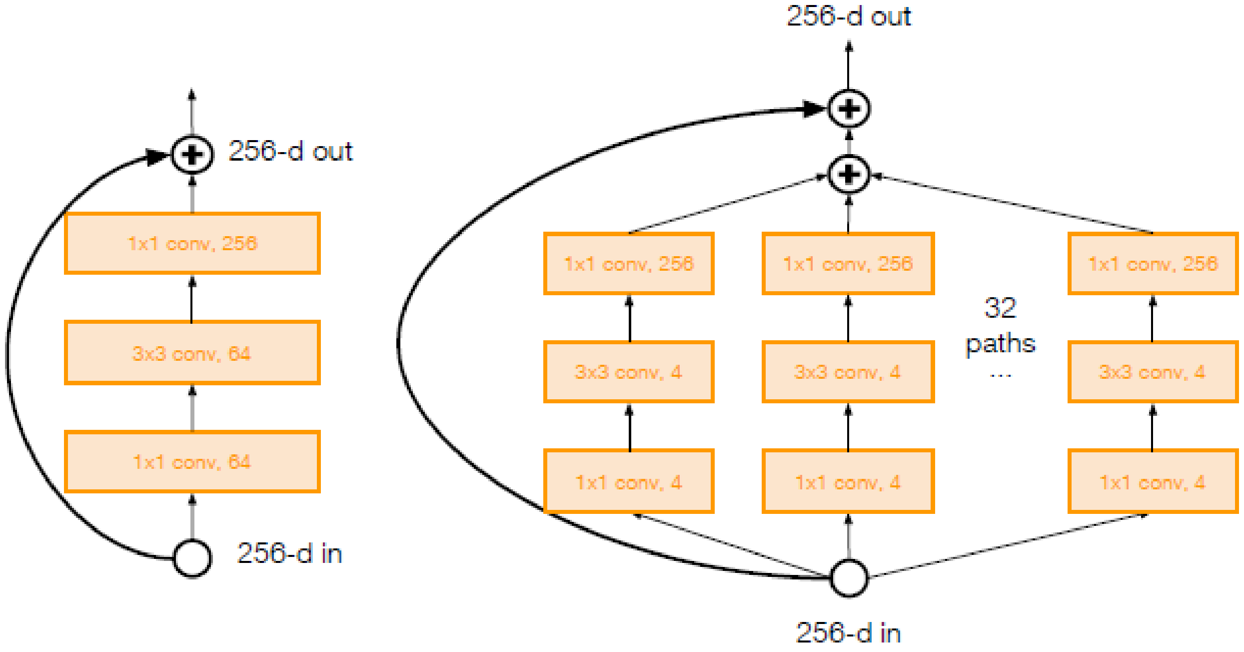


Other: ResNeXt

Aggregated Residual Transformations for Deep Neural Networks (ResNeXt)

[Xie et al. 2016]

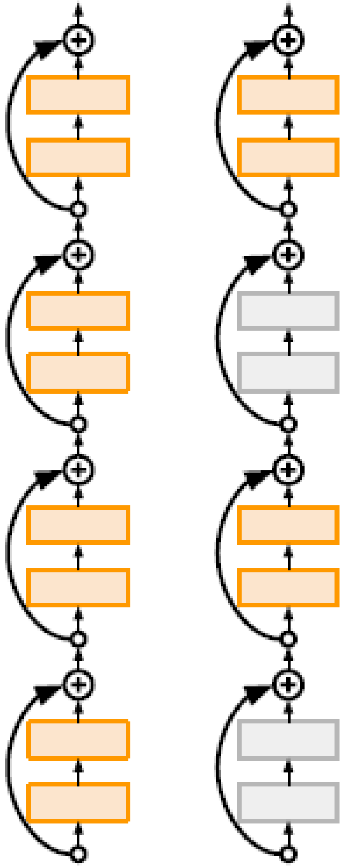
- Also from creators of ResNet
- Increases width of residual block through multiple parallel pathways ("cardinality")
- Parallel pathways similar in spirit to Inception module



Deep Networks with Stochastic Depth

[Huang et al. 2016]

- Motivation: reduce vanishing gradients and training time through short networks during training
- Randomly drop a subset of layers during each training pass
- Bypass with identity function
- Use full deep network at test time

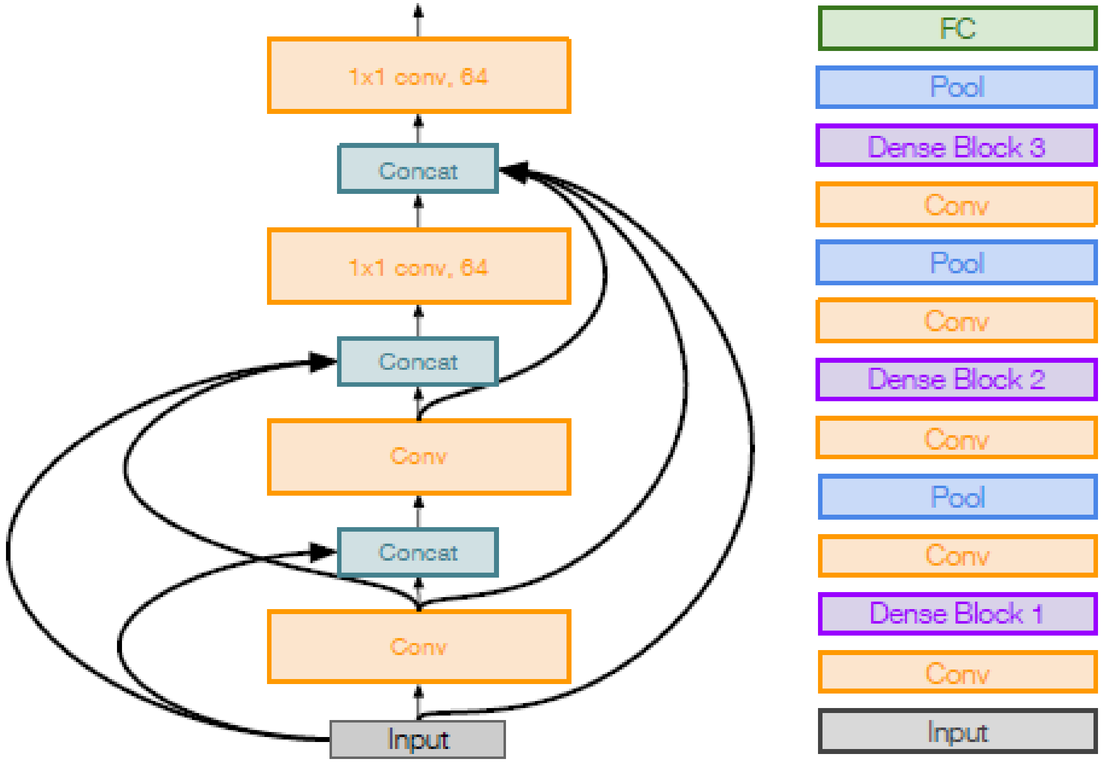


DenseNet

Densely Connected Convolutional Networks

[Huang et al. 2017]

- Dense blocks where each layer is connected to every other layer in feedforward fashion
- Alleviates vanishing gradient, strengthens feature propagation, encourages feature reuse

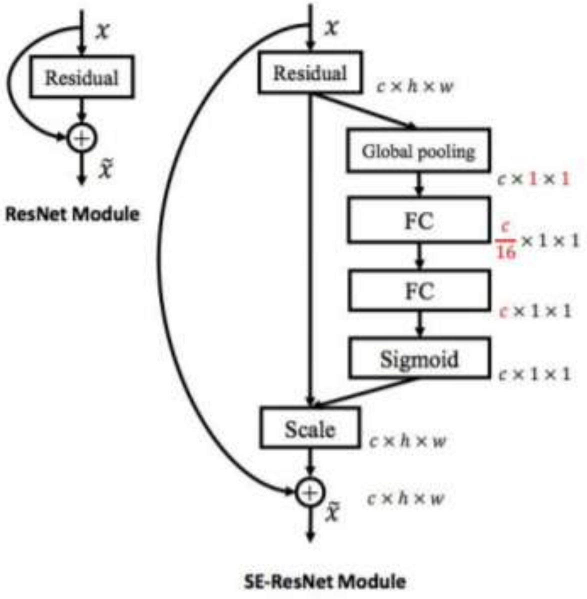


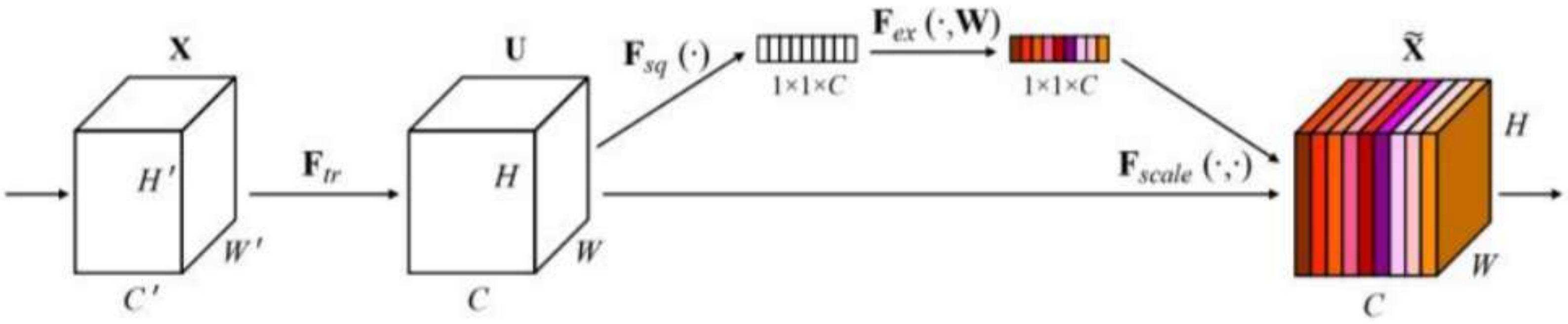
Dense Block

Squeeze-and-Excitation Networks (SENet)

[Hu et al. 2017]

- Add a “feature recalibration” module that learns to adaptively reweight feature maps
- Global information (global avg. pooling layer) + 2 FC layers used to determine feature map weights
- ILSVRC’17 classification winner (using ResNeXt-152 as a base architecture)

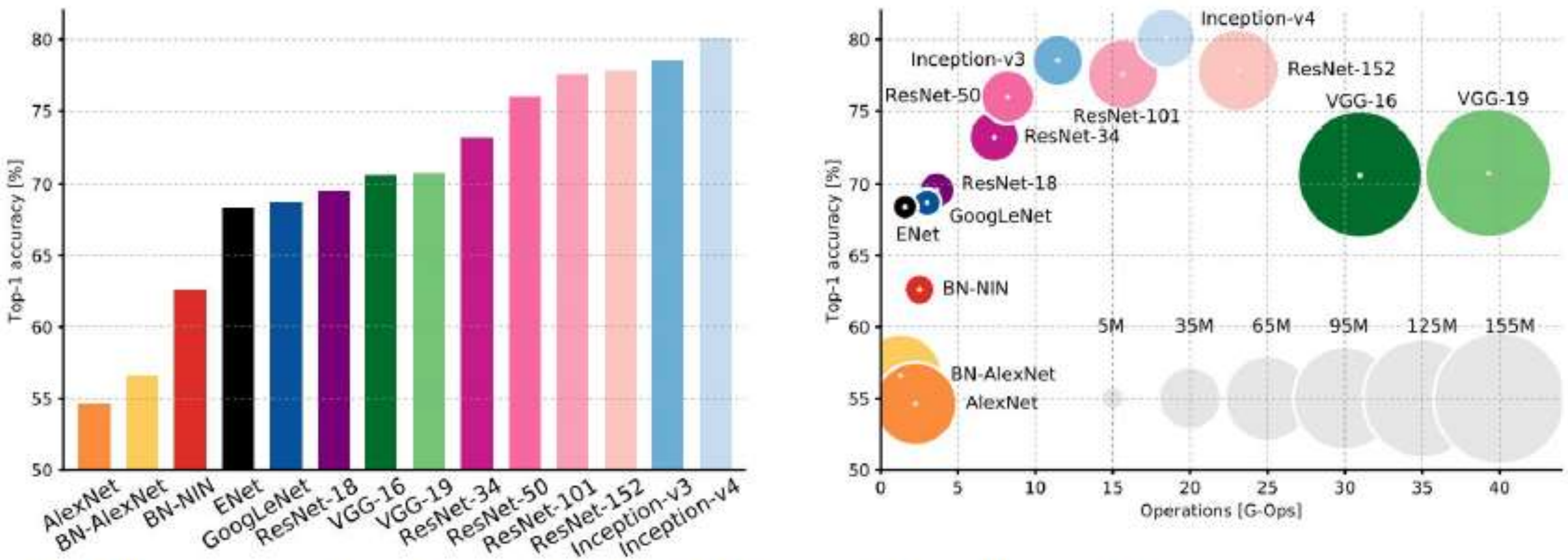




Model complexity

上海科技大学
ShanghaiTech University

Comparing complexity...

