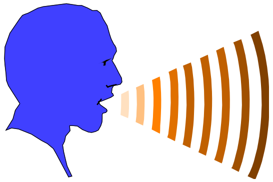


Image and Video Processing

Convolutional Networks for Image Processing (Part II)

Yao Wang
Tandon School of Engineering, New York University

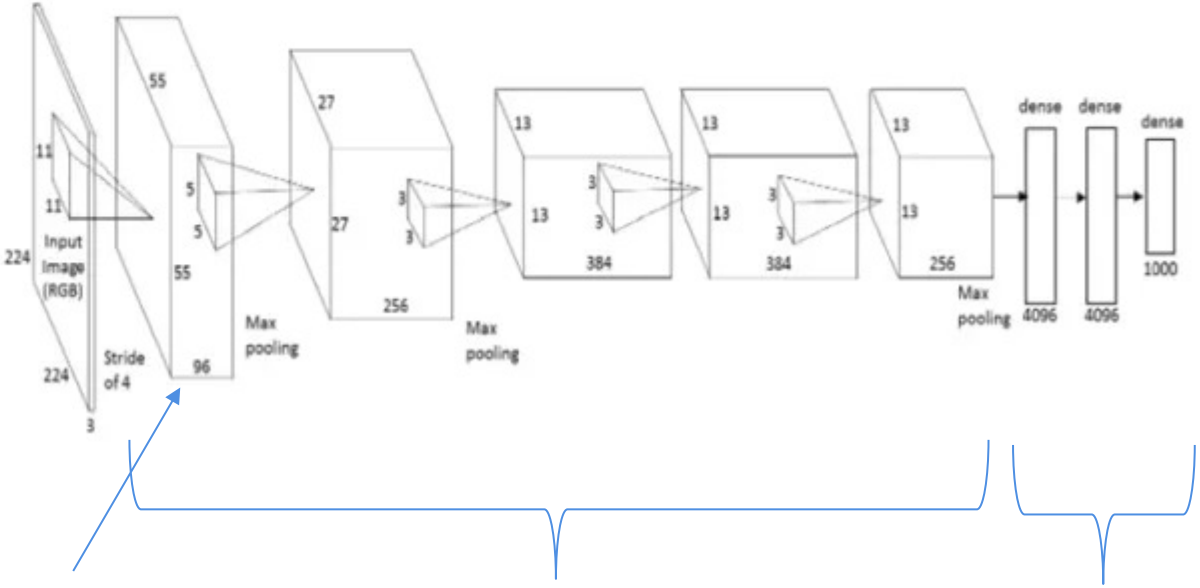
Outline (Part I)

- Supervised learning: General concepts
- Neural network architecture
- Convolutional network architecture
  - Why using convolution and many layers
  - Multichannel convolution
  - Pooling
- Deep networks
- Model training
  - Loss functions
  - Stochastic gradient descent: general concept
  - Data Preprocessing and Regularization
- Training, validation and testing and cross validation

Outline (Part II)

- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

Example Conv. Network



- Alex Net
- Each convolutional layer has:
  - 2D convolution
  - Activation (eg. ReLU)
  - Pooling or sub-sampling

96 feature maps of size 55x55 each

Convolutional layers
For feature extraction

2D convolution with Activation and pooling / sub-sampling

Fully connected layers
For Classification task

Matrix multiplication & activation

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012.

Training with Gradient Descent

- Given training data: $(\boldsymbol{x}_i, y_i), i = 1, \ldots, N$
- Learn parameters: $\theta = (W_H, b_H, W_o, b_o)$ (assuming a MLP with one hidden layer)
  - Weights and biases for hidden and output layers
  - W_H are filter kernels in conv. layer
- Neural network training (like all training): Minimize loss function

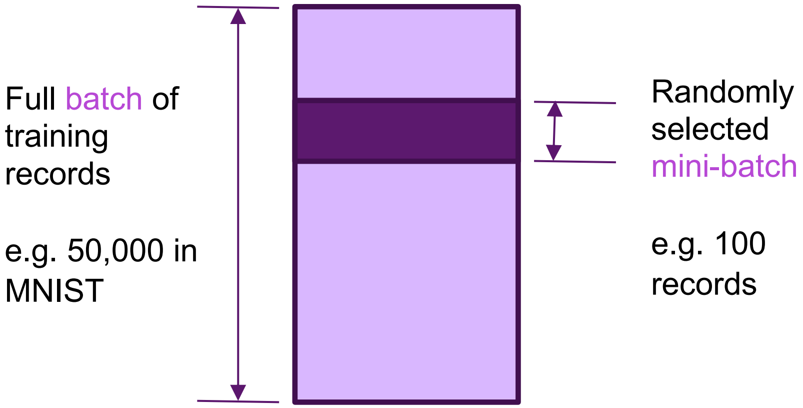
$$\hat{\theta} = \arg\min_{\theta} L(\theta)\,, \qquad L(\theta) = \sum_{i=1}^{N} L_i(\theta, \boldsymbol{x}_i, y_i)$$

  - $L_i(\theta, \boldsymbol{x}_i, y_i)$ = loss on sample i for parameter θ
- Standard gradient descent:

$$\theta^{k+1} = \theta^k - \alpha \nabla L(\theta^k) = \theta^k - \alpha \sum_{i=1}^{N} \nabla L_i(\theta^k, \boldsymbol{x}_i, y_i)$$

  - Each iteration requires computing N loss functions and gradients
  - But, gradient computation is expensive when data size N large

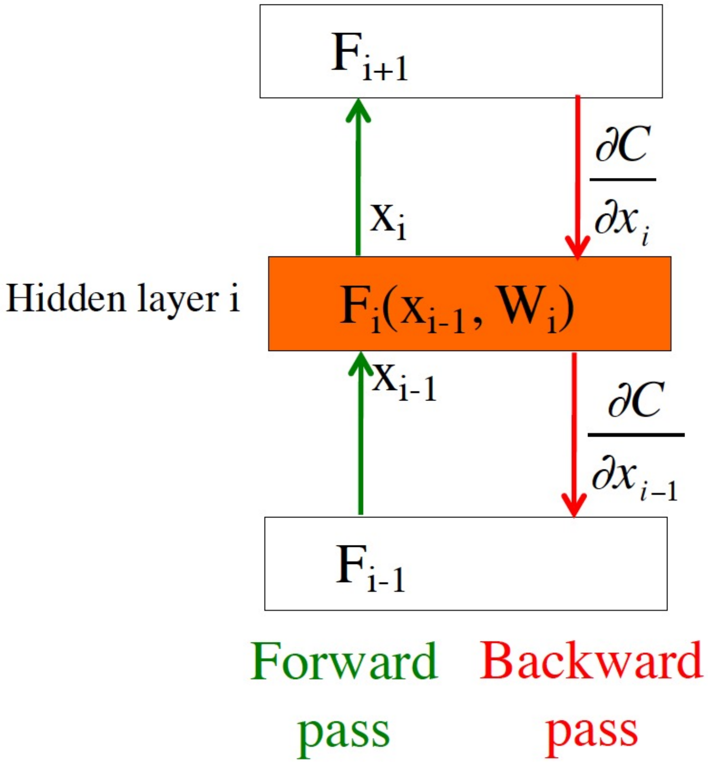
Stochastic Gradient Descent



- In each step:
  - Select random small "mini-batch"
  - Evaluate gradient on mini-batch

- For $t = 1$ to N_{steps}
  - Select random mini-batch $I \subset \{1, \ldots, N\}$
  - Compute gradient approximation:
    $$g^t = \frac{1}{|I|} \sum_{i \in I} \nabla L(x_i, y_i, \theta)$$
  - Update parameters: $\theta^{t+1} = \theta^t - \alpha^t g^t$

Backpropagation: layer i



- Layer i has two inputs (during training)

$$x_{i-1} \qquad \frac{\partial C}{\partial x_i}$$

- For layer i, we need the derivatives:

$$\frac{\partial F_i(x_{i-1}, w_i)}{\partial x_{i-1}} \qquad \frac{\partial F_i(x_{i-1}, w_i)}{\partial w_i}$$

- We compute the outputs

$$x_i = F_i(x_{i-1}, w_i)$$

$$\frac{\partial C}{\partial x_{i-1}} = \frac{\partial C}{\partial x_i} \cdot \frac{\partial F_i(x_{i-1}, w_i)}{\partial x_{i-1}}$$

- The weight update equation is:

$$\frac{\partial C}{\partial w_i} = \frac{\partial C}{\partial x_i} \cdot \frac{\partial F_i(x_{i-1}, w_i)}{\partial w_i}$$

$$w_i^{k+1} \leftarrow w_i^k + \eta_t \frac{\partial E}{\partial w_i}$$

(sum over all
training examples
to get E)

From Fergus: https://cs.nyu.edu/~fergus/teaching/vision/2_neural_nets.pdf

Tensor gradient

- Generally, you may take gradient of a variable of dimension N, $y = [y_1, y_2, \ldots, y_N]$ with respect to av variable of dimension M, $x = [x_1, x_2, \ldots, x_M]$
- $\frac{\delta y}{\delta x} = \left[\frac{\delta y_n}{\delta x_m}\right]$ (a $N \times M$ matrix)
- Previous example if it is a fully connected layer:
  - C is the loss function, a scalar
  - x_i may have dimension N_i, x_{i-1} may have dimension N_{i-1}, w_i may have dimension $N_{i-1}N_i$
- Previous example if it is a convolution layer:
  - x_i may contain L_i feature maps of spatial dimension $M_i \times N_i$, x_{i-1} may contain L_{i-1} feature maps of spatial dimension $M_{i-1} \times N_{i-1}$, w_i would correspond to multichannel filters with dimension $L_i L_{i-1} R_{i-1} C_{i-1}$ for filter size of $R_{i-1} \times C_{i-1}$

Outline (Part II)

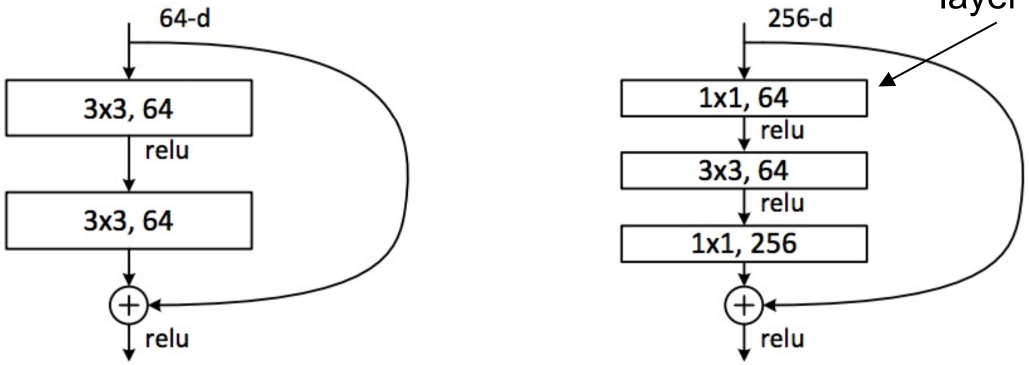
- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

Some Important Extensions

- Residual connections
- Dense connections
- Dilated convolution

Residual Connections (ResNET)

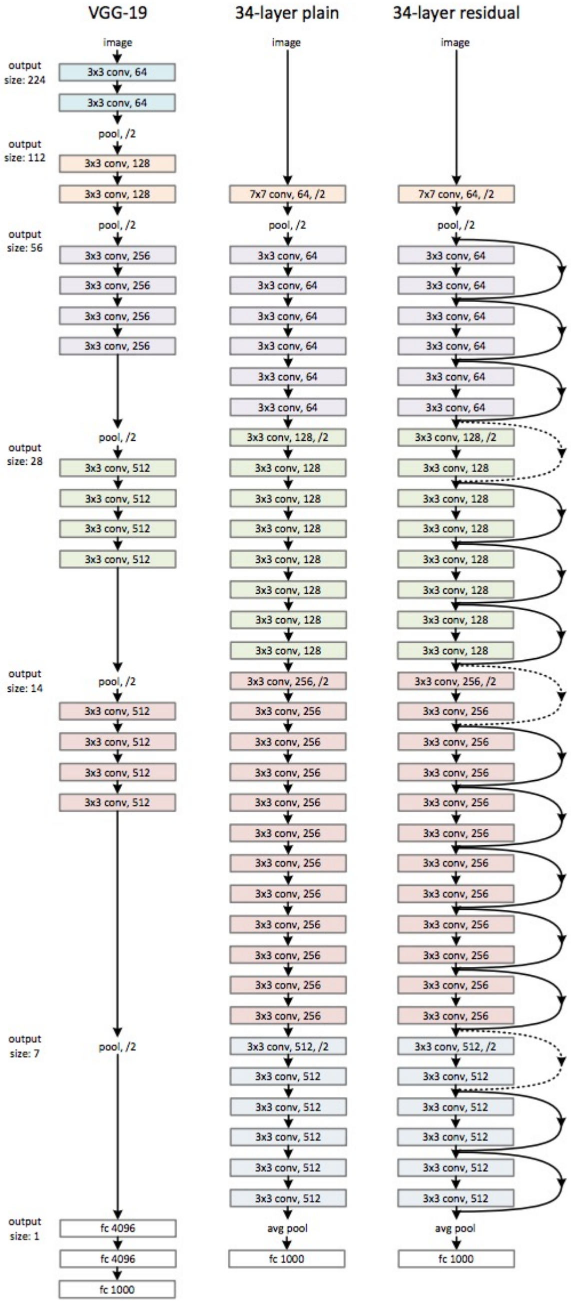
- Really, really deep convnets don’t train well
  - Gradient of final loss does not propagate back to earlier layers (vanishing of gradients)
- Key idea: introduce “pass through” into each layer for back propagation



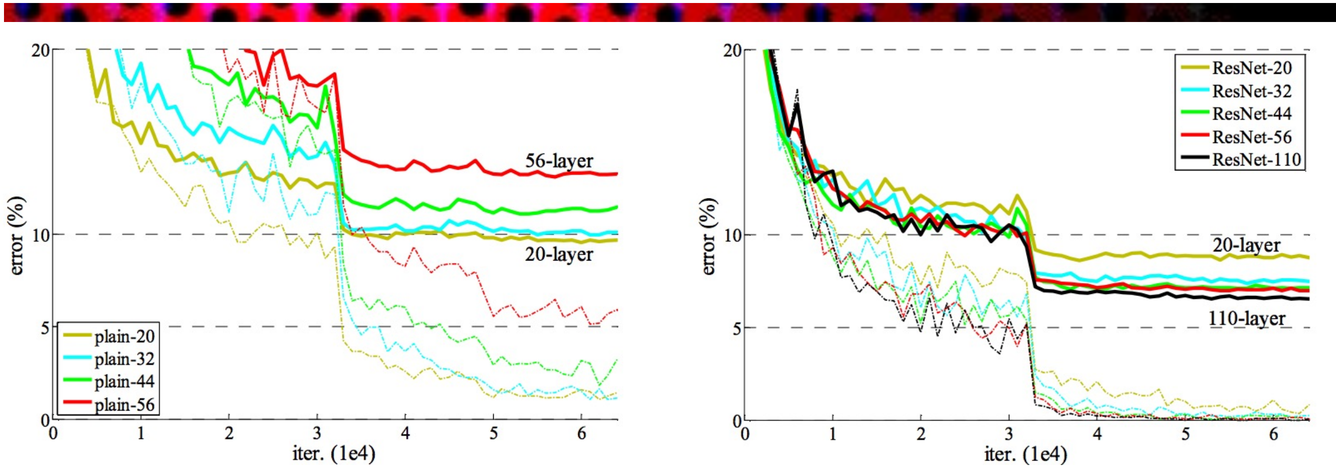
Bottleneck layer

Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a “bottleneck” building block for ResNet-50/101/152.

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. http://openaccess.thecvf.com/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf



Benefit of residual connection

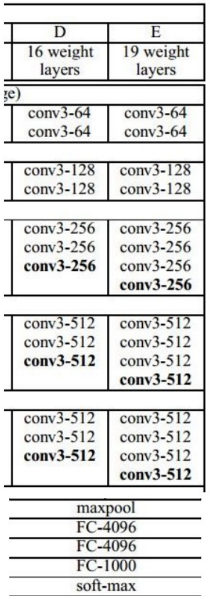


W/o residual layer: deeper networks perform worse even for the training data.
W/ residual layer: deeper networks perform better!