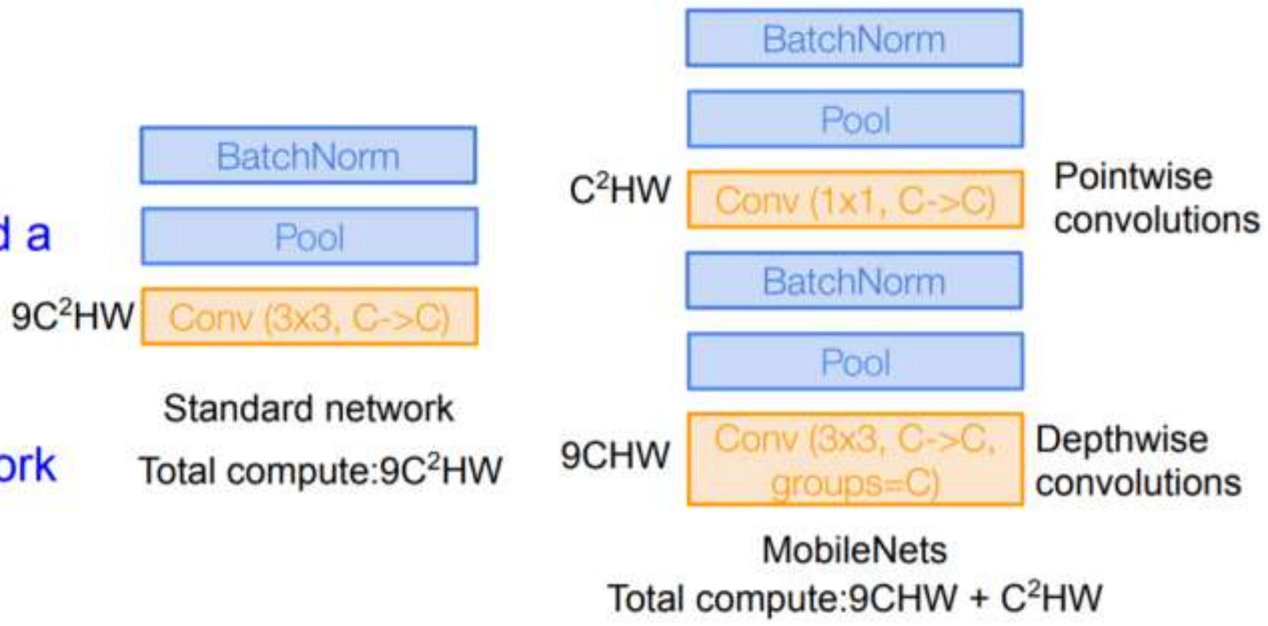


An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Efficient networks

- MobileNets: Efficient Convolutional Neural Networks for Mobile Applications [Howard et al. 2017]

- Depthwise separable convolutions replace standard convolutions by factorizing them into a depthwise convolution and a 1x1 convolution
- Much more efficient, with little loss in accuracy
- Follow-up MobileNetV2 work in 2018 (Sandler et al.)
- ShuffleNet: Zhang et al, CVPR 2018



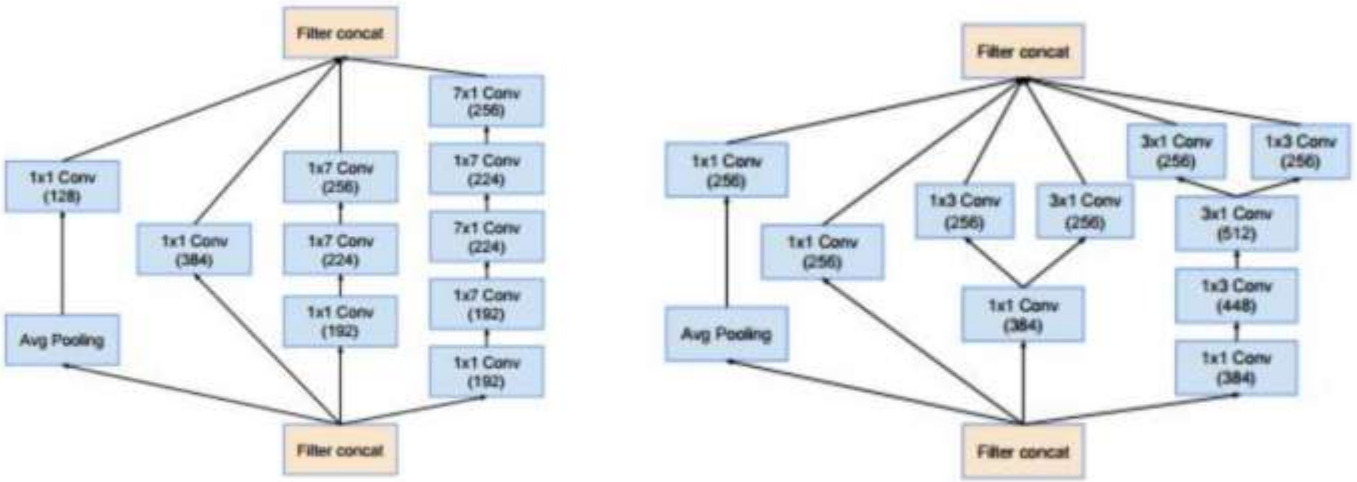
Outline

- CNN architectures
  - Sequential structure: LeNet/AlexNet/VGGNet
  - Parallel branches: GoogLeNet
  - Residual structure: ResNet/DenseNet
  - Network Architecture Search

Acknowledgement: Zemel et al's CSC411 and Feifei Li et al's cs231n notes

Network Architecture

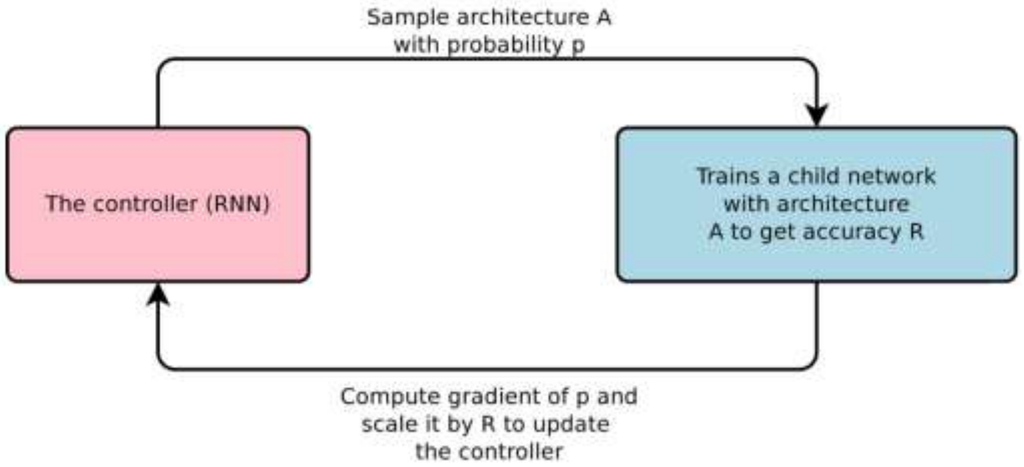
- Problems with network architecture
  - Designing NA is hard
  - Lots of human efforts go into tuning them
  - Not a lot of intuition into how to design them well
  - Can we learn good architectures automatically?



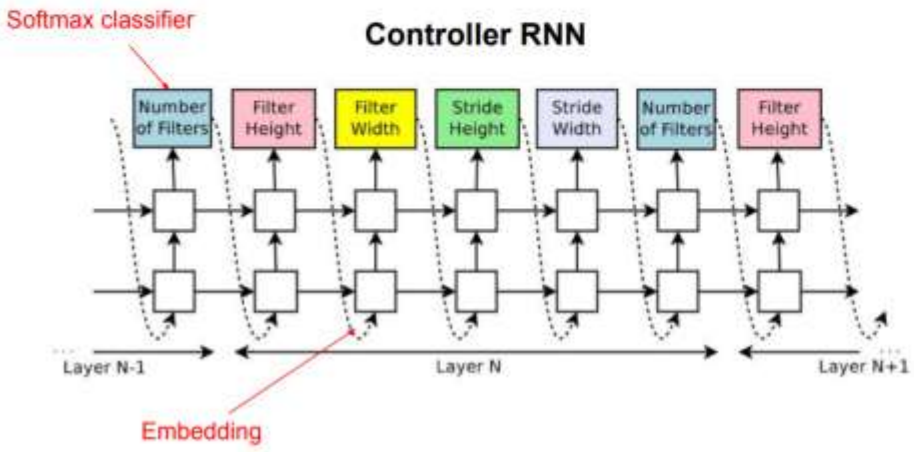
Two layers from the famous Inception V4 computer vision model.
Szegedy et al, 2017

Network Architecture

■ Neural architecture search (Zoph and Le, ICLR 2016)



- "Controller" network that learns to design a good network architecture (output a string corresponding to network design)
- Iterate:
  1) Sample an architecture from search space
  2) Train the architecture to get a "reward" R corresponding to accuracy
  3) Compute gradient of sample probability, and scale by R to perform controller parameter update (i.e. increase likelihood of good architecture being sampled, decrease likelihood of bad architecture)



Network structure summary

上海科技大学 ShanghaiTech University

- AlexNet showed that you can use CNNs to train Computer Vision models.
- ZFNet, VGG shows that bigger networks work better
- GoogLeNet is one of the first to focus on efficiency using 1x1 bottleneck convolutions and global avg pool instead of FC layers
- ResNet showed us how to train extremely deep networks
  - Limited only by GPU & memory!
  - Showed diminishing returns as networks got bigger
- After ResNet: CNNs were *better than the human metric* and focus shifted to Efficient networks:
  - Lots of tiny networks aimed at mobile devices: MobileNet, ShuffleNet
- Neural Architecture Search can now automate architecture design

More on classification

上海科技大学
ShanghaiTech University

- Is image classification a solved problem?
  - "(Super-)Human level" performance on some benchmarks
    - Face identification
    - ImageNet 1000 classes
- But compared to human vision…
  - Limitations in learning
    - We can learn new classes using one or two examples
    - We can also handle label noises
    - We can generalize to unfamiliar scenes
  - Limitation in prediction
    - We can also predict the uncertainty
    - We can easily handle adversarial examples
    - We are much more efficient in power consumption

CNN Applications in Dense Prediction

上海科技大学
ShanghaiTech University

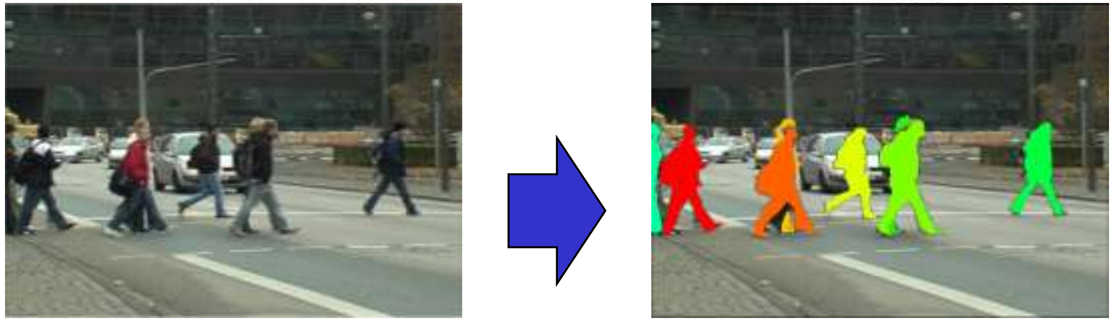
- What is semantic segmentation?
- Network architecture for semantic segmentation
  - Main idea for dense prediction
  - Fully convolutional network
  - Upsampling operators
  - Multiscale context modeling
- Network training losses

Acknowledgement: Feifei Li et al's cs231n notes

Review

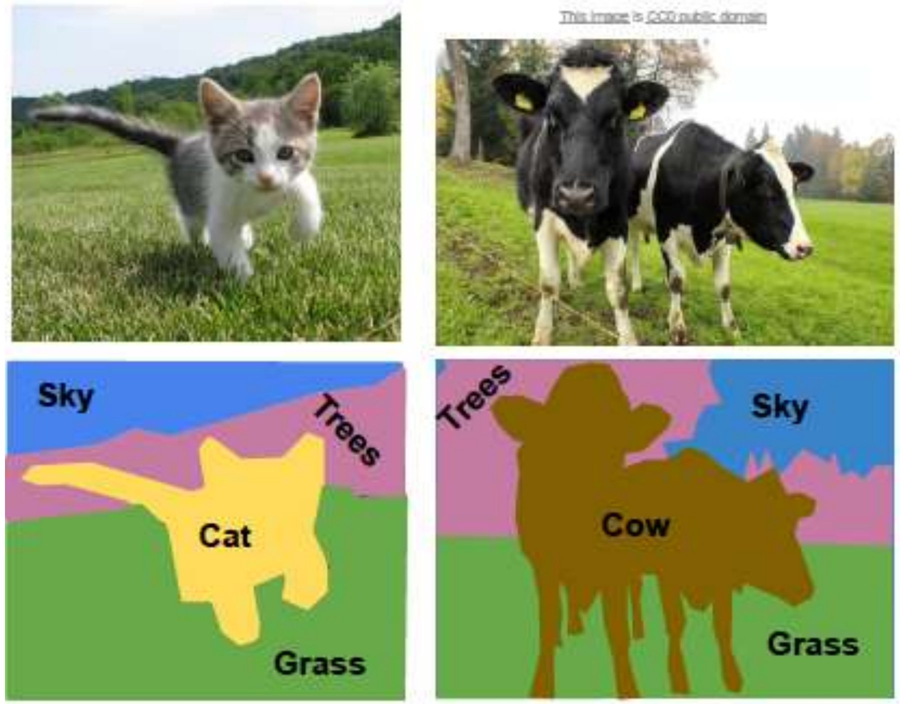
- In general, our goal is to learn a mapping from a signal to a ‘semantically meaningful’ representation.
  - Output can have many different forms:

red panda (*Ailurus fulgens*)



Semantic Segmentation

- Problem setup
  - Label each pixel in the image with an object category label
  - Do not differentiate object instances



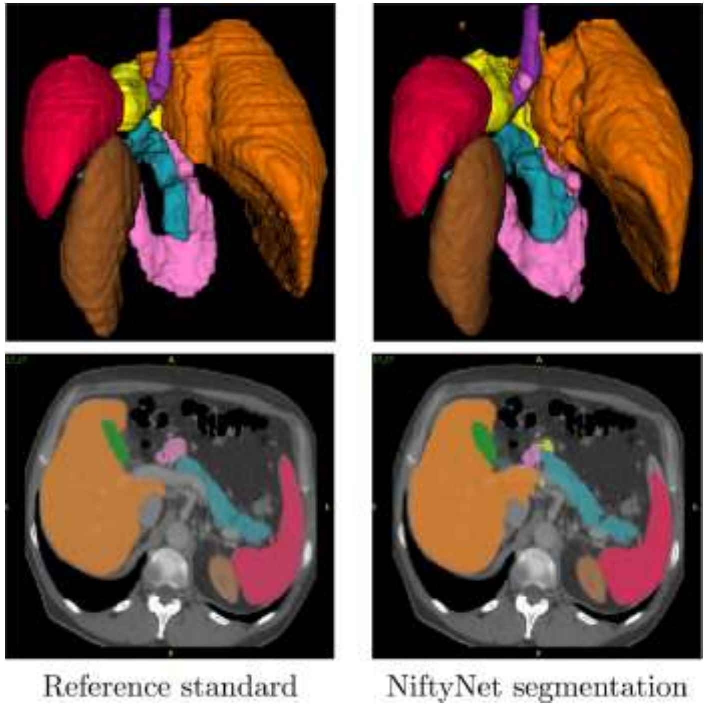
Key to many applications

- Autonomous robots and cars

- Safety and security

- Medical analysis and health

Multi-organ abdominal
CT segmentation

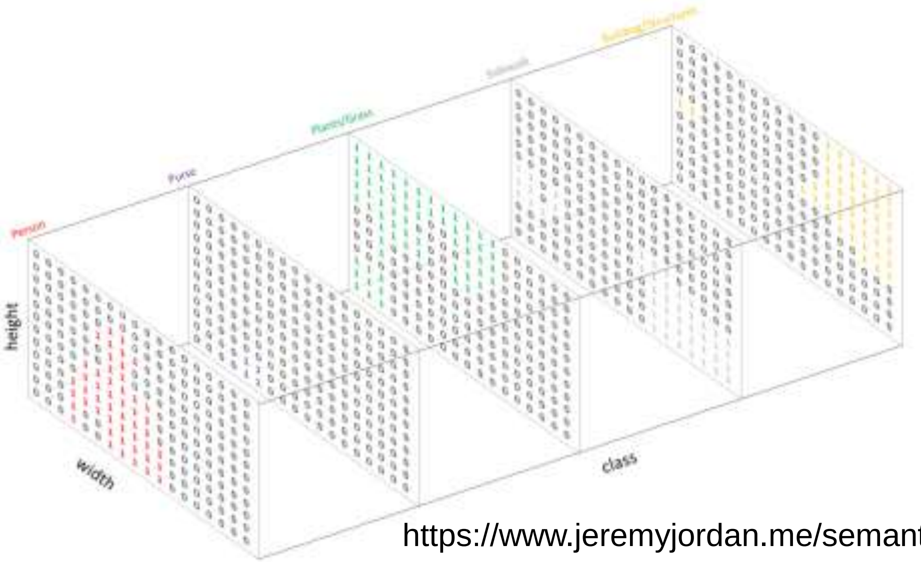


- etc…

Semantic Segmentation

- Problem formulation
  - Pixel-wise object classification task

  - One-hot encoding



https://www.jeremyjordan.me/semantic-segmentation/

Why this is challenging?