Using shortcut 2 is theoretically optimal
Demystifying ResNet
https://arxiv.org/abs/1611.01186

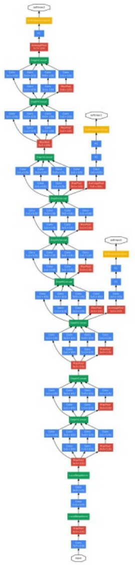
Revolution of Depth

Case Studies



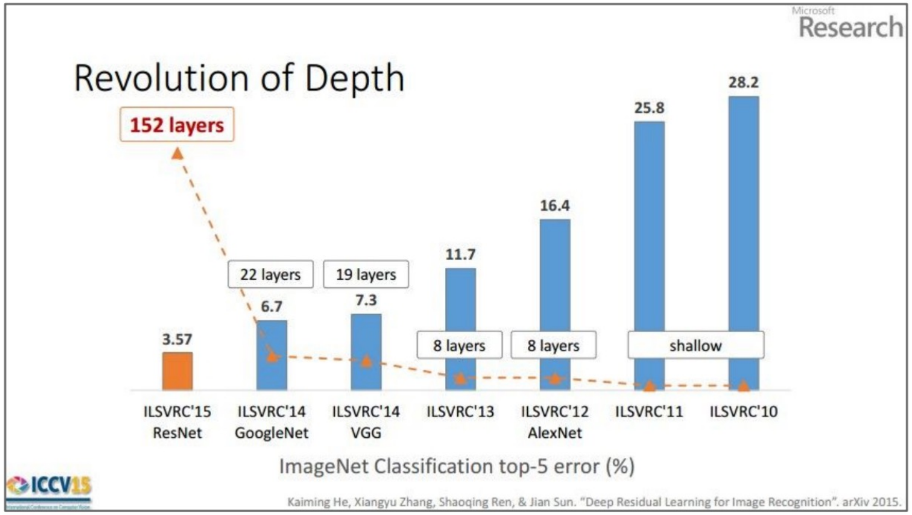
D	E
16 weight layers	19 weight layers
ge)	
conv3-64 conv3-64	conv3-64 conv3-64
conv3-128 conv3-128	conv3-128 conv3-128
conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool	
FC-4096	
FC-4096	
FC-1000	
soft-max	

VGG (2014)



GoogLeNet (2014)

ResNet (2015)



Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

From: http://cs231n.stanford.edu/slides/2016/winter1516_lecture8.pdf

A variation of residual connection: Concatenation (DenseNet)

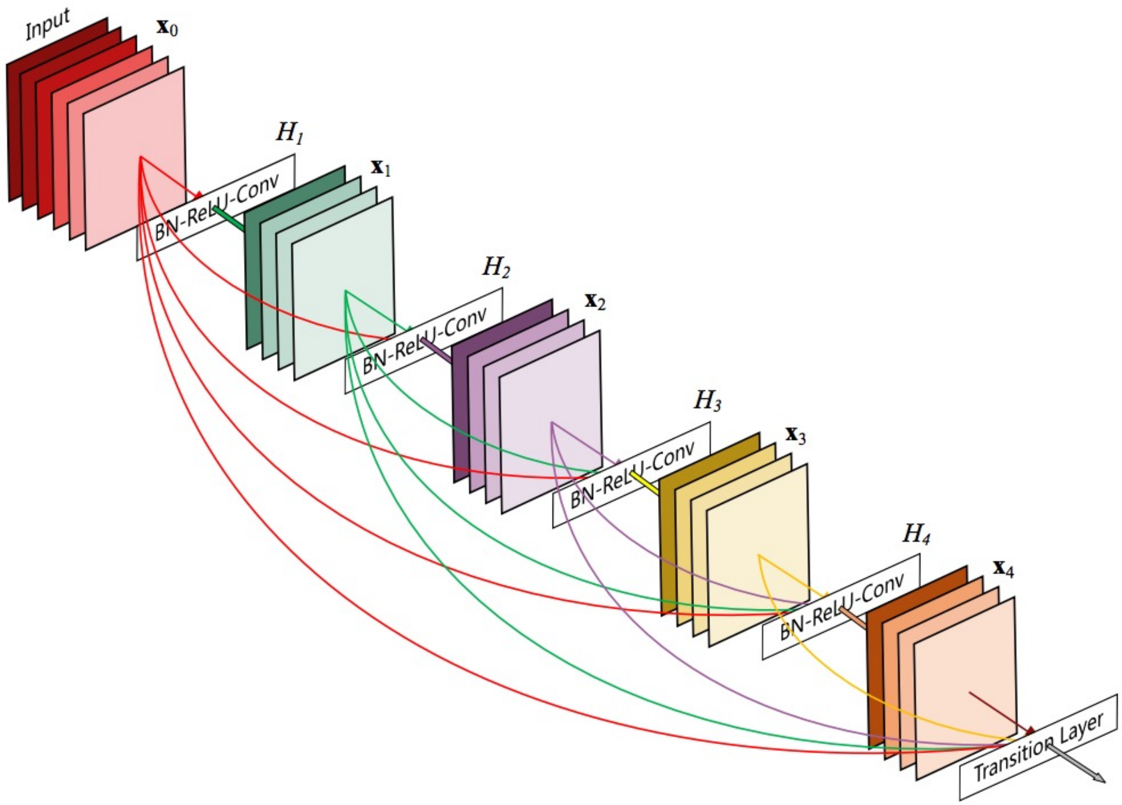


Figure 1: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.

- Feature maps of all preceding layers are concatenated and used as input for the current layer.
- Facilitate gradient back propagation, as with residual connection
- Strengthen feature forward propagation and reuse

From: Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700-4708. 2017.

Stacking Dense Blocks

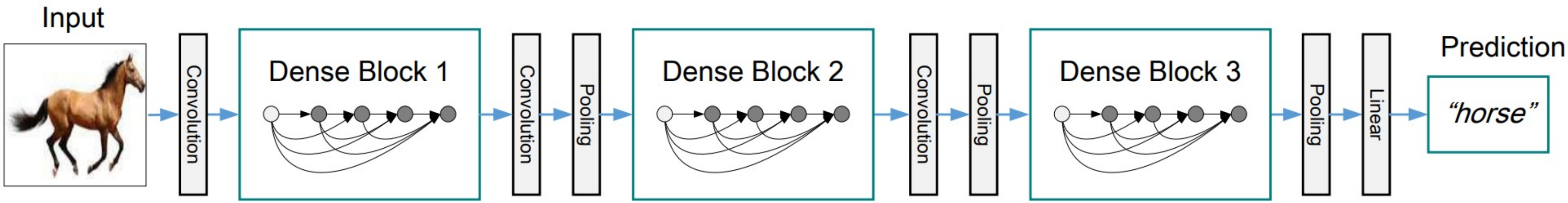
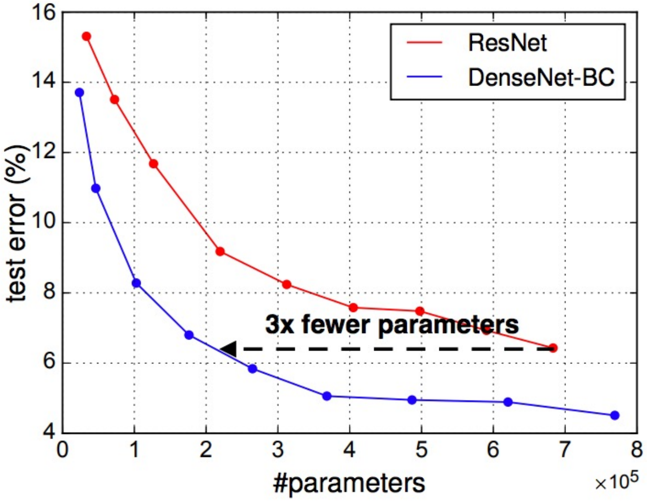


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

Use bottleneck layer (1x1 conv) to reduce the number of feature maps between blocks

- Can use fewer layers to achieve same performance as ResNET



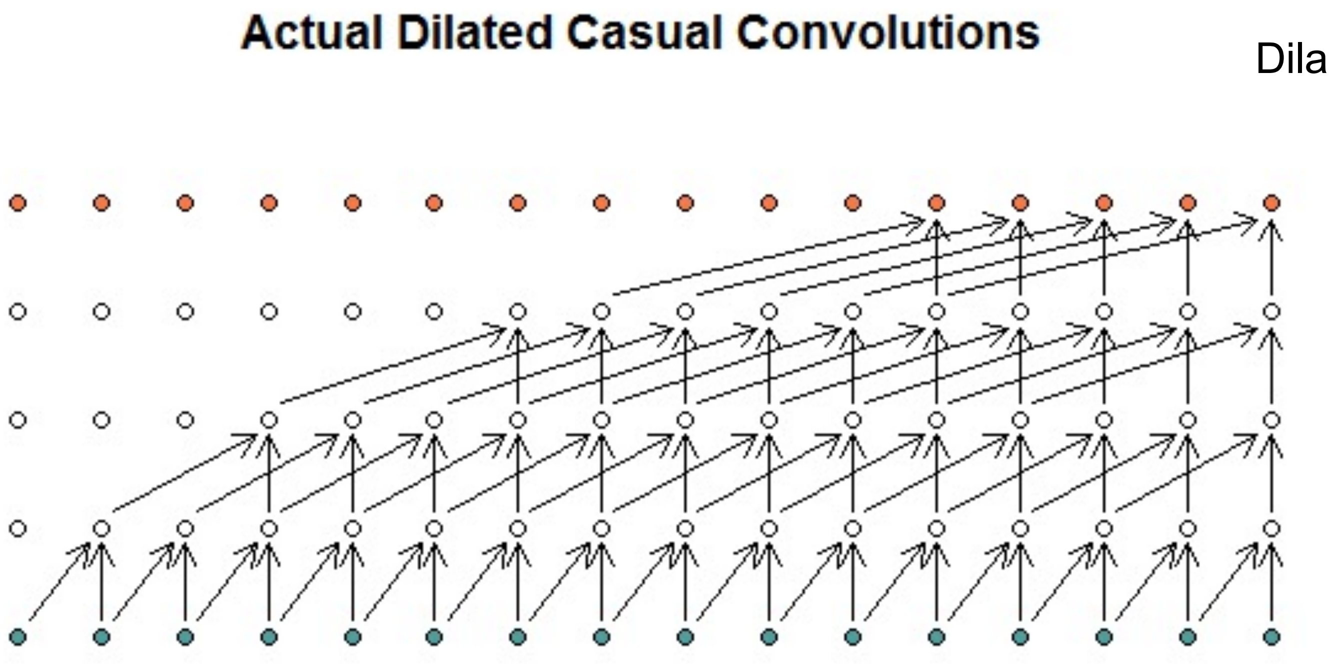
From: Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700-4708. 2017.

Dilated Convolution

- Large receptive field is important to incorporate global information
- How to increase the receptive field
  - Larger filter
  - More layers of small filters
  - Dilated conv.

Figure from Fergus: https://cs.nyu.edu/~fergus/teaching/vision/3_convnets.pdf

Dilated Conv in 1D

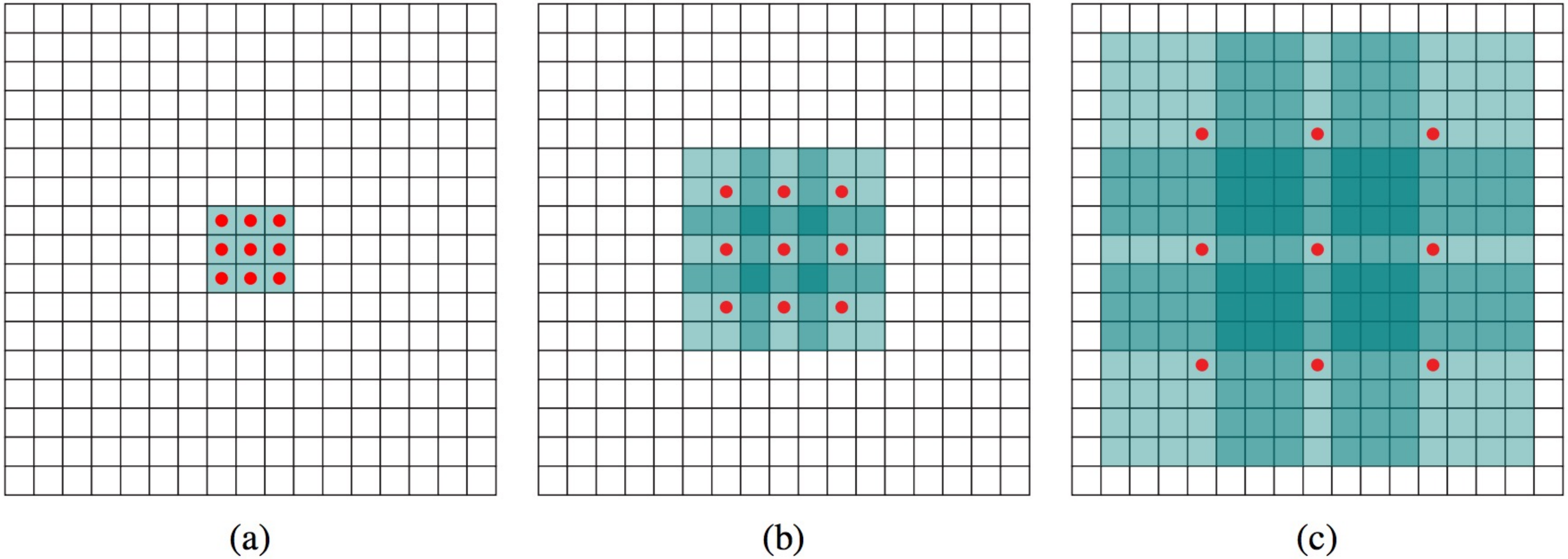


Dilation rate	Receptive field
4	11
3	7
2	4
1	2

Figure from https://i.stack.imgur.com/RmJSu.png

Multiscale processing while maintaining original resolution!
Used for speech waveform generation (WaveNet).

Dilated Conv. In 2D


(a)

(b)

(c)

Figure 1: Systematic dilation supports exponential expansion of the receptive field without loss of resolution or coverage. (a) F_1 is produced from F_0 by a 1-dilated convolution; each element in F_1 has a receptive field of 3×3. (b) F_2 is produced from F_1 by a 2-dilated convolution; each element in F_2 has a receptive field of 7×7. (c) F_3 is produced from F_2 by a 4-dilated convolution; each element in F_3 has a receptive field of 15×15. The number of parameters associated with each layer is identical. The receptive field grows exponentially while the number of parameters grows linearly.

Yu, Fisher, and Vladlen Koltun. "Multi-scale context aggregation by dilated convolutions." *arXiv preprint arXiv:1511.07122* (2015).

Multiscale processing while maintaining original resolution!
Good for dense prediction: image to image

Pop Quizzes

- What is the benefit of residual connection?
- What is dense connection?
- What is the benefit of dilated convolution?

Pop Quizzes

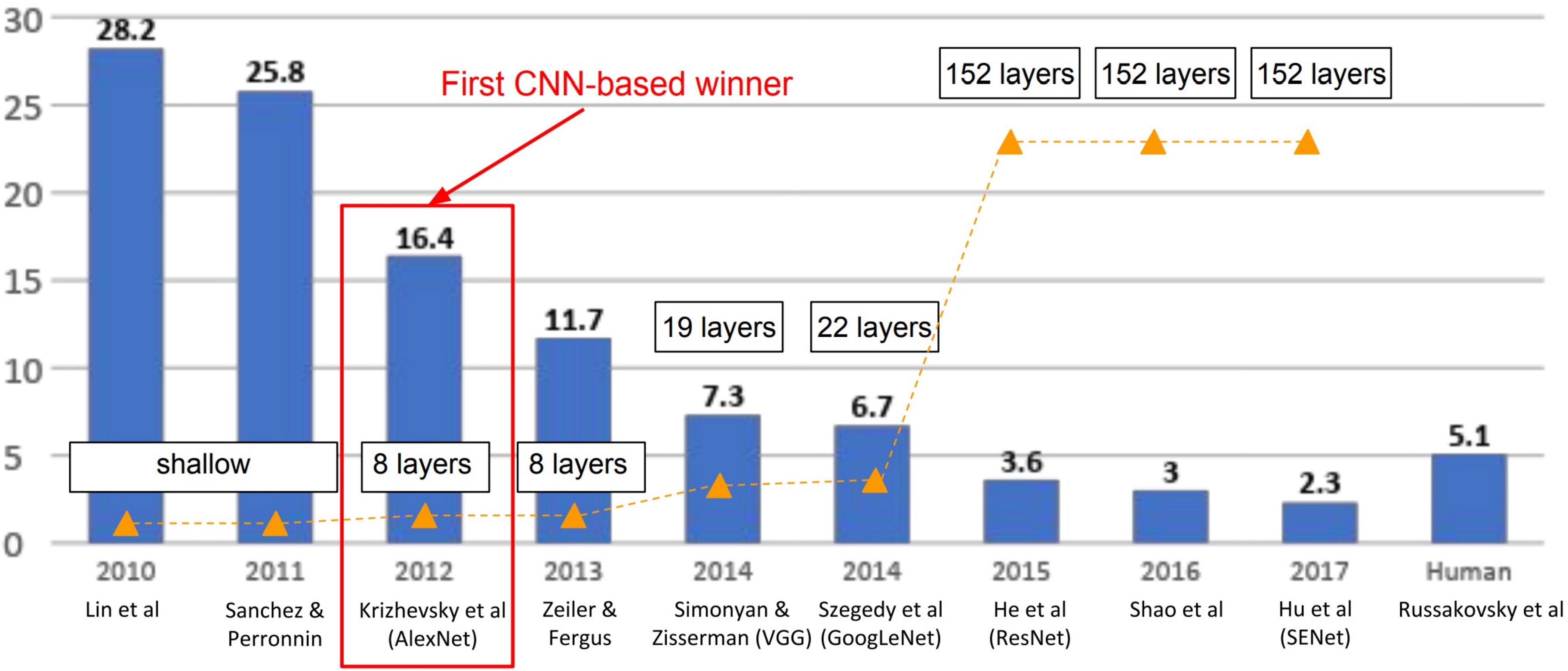
- What is the benefit of residual connection?
  - Enable gradient backpropagation
  - Each layer learn the residual from the previous layer
  - Critical for deep networks
- What is dense connection?
  - Use multiple skip connections, also facilitate gradient backpropagation
  - Concatenating output from past layers instead of using addition
  - Enable feature reuse
- What is the benefit of dilated convolution?
  - Obtain large receptive field w/o downsampling