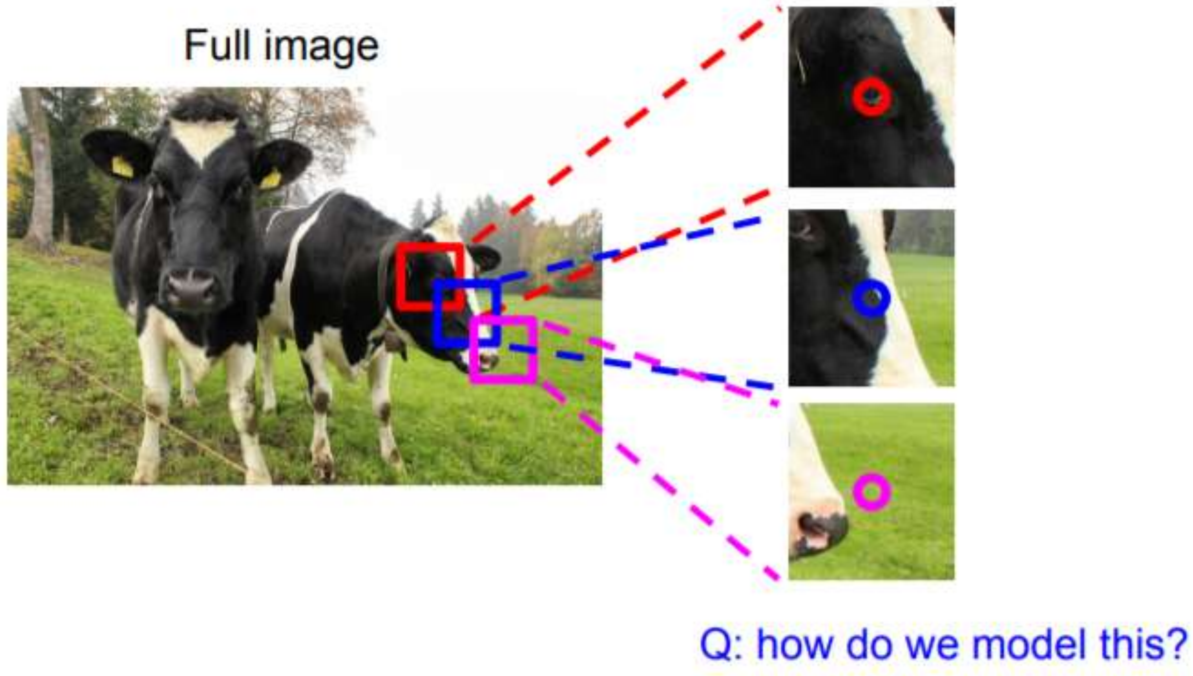
- A naïve approach

Impossible to classify without context

Q: how do we include context?

Why this is challenging?

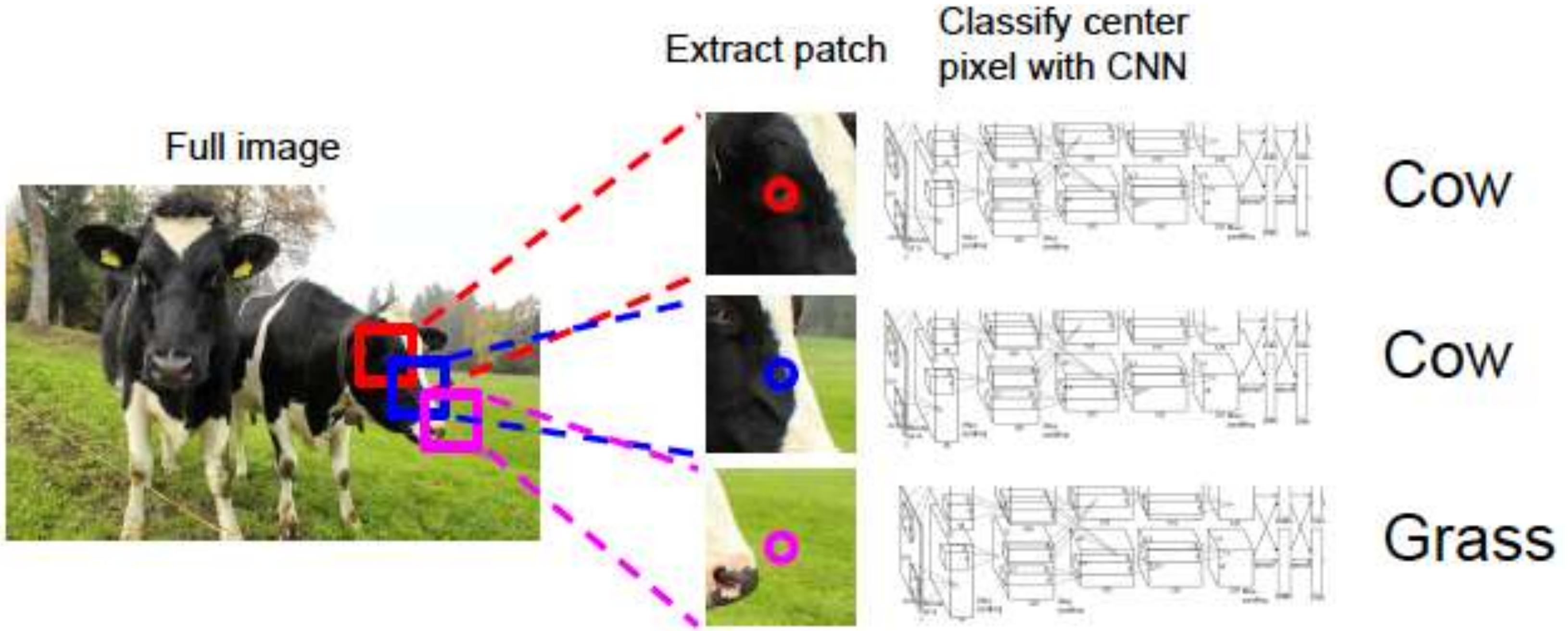
- A naïve approach



Why this is challenging?

上海科技大学 ShanghaiTech University

- A naïve approach



Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013
Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

Network for semantic segmentation

- Main idea for dense prediction
- Fully convolutional network
- Upsampling operators
- Multiscale context modeling

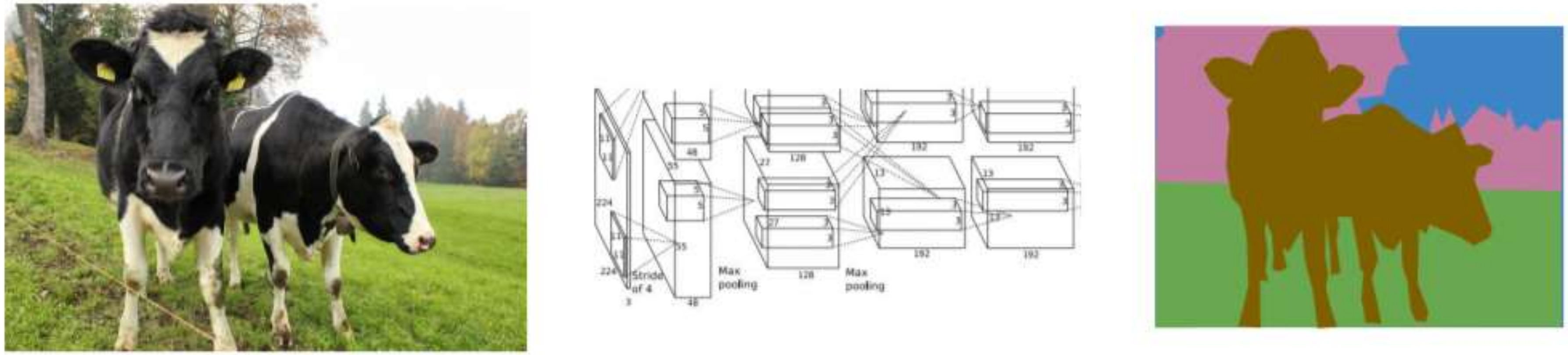
Acknowledgement: Feifei Li et al's cs231n notes

Several Ideas for SS

上海科技大学
ShanghaiTech University

- First idea

Full image

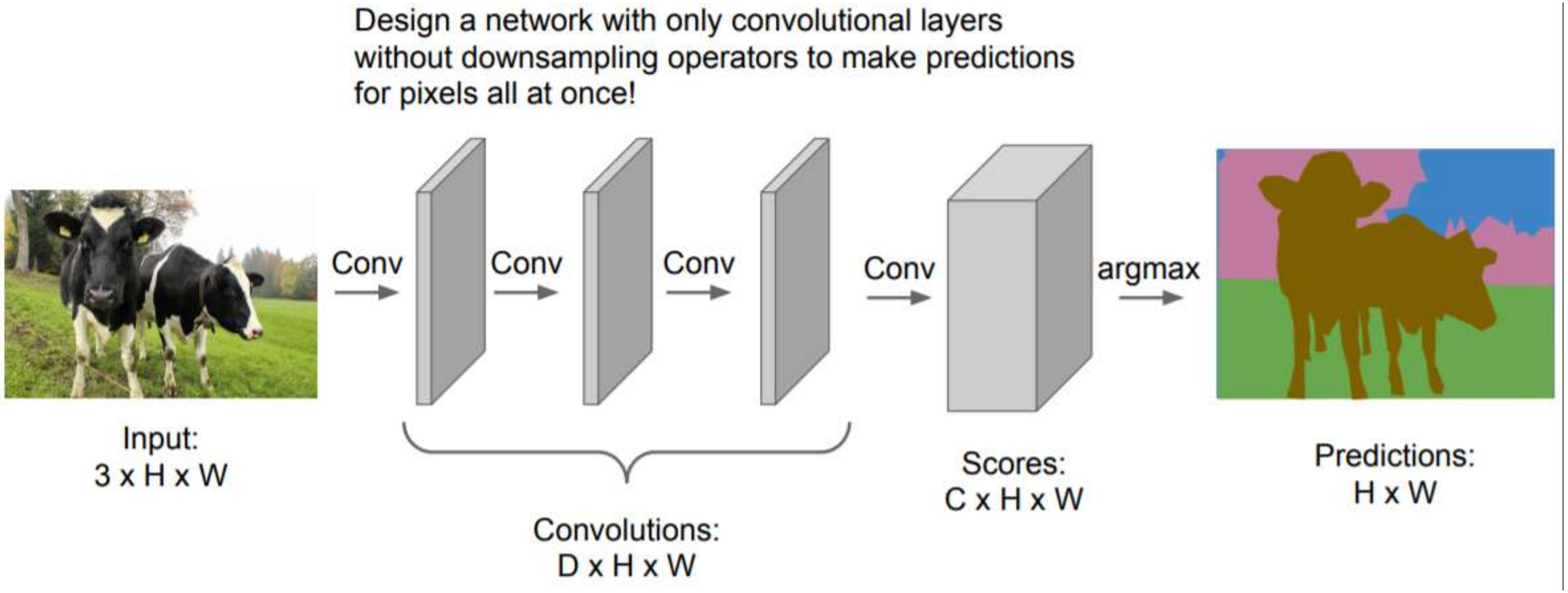


An intuitive idea: encode the entire image with conv net, and do semantic segmentation on top.

Problem: classification architectures often reduce feature spatial sizes to go deeper, but semantic segmentation requires the output size to be the same as input size.