Outline (Part II)

- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

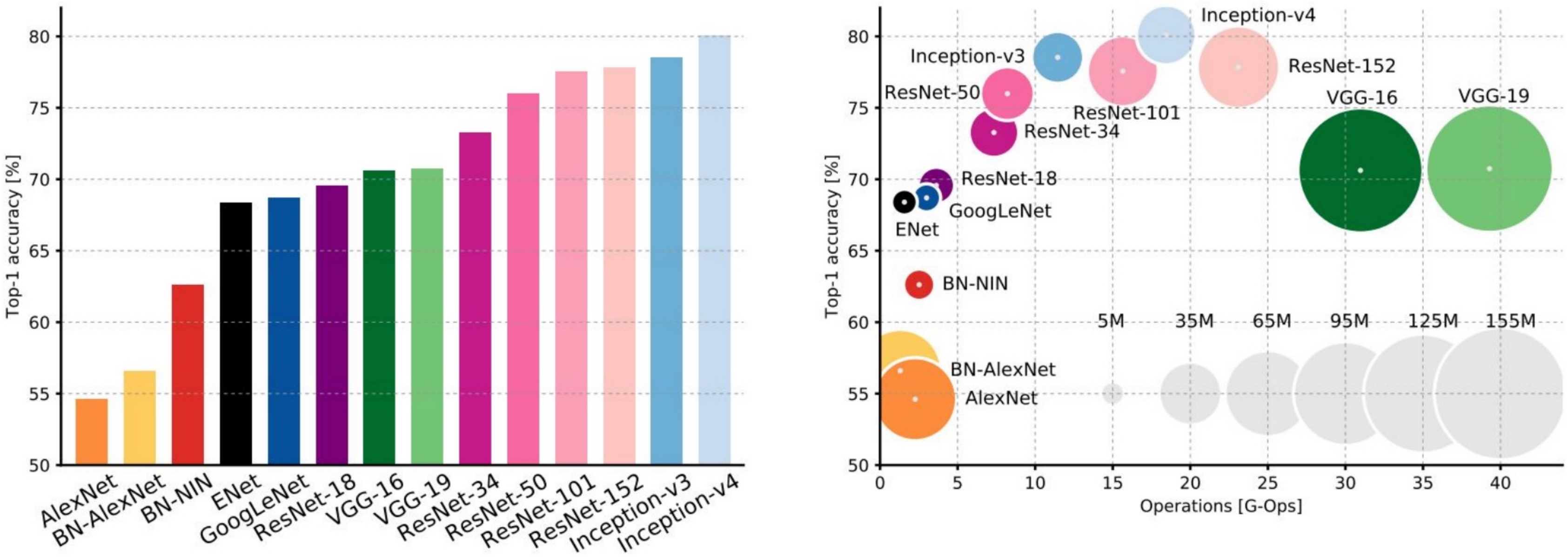
Well-Known Models

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture09.pdf

Performance vs. Complexity



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

http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture09.pdf

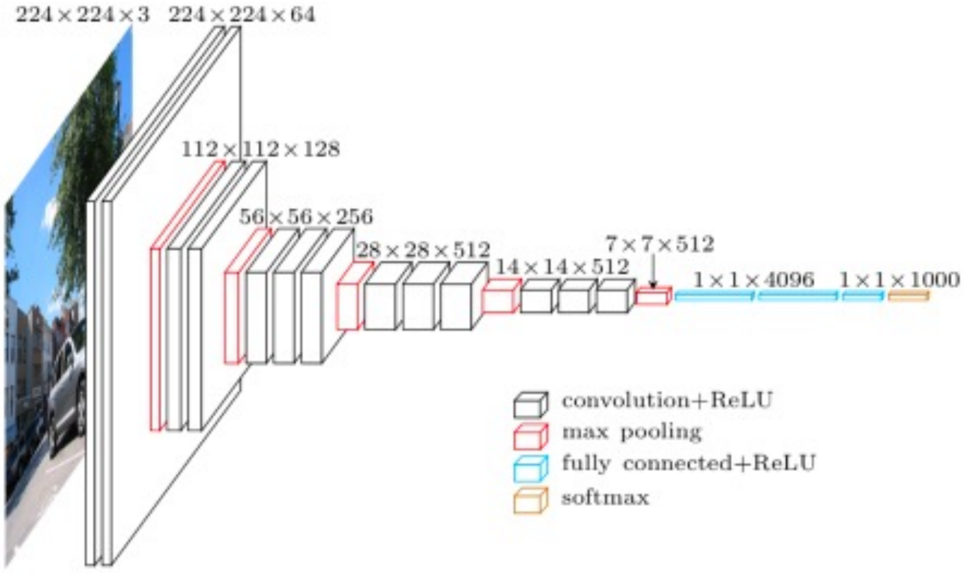
Transfer Learning

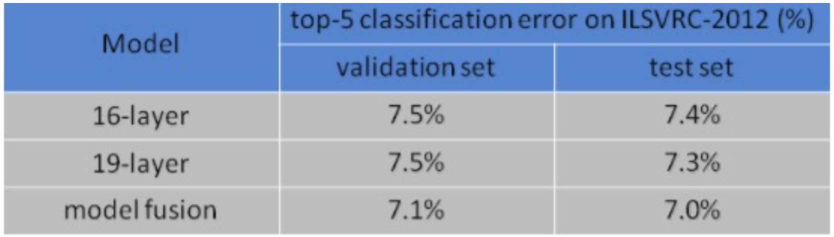
- For image classification or other applications, training from scratch takes tremendous resources and requires a lot of training data
- Instead, can refine the VGG or other well trained networks
- Can use VGG convolutional layers, and retrain only the fully connected layers (possibly some later convolutional layers) for different classification problems.
- Or can use VGG conv layers as the "initial model" and further refine.
- Computer Assignment (optional): load VGG model, and fix all conv. layers, retrain additional fully connected layers for different image classification tasks, try and compare different training tricks
  - Using Flickr API (courtesy of Sundeep Rangan) for downloading images for a given keyword

VGG16

- From the Visual Geometry Group
  - Oxford, UK

- Won ImageNet ILSVRC-2014
- Remains a very good network
- Lower layers are often used as feature extraction layers for other tasks





Model	top-5 classification error on ILSVRC-2012 (%)	
	validation set	test set
16-layer	7.5%	7.4%
19-layer	7.5%	7.3%
model fusion	7.1%	7.0%

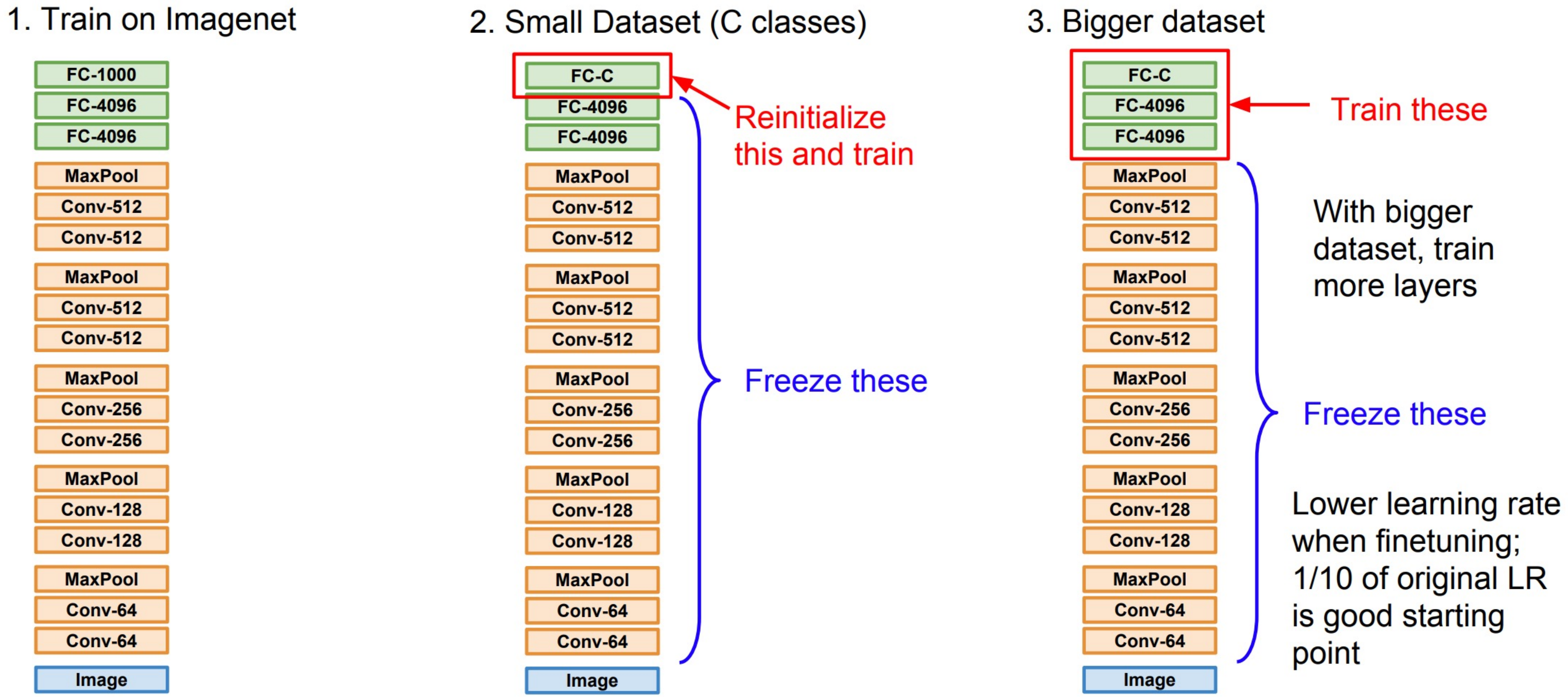
http://www.robots.ox.ac.uk/~vgg/research/very_deep/

K. Simonyan, A. Zisserman
Very Deep Convolutional Networks for Large-Scale Image Recognition
arXiv technical report, 2014

Transfer Learning

Transfer Learning with CNNs

Donahue et al, “DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition”, ICML 2014
Razavian et al, “CNN Features Off-the-Shelf: An Astounding Baseline for Recognition”, CVPR Workshops 2014



Reinitialize this and train

Freeze these

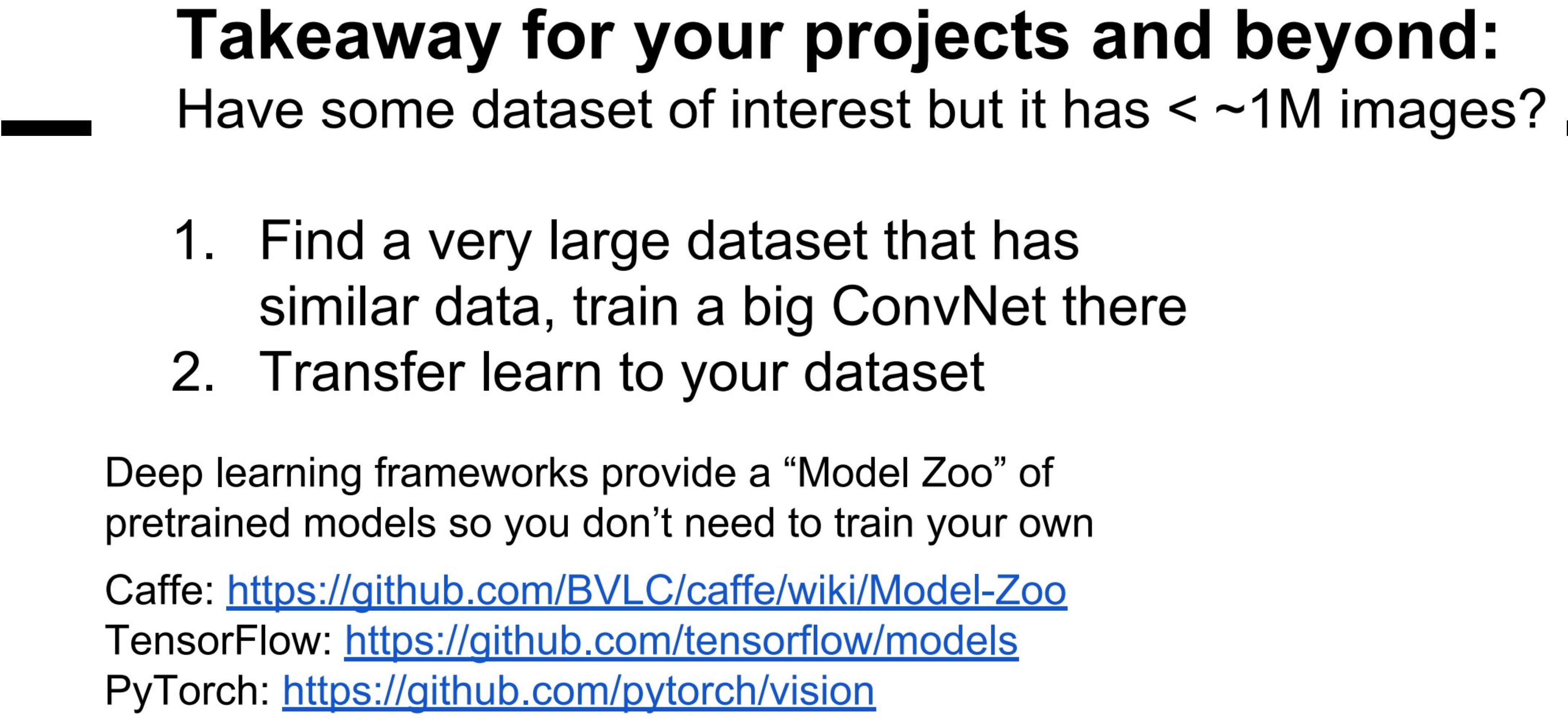
Train these

With bigger dataset, train more layers

Freeze these

Lower learning rate when finetuning; 1/10 of original LR is good starting point

From http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture07.pdf



Takeaway for your projects and beyond:

Have some dataset of interest but it has < ~1M images?

1. Find a very large dataset that has similar data, train a big ConvNet there
2. Transfer learn to your dataset

Deep learning frameworks provide a “Model Zoo” of pretrained models so you don’t need to train your own

Caffe: https://github.com/BVLC/caffe/wiki/Model-Zoo
TensorFlow: https://github.com/tensorflow/models
PyTorch: https://github.com/pytorch/vision

From http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture07.pdf

Pop quizzes

- What does transfer learning mean?