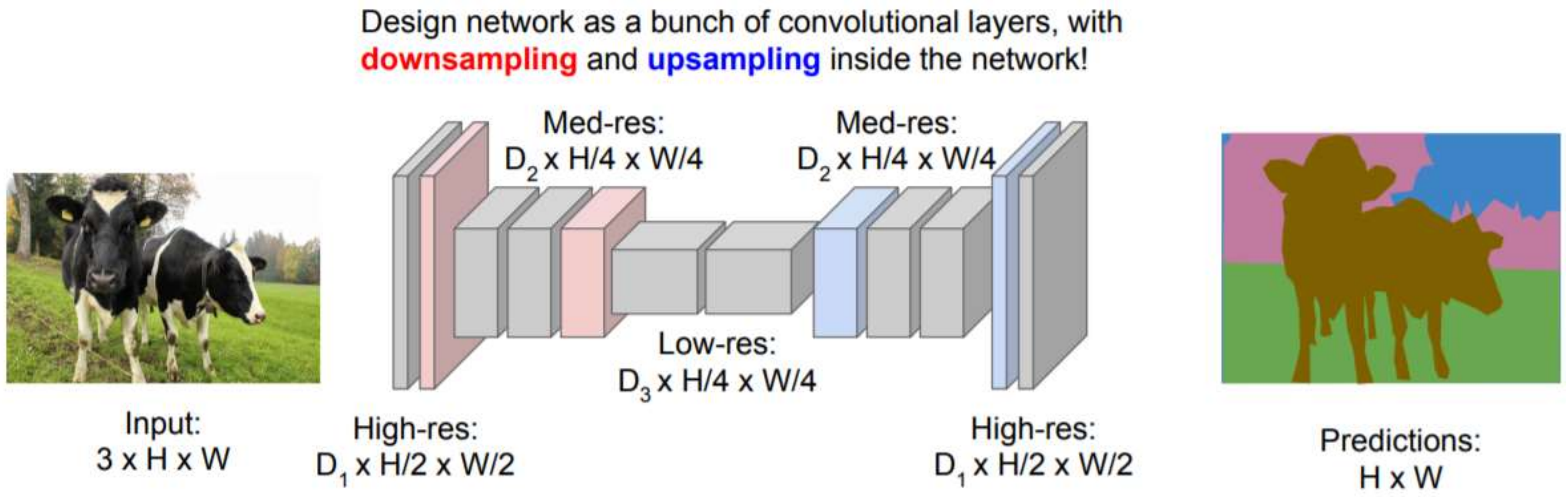
Several Ideas for SS

- Second idea



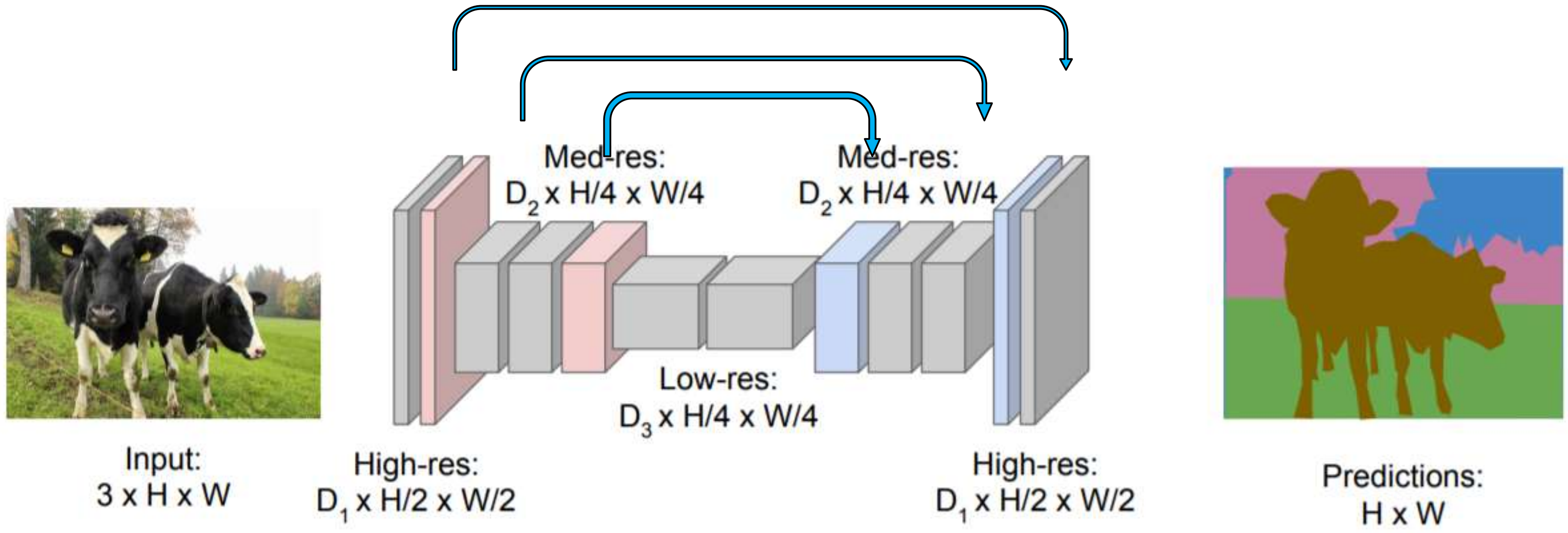
Several Ideas for SS

- Second idea improved



Several Ideas for SS

- Third idea



Network for semantic segmentation

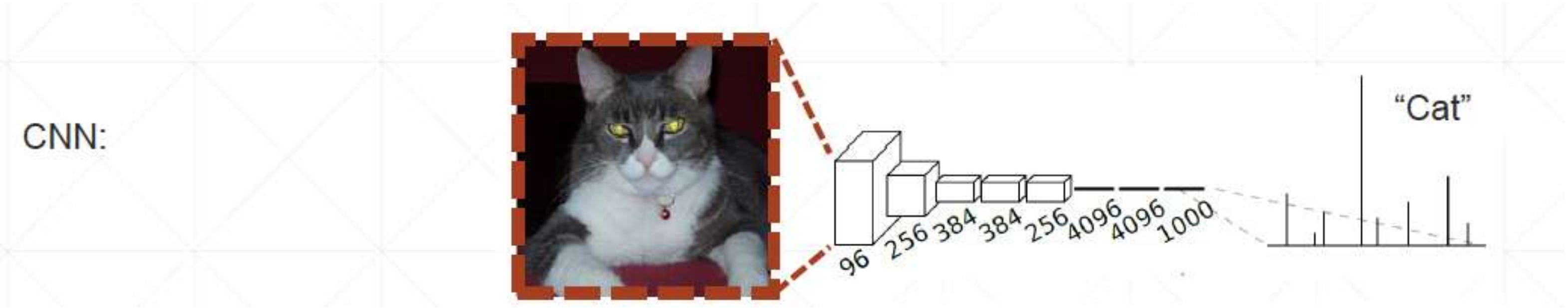
上海科技大学
ShanghaiTech University

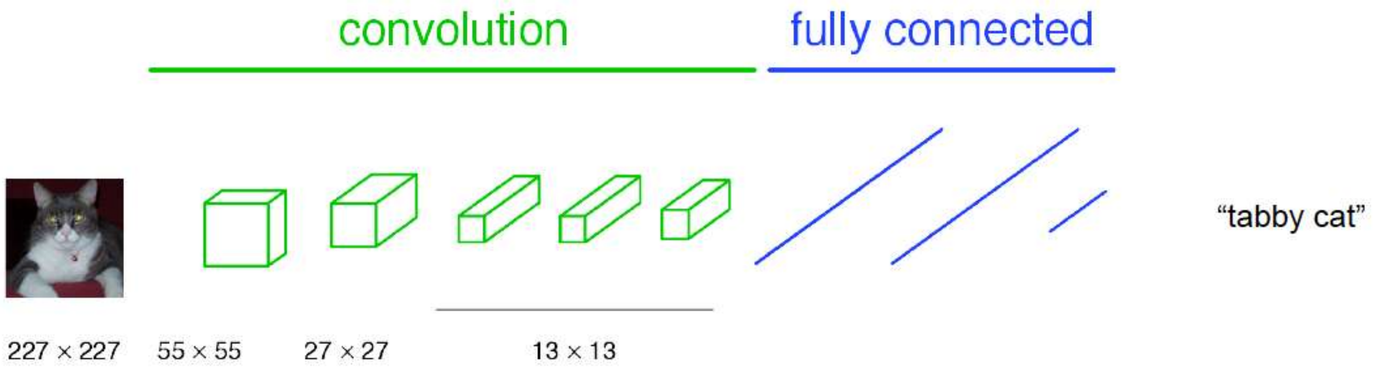
- Main idea for dense predictionedicti1o
- Fully convolutional network
- Upsampling operators
- Multiscale context modeling

Acknowledgement: Feifei Li et al's cs231n notes

Network Design I: Efficiency

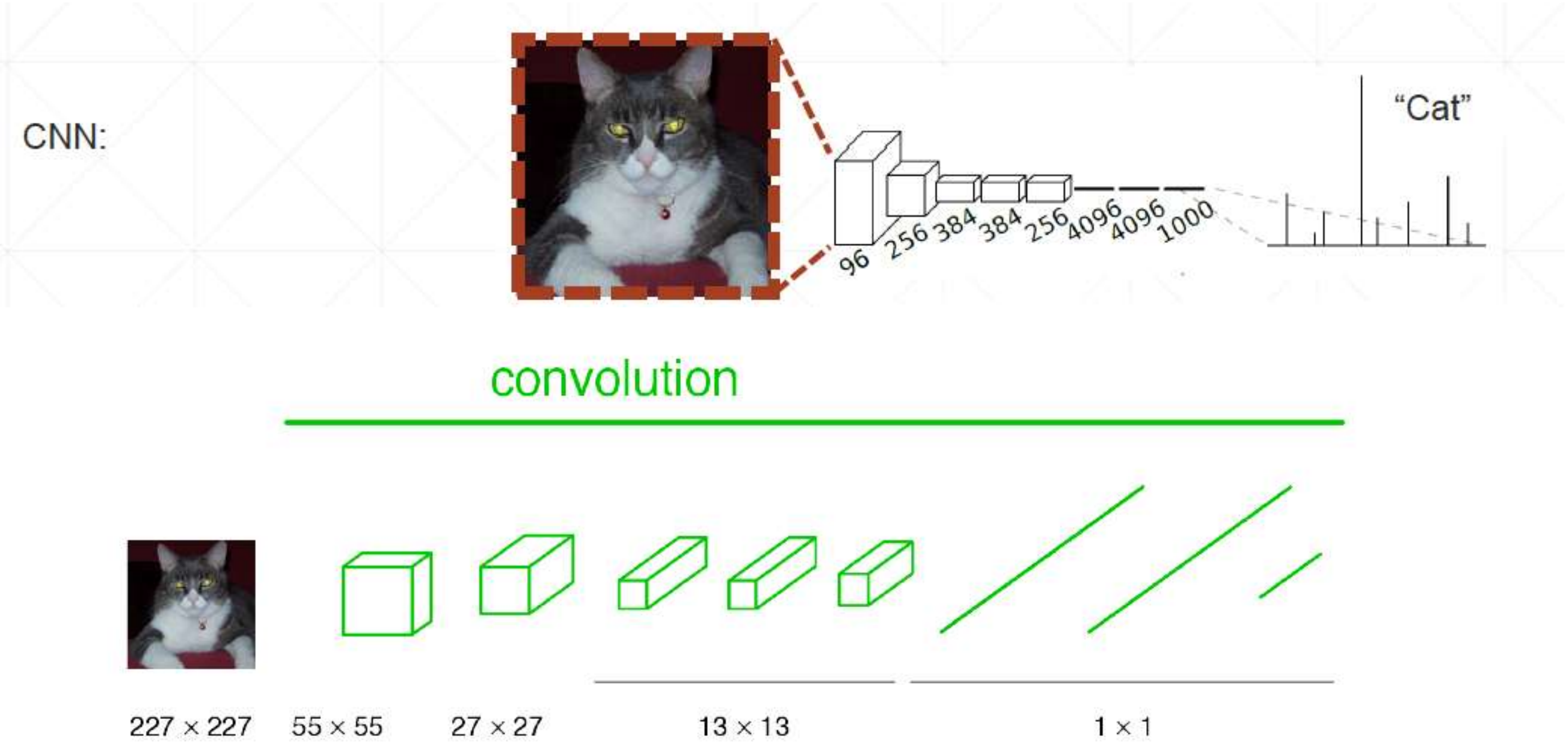
- Starting from a classification network





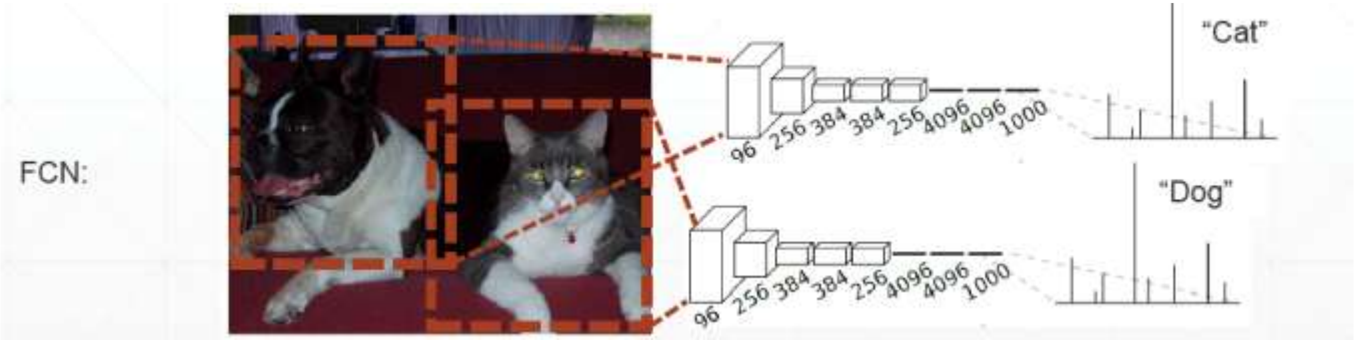
Network Design I: Efficiency

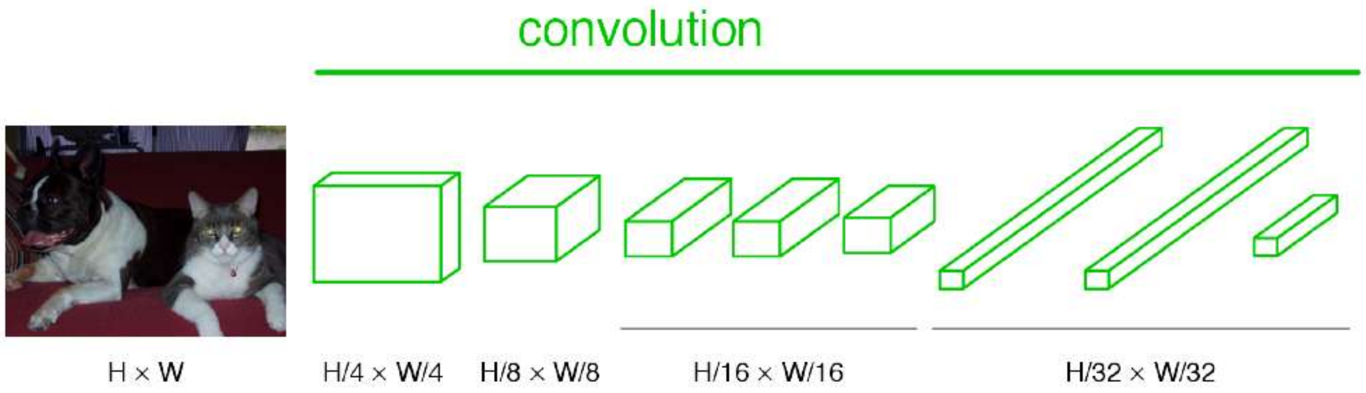
- Interpreting fully connected layers as 1x1 convolution (after reshaping)



Network Design I: Efficiency

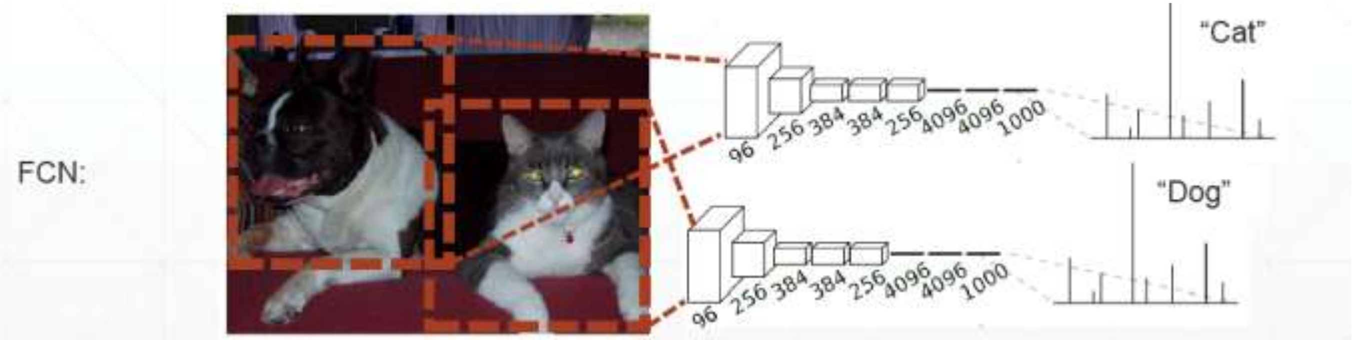
- Extending to a complete image

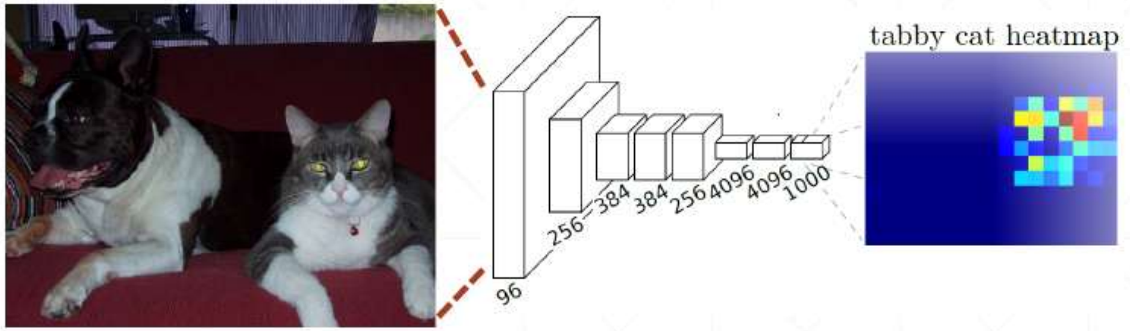




Network Design I: Efficiency

- Extending to a complete image





- Keep kernel sizes and strides
- Replace dense layer with convolution

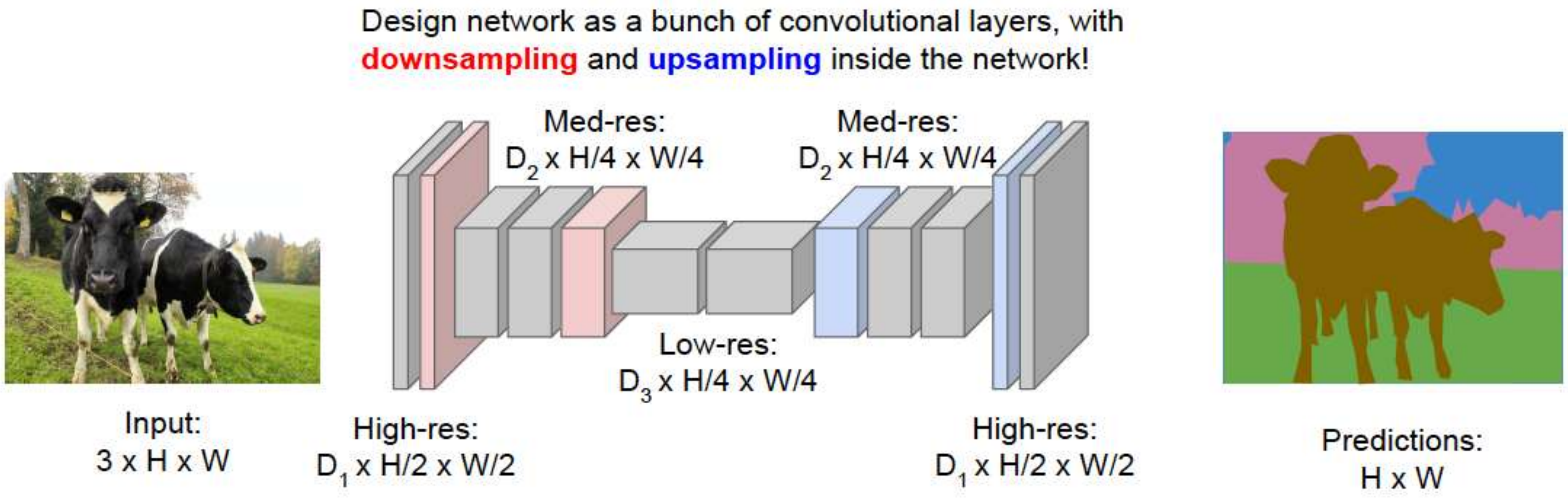
Network for semantic segmentation

- Main idea for dense predictionedicti1o
- Fully convolutional network
- Upsampling operators
- Multiscale context modeling

Acknowledgement: Feifei Li et al's cs231n notes

Network Design: Spatial resolution

- General encoder-decoder architecture



In-Network upsampling

- Unpooling

Nearest Neighbor

1	2
3	4

Input: 2 x 2

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Output: 4 x 4

"Bed of Nails"

1	2
3	4

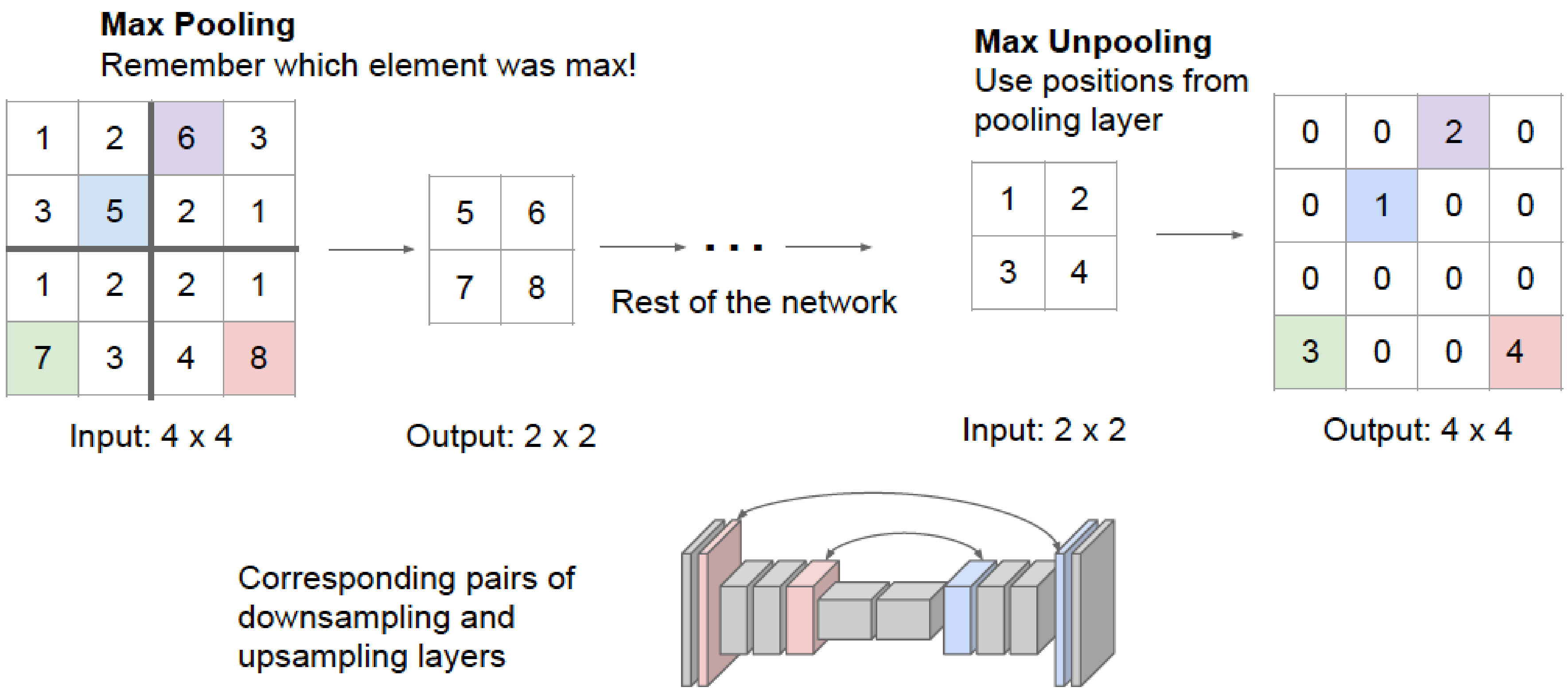
Input: 2 x 2

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Output: 4 x 4

In-Network upsampling

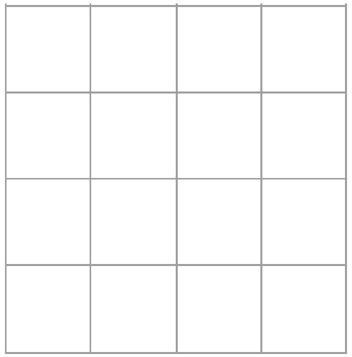
- Max Unpooling



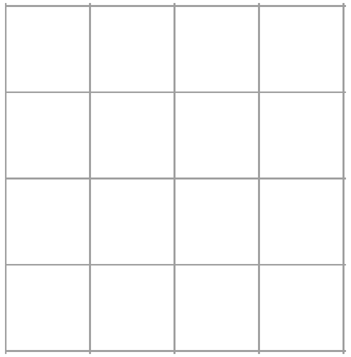
In-Network upsampling

- Learnable Upsampling: Transpose convolution

Recall: Typical 3 x 3 convolution, stride 1 pad 1



Input: 4 x 4

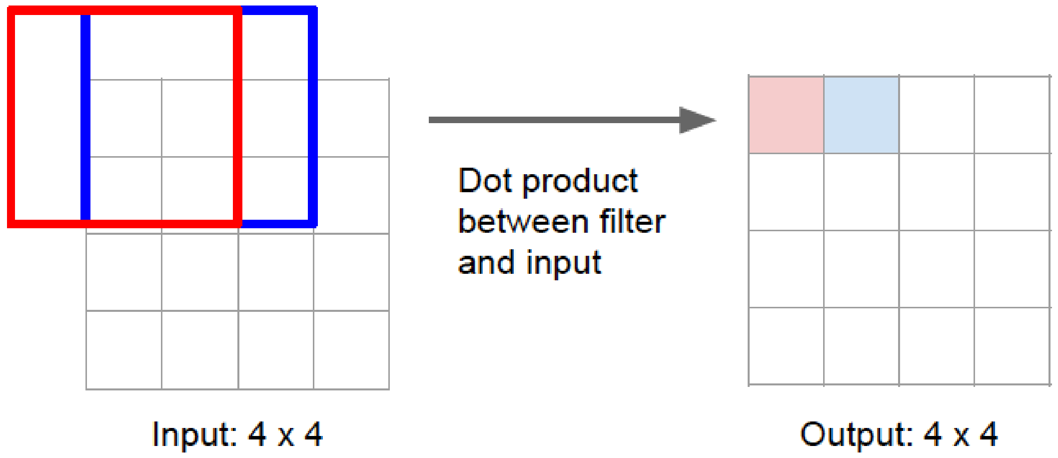


Output: 4 x 4

In-Network upsampling

- Learnable Upsampling: Transpose convolution

Recall: Normal 3 x 3 convolution, stride 1 pad 1



In-Network upsampling

- Learnable Upsampling: Transpose convolution

Recall: Normal 3 x 3 convolution, stride 2 pad 1

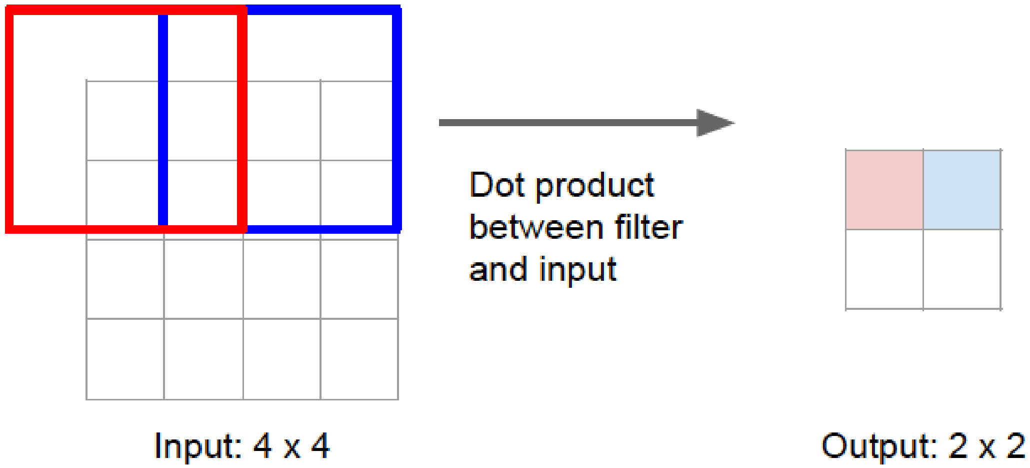
Input: 4 x 4

Output: 2 x 2

In-Network upsampling

- Learnable Upsampling: Transpose convolution

Recall: Normal 3 x 3 convolution, stride 2 pad 1



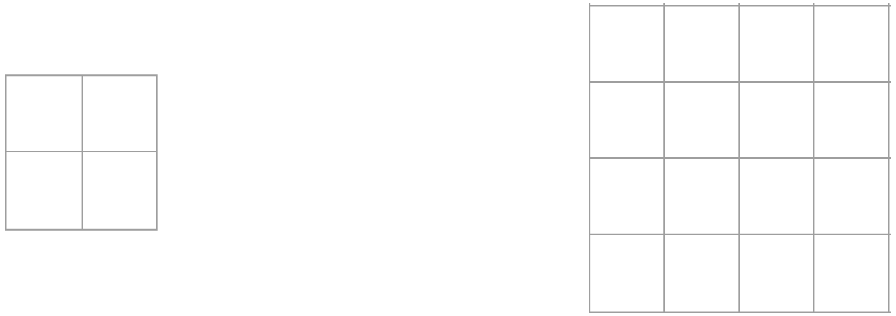
Filter moves 2 pixels in the input for every one pixel in the output

Stride gives ratio between movement in input and output

In-Network upsampling

- Learnable Upsampling: Transpose convolution

3 x 3 **transpose** convolution, stride 2 pad 1



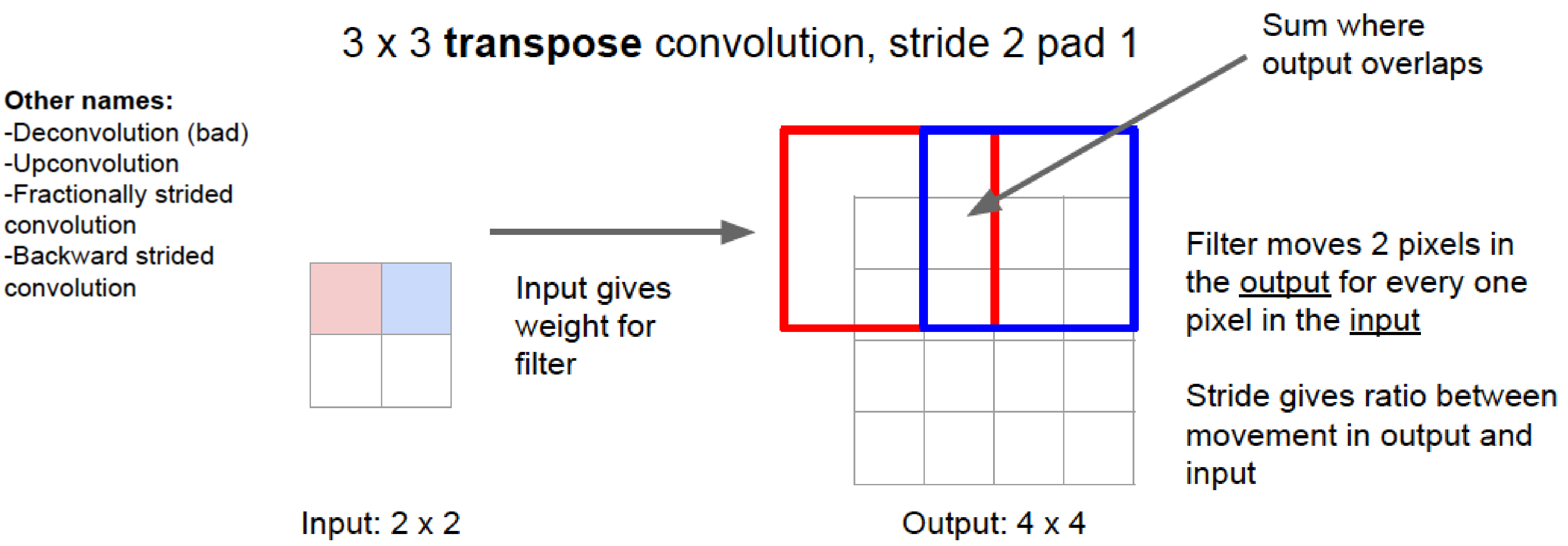
Input: 2 x 2

Output: 4 x 4

In-Network upsampling

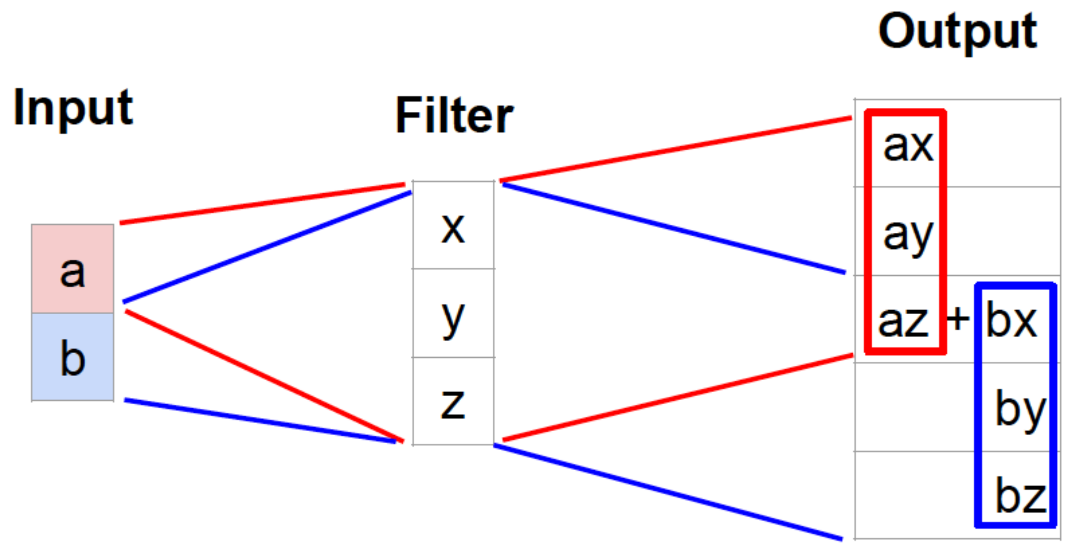
上海科技大学
ShanghaiTech University

- Learnable Upsampling: Transpose convolution



In-Network upsampling

- Learnable Upsampling: Transpose convolution
  - 1D example



Output contains copies of the filter weighted by the input, summing at where at overlaps in the output