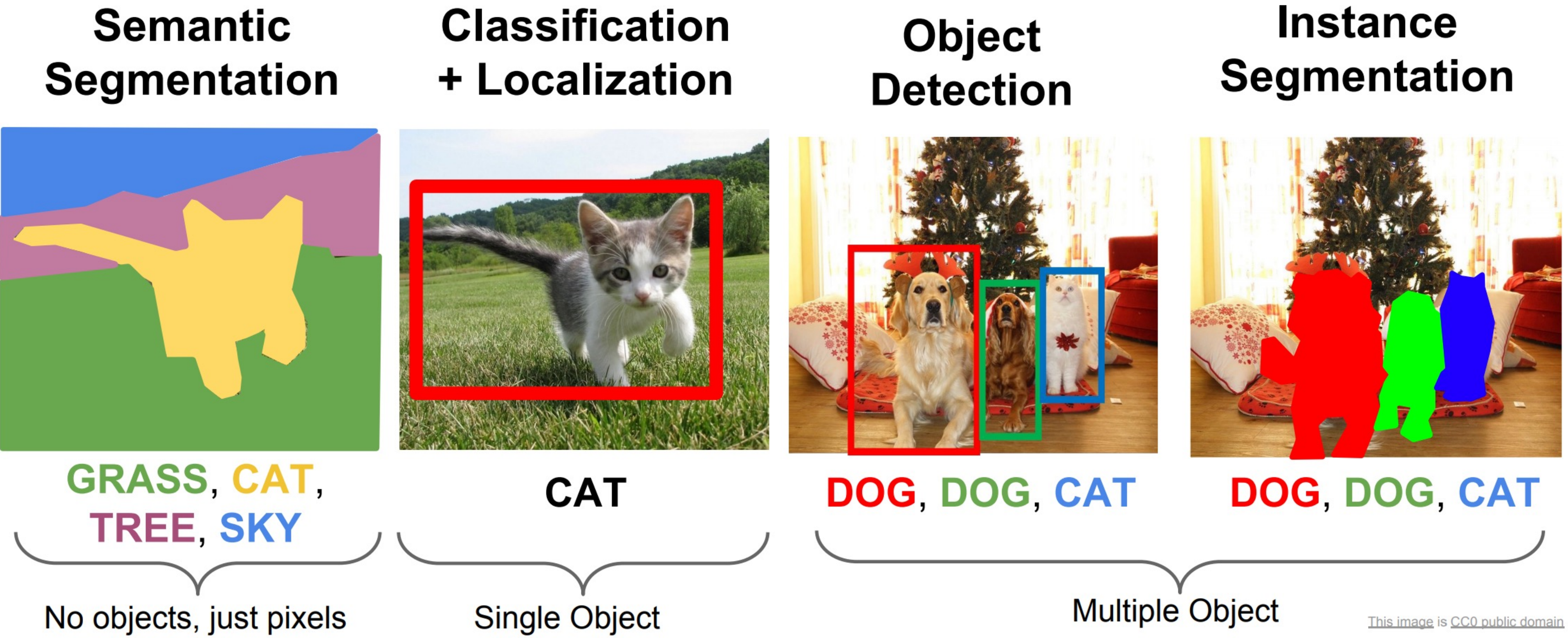
Pop quizzes

- What does transfer learning mean?
  - Take a popular well trained model for a different task but with the same type of input (e.g. images)
  - Reuse some of the feature extraction layers, only train the fully connected layers
  - Or also refine some of the convolution layers for feature extraction, depending on available training samples

Outline (Part II)

- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

Beyond Image Classification …



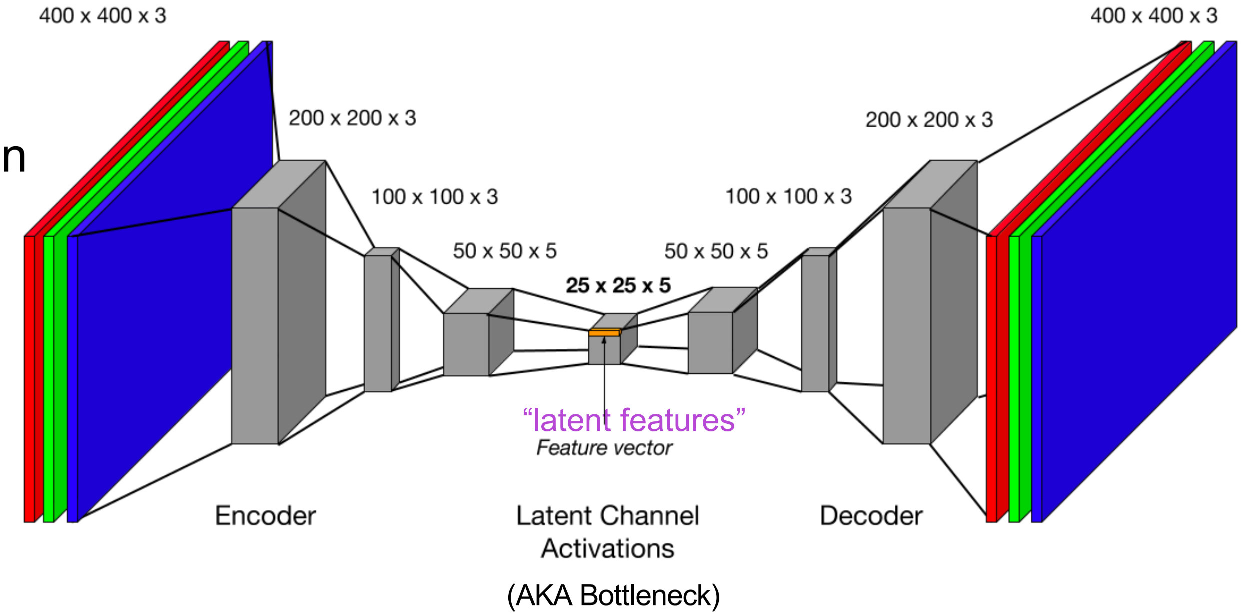
From: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture11.pdf

Image to Image Autoencoder

- Denoising and other applications
- Upsampling through learnable filters

Autoencoder

- CNN is not limited for classification!
- When all the layers are convolution, the output can have the same shape as the input (speech->speech, image->image)
- Autoencoder= Encoder+Decoder
- Encoder: image-> features
- Decoder: features -> image
- Fully convolutional network (FCN)

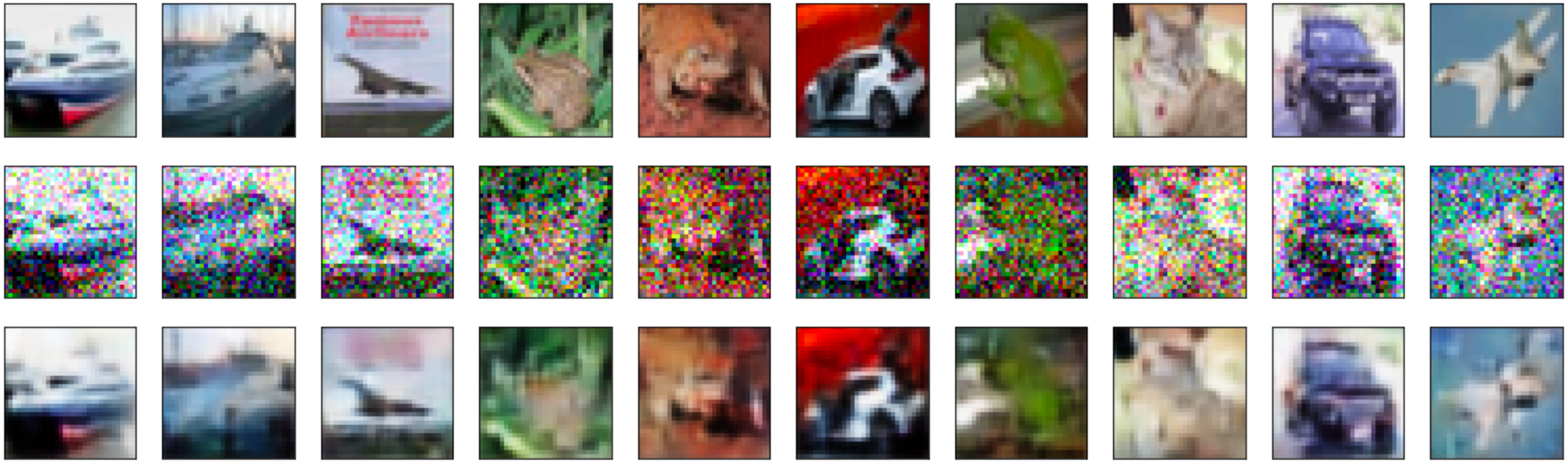


From http://warp.whoi.edu/content/images/2017/08/caearch_shallow.png

Can be applied at the whole image level, or (overlapped) block level.

Autoencoder for image denoising

- Input: noisy image; Output= denoised image
- Need pairs of clean and noisy images as training samples, normalized to range (0,1)
- Following from a simple network (with only three conv layers in encoder and two conv layers in decoder)