Need to crop one pixel from output to make output exactly 2x input

In-Network upsampling

上海科技大学 ShanghaiTech University

- Learnable Upsampling: Transpose convolution
  - 1D example

We can express convolution in terms of a matrix multiplication

$$\vec{x} * \vec{a} = X\vec{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & 0 & x & y & z & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ bx + cy + dz \end{bmatrix}$$

Example: 1D conv, kernel size=3, stride=2, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

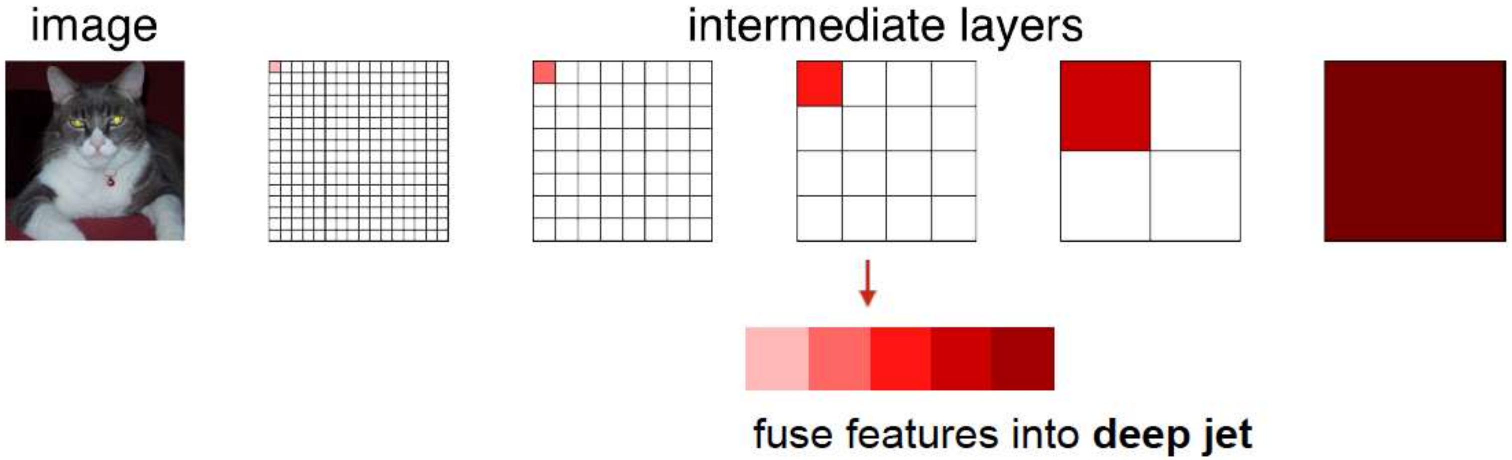
$$\vec{x} *^T \vec{a} = X^T\vec{a}$$

$$\begin{bmatrix} x & 0 \\ y & 0 \\ z & x \\ 0 & y \\ 0 & z \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} ax \\ ay \\ az + bx \\ by \\ bz \\ 0 \end{bmatrix}$$

When stride>1, convolution transpose is no longer a normal convolution!

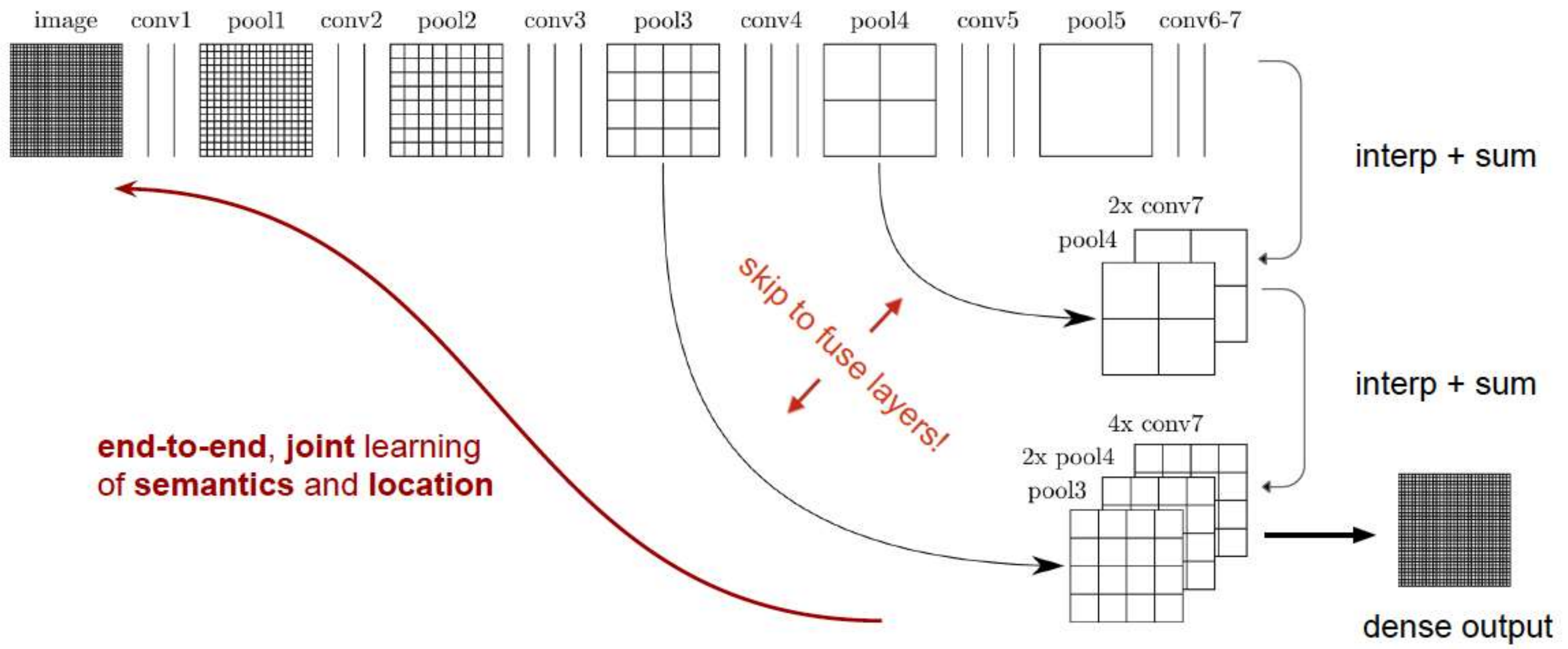
Network Design: Examples

- Fully Convolutional Network [Long et al, CVPR 2015]
  - Upsampling: low-resolution, lack spatial details
  - Combining *where (local, shallow)* with *what (global, deep)*



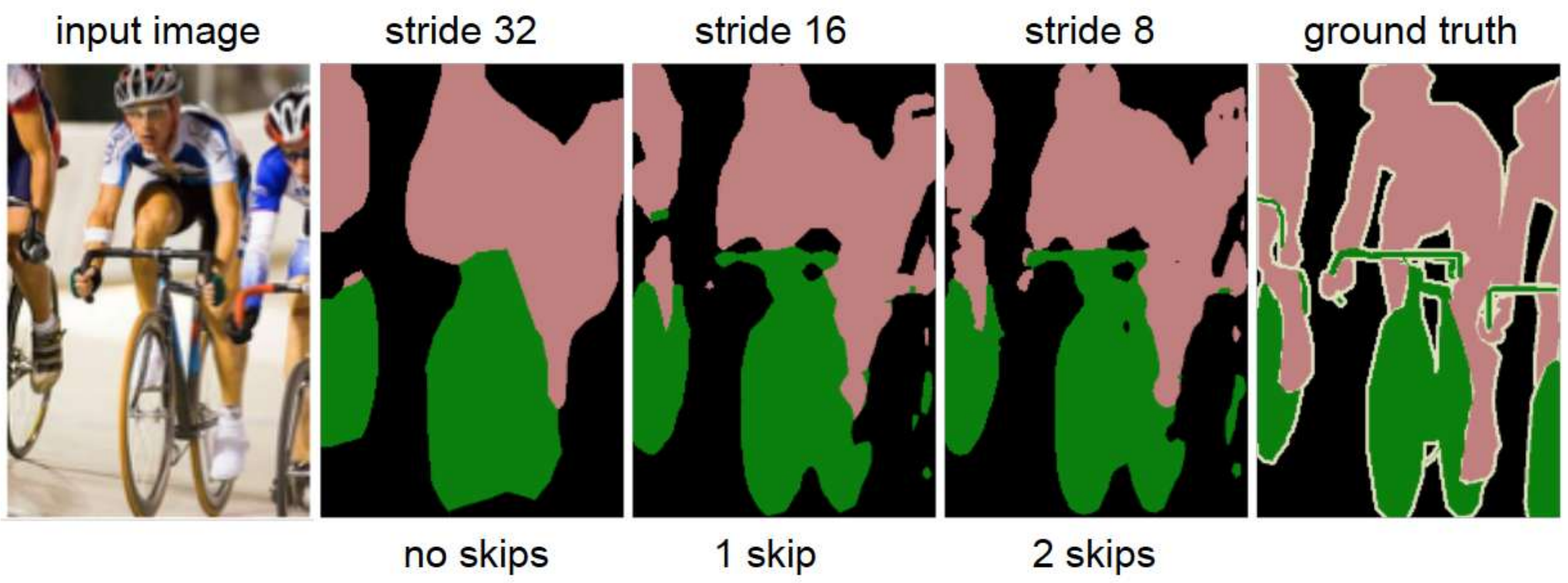
Network Design: Examples

- Fully Convolutional Network [Long et al, CVPR 2015]
  - Upsampling: low-resolution, lack spatial details
  - Introducing ***skip layers***



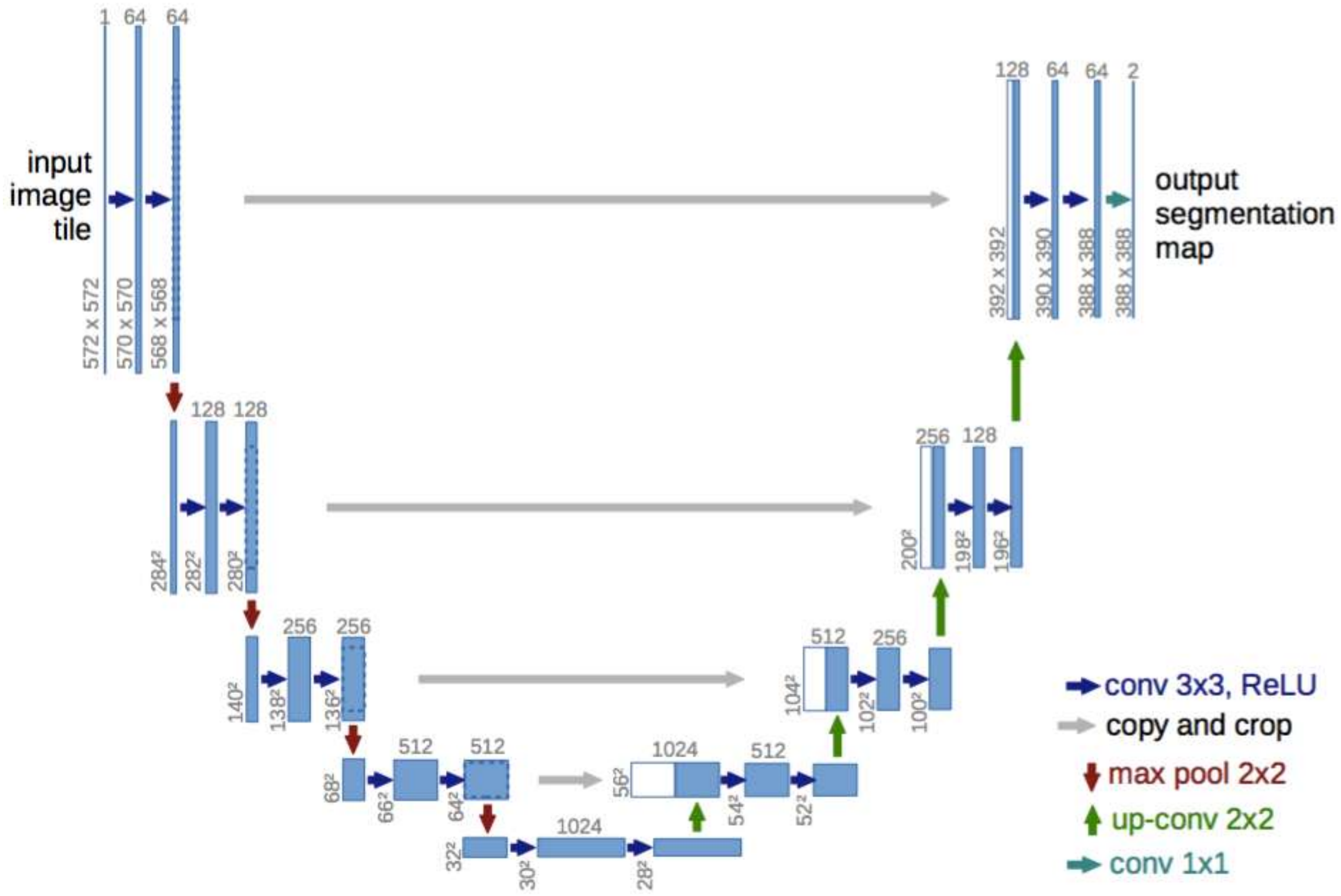
Network Design: Examples

- Fully Convolutional Network [Long et al, CVPR 2015]
  - Upsampling: low-resolution, lack spatial details
  - Skip layer refinement



Network Design: Examples

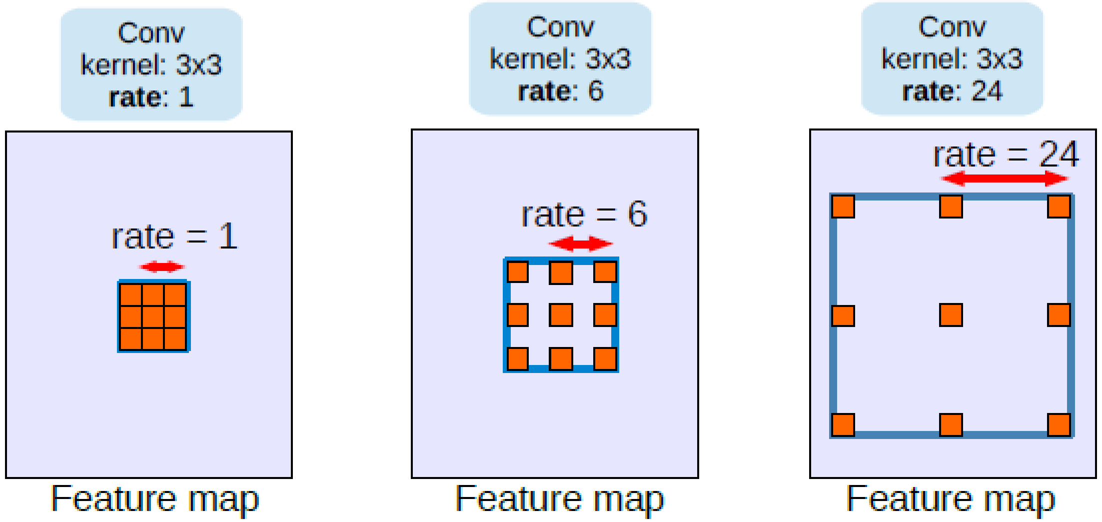
- U-Net [Ronneberger et al, MICCAI 2015]



Network Design: Spatial resolution

上海科技大学 ShanghaiTech University

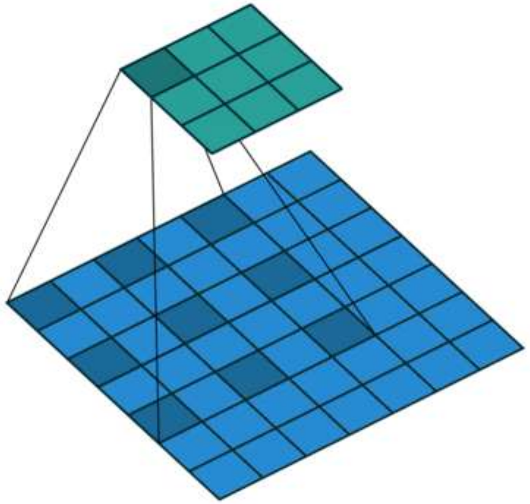
- Dilated Convolutional Network [Yu and Koltun, ICLR 2016]
  - Dense feature map without upsampling
  - ***Dilated (or Atrous) convolution***



$$y[i] = \sum_{k=1}^{K} x[i + r \cdot k]w[k].$$

Network Design: Spatial resolution

- Dilated Convolutional Network [Yu and Koltun, ICLR 2016]
  - Dense feature map without upsampling
  - ***Dilated (or Atrous) convolution***

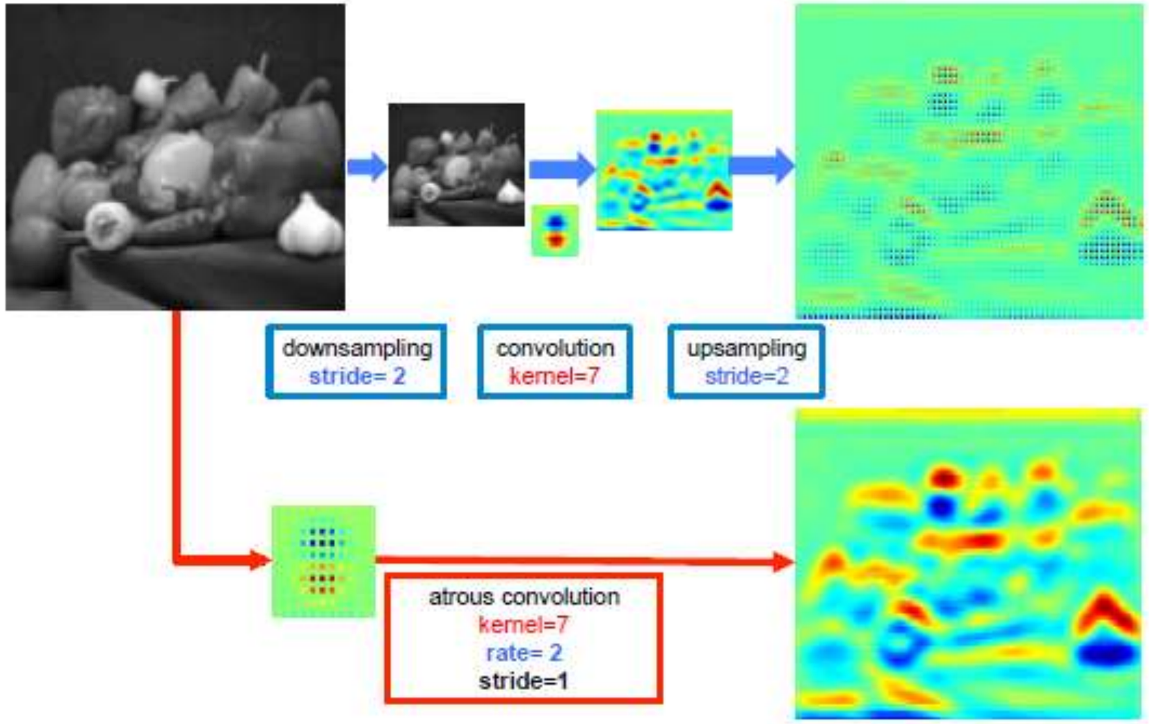


$$y[i] = \sum_{k=1}^{K} x[i + r \cdot k] w[k].$$

Network Design: Spatial resolution

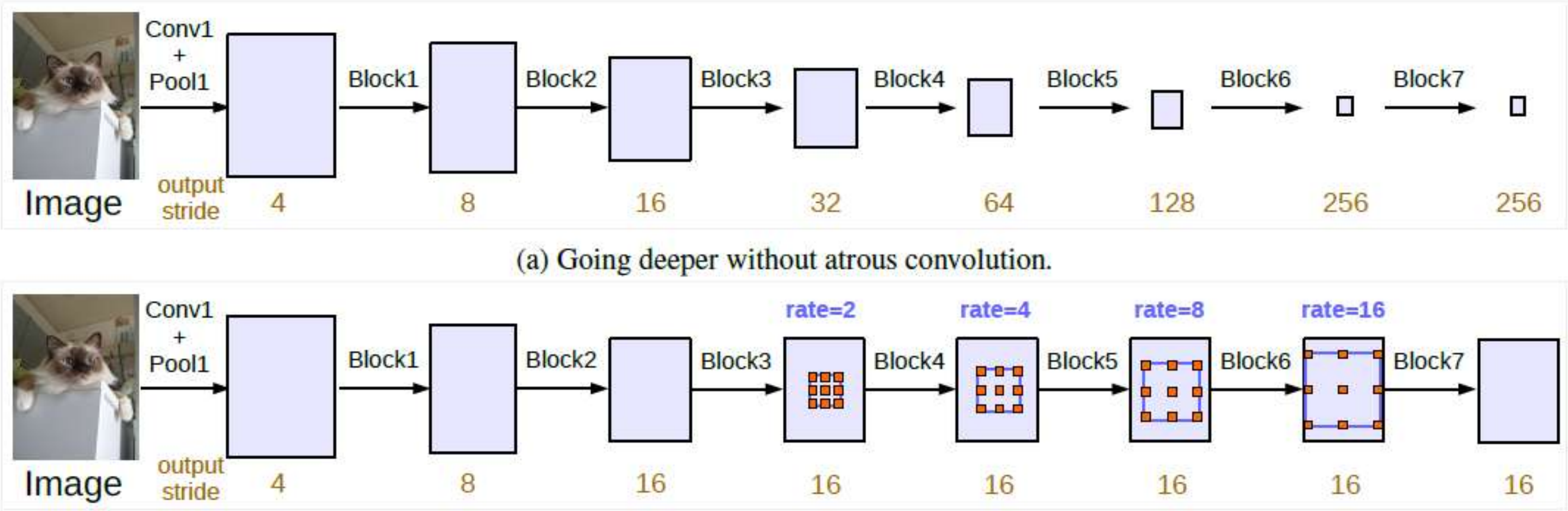
上海科技大学 ShanghaiTech University

- Dilated Convolutional Network [Yu and Koltun, ICLR 2016]
  - Dense feature map without upsampling
  - ***Dilated (or Atrous) convolution***



Network Design: Spatial resolution

- Dilated Convolutional Network [Yu and Koltun, ICLR 2016]
  - Dense feature map without upsampling
  - ***Dilated (or Atrous) convolution***

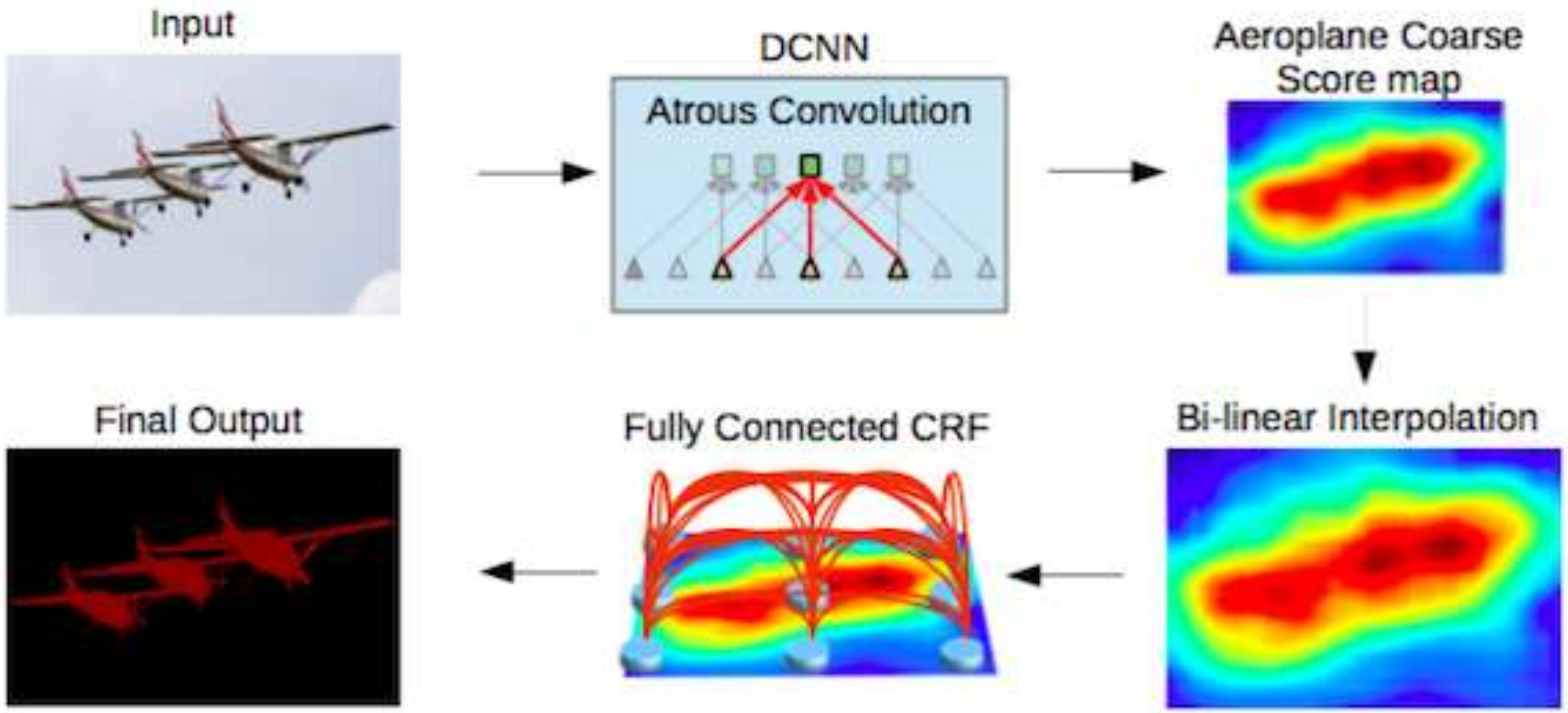


(a) Going deeper without atrous convolution.

(b) Going deeper with atrous convolution. Atrous convolution with $rate > 1$ is applied after block3 when $output\_stride = 16$.

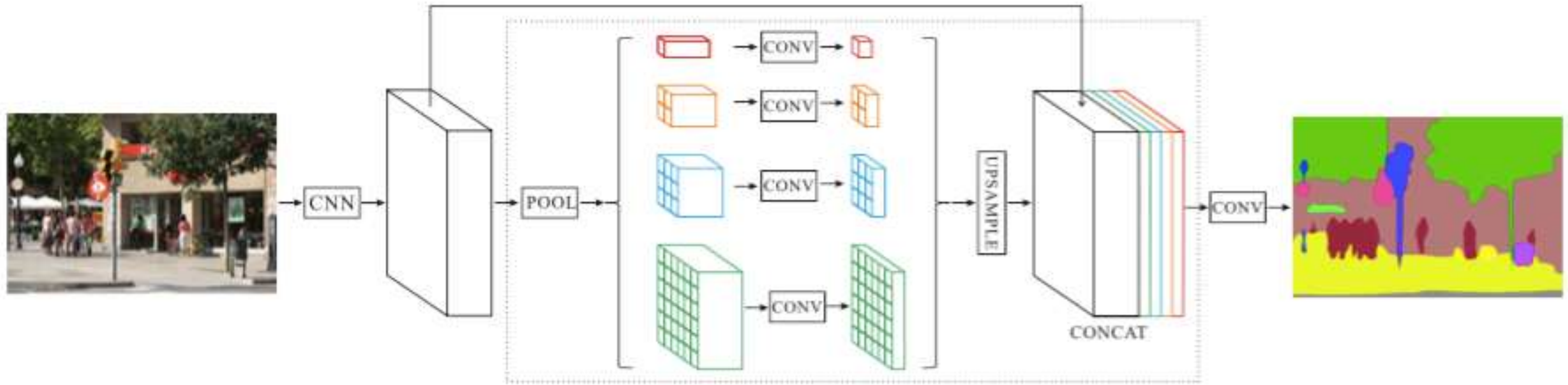
Network Design: Multi-scale context

- DeepLab v1&v2
  - Post-processing with dense CRFs.



Network Design: Multi-scale context

- PSPNet [Zhao et al CVPR 2017]
  - A pyramid parsing module that carries both local and global context information

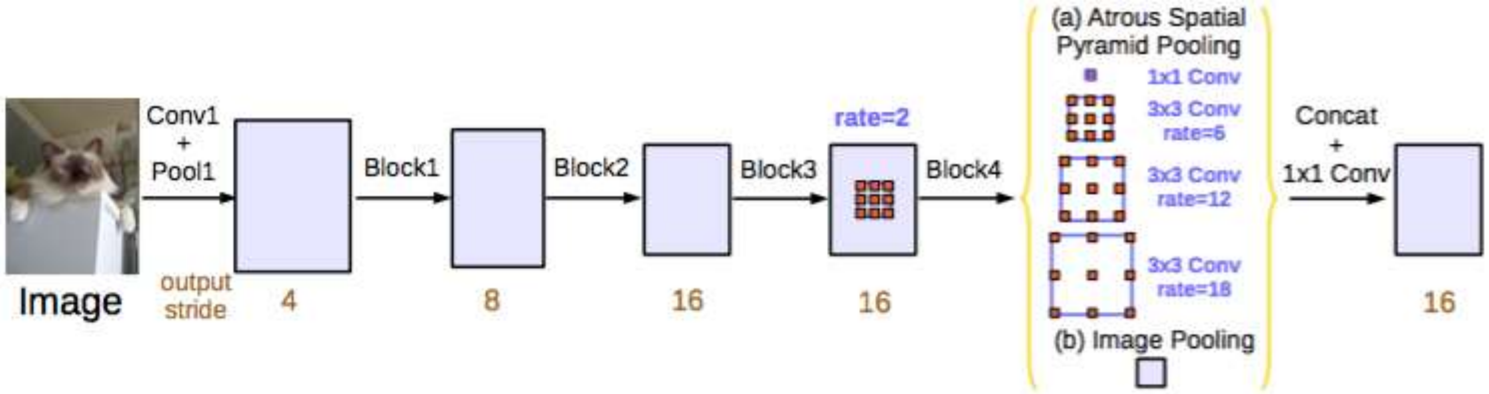


(a) Input Image (b) Feature Map (c) Pyramid Pooling Module (d) Final Prediction

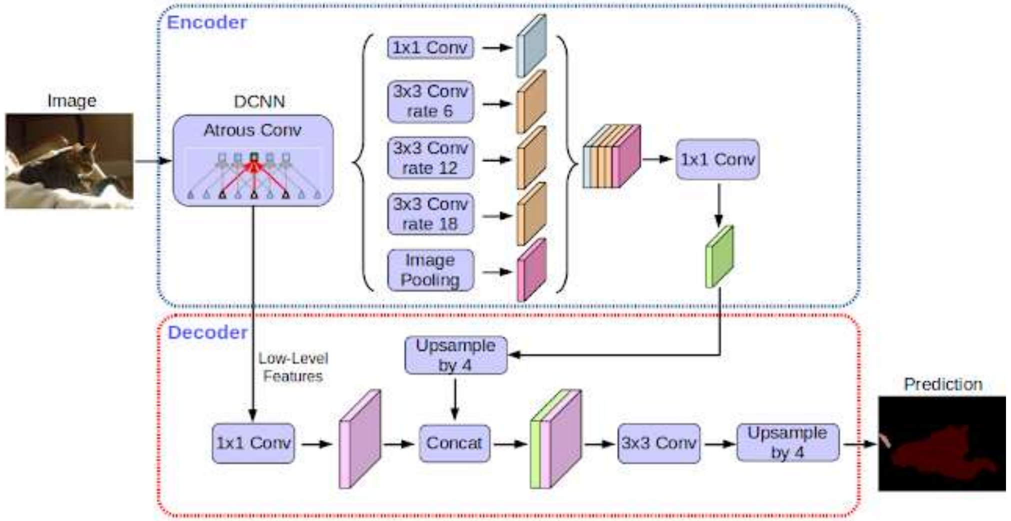
Network Design: Multi-scale context

上海科技大学 ShanghaiTech University

- DeepLab v3



- Deeplab v3+

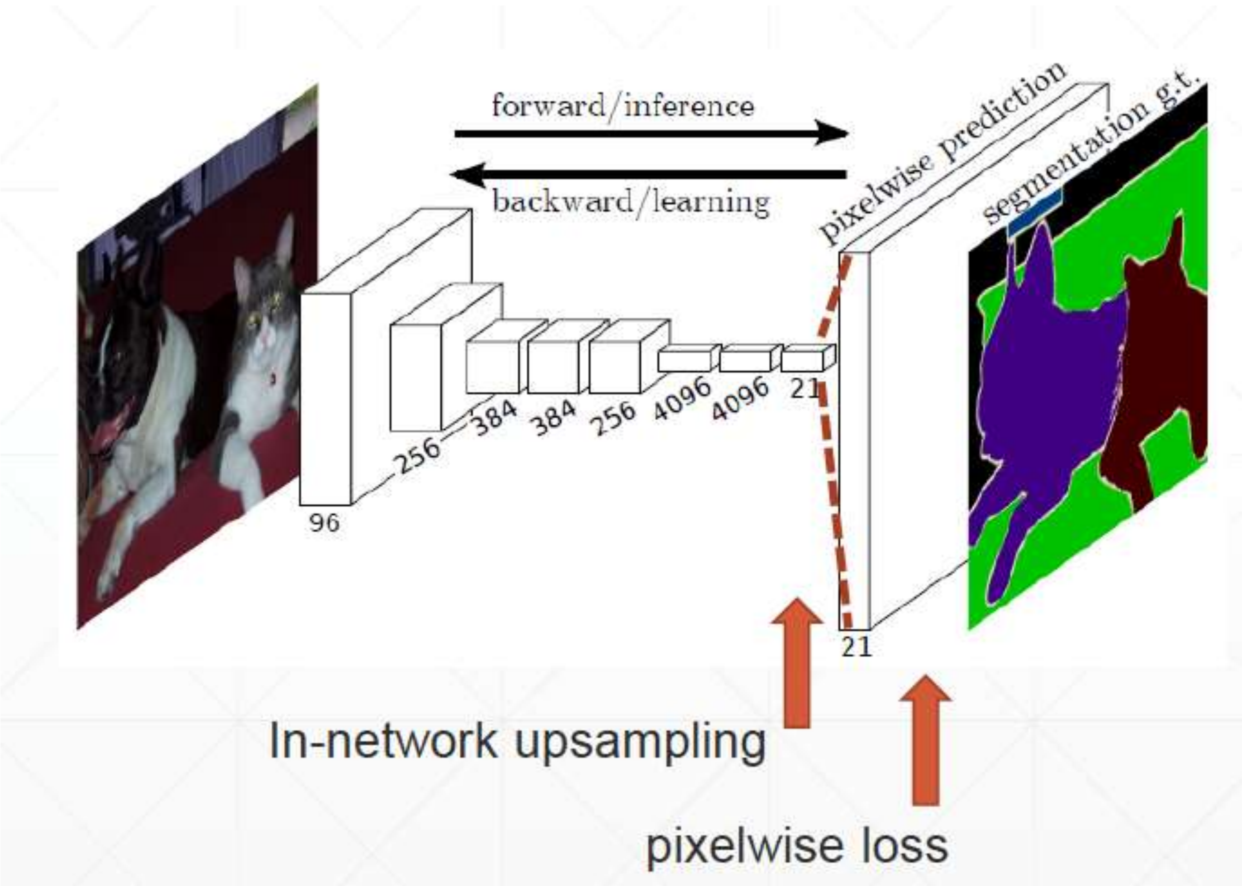


https://ai.googleblog.com/2018/03/semantic-image-segmentation-with.html

Semantic segmentation: loss function

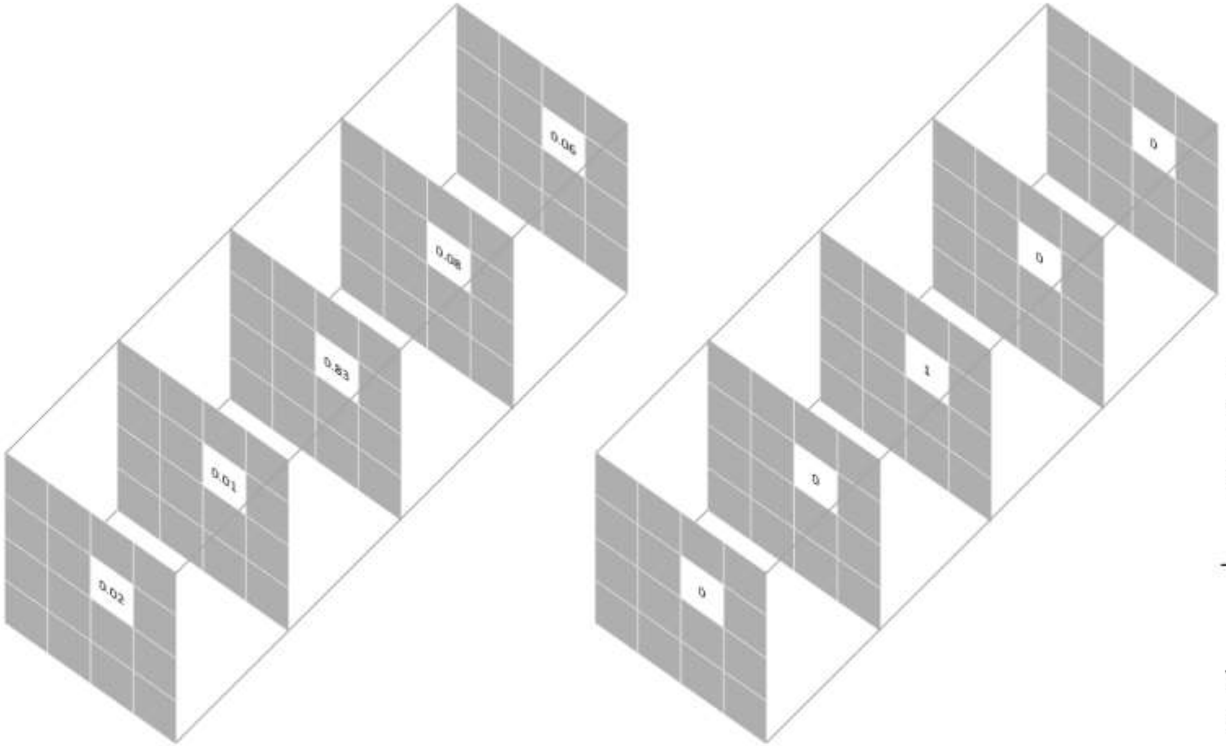
上海科技大学 ShanghaiTech University

- Main idea: pixel-wise classification



Semantic segmentation: loss function

- Pixel-wise loss



Prediction for a selected pixel

Target for the corresponding pixel

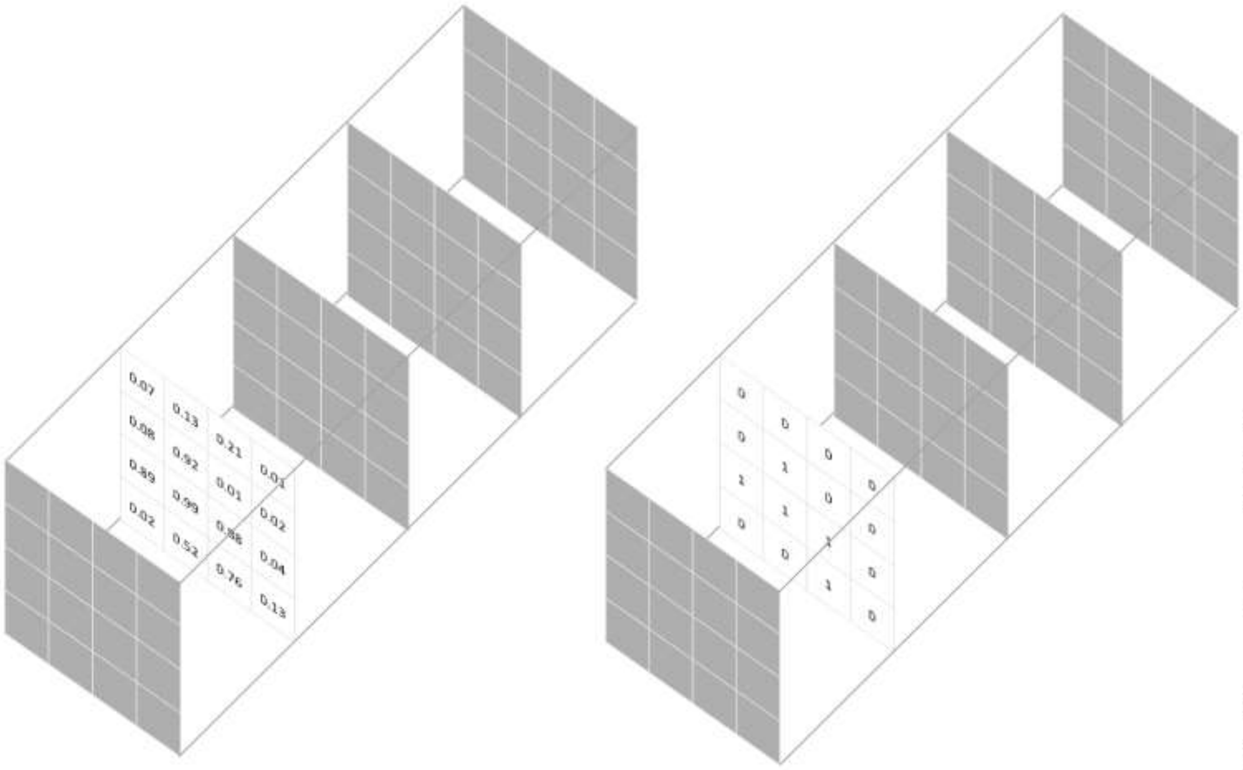
Pixel-wise loss is calculated as the log loss, summed over all possible classes

$$-\sum_{classes} y_{true} \log\left(y_{pred}\right)$$

This scoring is repeated over all **pixels** and averaged

Semantic segmentation: loss function

- Region-based loss



$$Dice = \frac{2|A \cap B|}{|A| + |B|}$$

Soft Dice coefficient is calculated for each class mask

$$1 - \frac{2\sum_{pixels} y_{true} y_{pred}}{\sum_{pixels} y_{true}^2 + \sum_{pixels} y_{pred}^2}$$

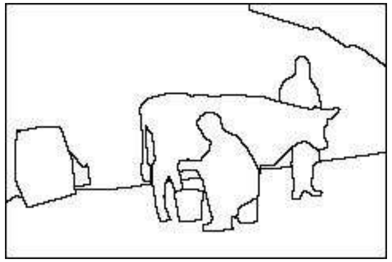
This scoring is repeated over all **classes** and averaged

Prediction for a selected class

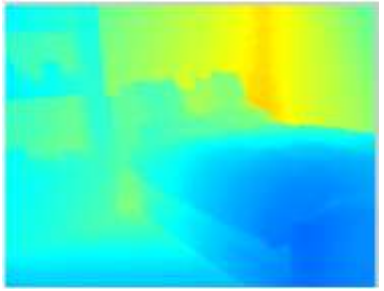
Target for the corresponding class

Semantic Segmentation: Summary

- Pixel-wise annotation of images
  - An instance of scene understanding

Boundary

Depth

- Other research topics (not discussed)
  - *Low-level vision: superresolution, deblurring, inpainting, depth*
  - *Video: optical flow, action and activity recognition and detection*
  - *Volumetric/Multimodality: RGB-D images, medical imaging, etc.*