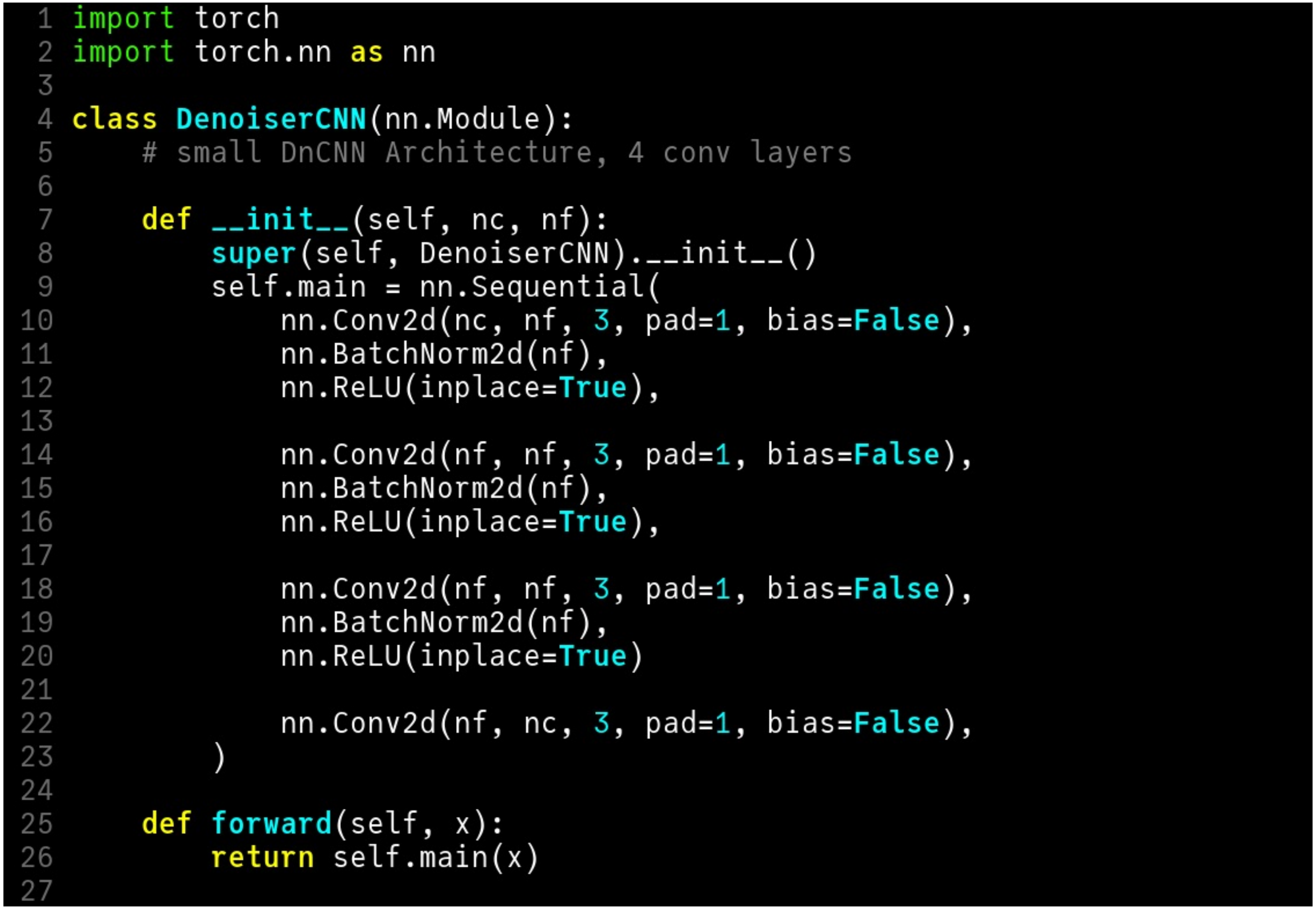


- Better image denoising networks:

  Zhang, Kai, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising." *IEEE Transactions on Image Processing* 26, no. 7 (2017): 3142-3155.

  …

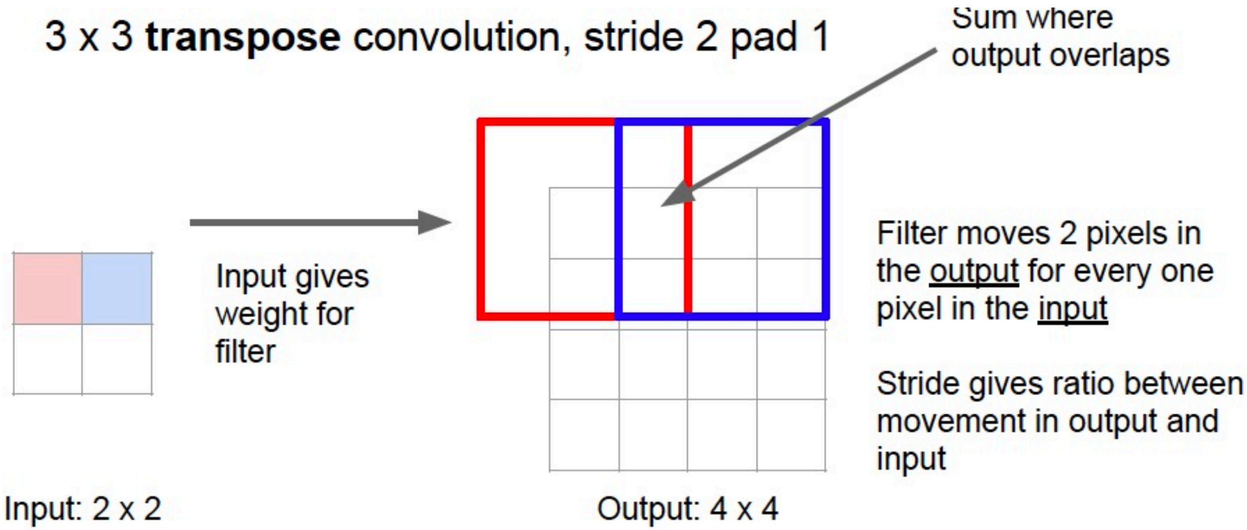
Sample PyTorch implementation for denoising



```python
import torch
import torch.nn as nn

class DenoiserCNN(nn.Module):
    # small DnCNN Architecture, 4 conv layers

    def __init__(self, nc, nf):
        super(self, DenoiserCNN).__init__()
        self.main = nn.Sequential(
            nn.Conv2d(nc, nf, 3, pad=1, bias=False),
            nn.BatchNorm2d(nf),
            nn.ReLU(inplace=True),

            nn.Conv2d(nf, nf, 3, pad=1, bias=False),
            nn.BatchNorm2d(nf),
            nn.ReLU(inplace=True),

            nn.Conv2d(nf, nf, 3, pad=1, bias=False),
            nn.BatchNorm2d(nf),
            nn.ReLU(inplace=True)

            nn.Conv2d(nf, nc, 3, pad=1, bias=False),
        )

    def forward(self, x):
        return self.main(x)

```

How to perform upsampling?

- Using default upsampling filter (nearest, linear)
- Learn the interpolation filter (transposed convolution or deconvolution)
  - First generate zero-filled image (inserting zeros between known samples)
  - Then apply the filter



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

Sum where output overlaps

Input gives weight for filter

Filter moves 2 pixels in the <u>output</u> for every one pixel in the <u>input</u>

Stride gives ratio between movement in output and input

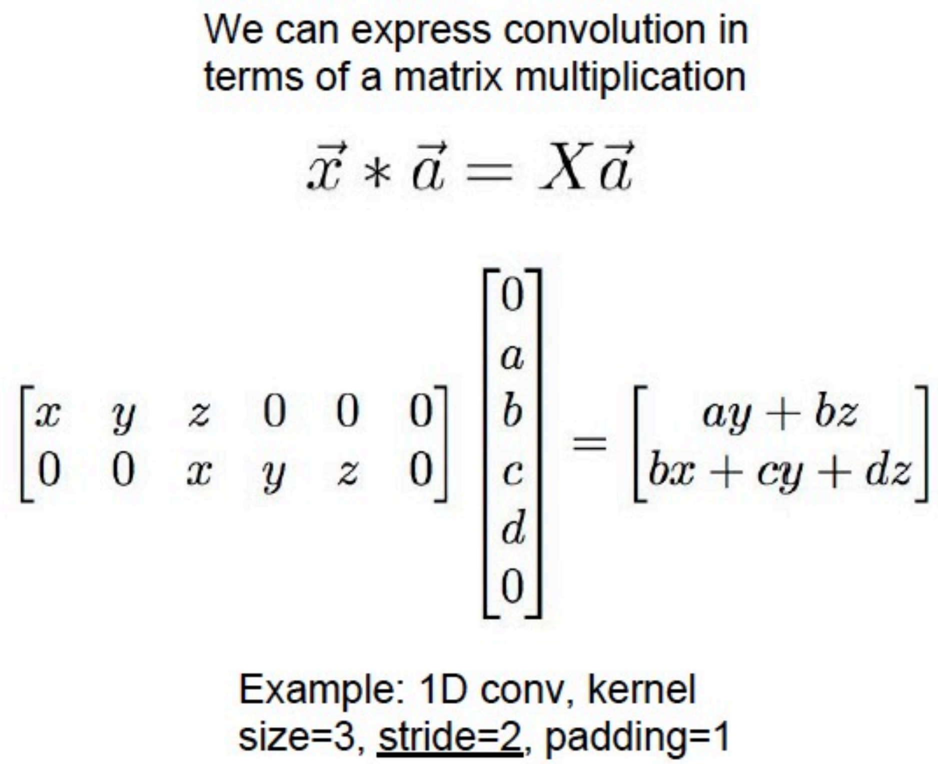
Input: 2 x 2

Output: 4 x 4

From http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture11.pdf

Also known as “Deconvolution” (not a proper name!)

Why “Transpose Convolution”?

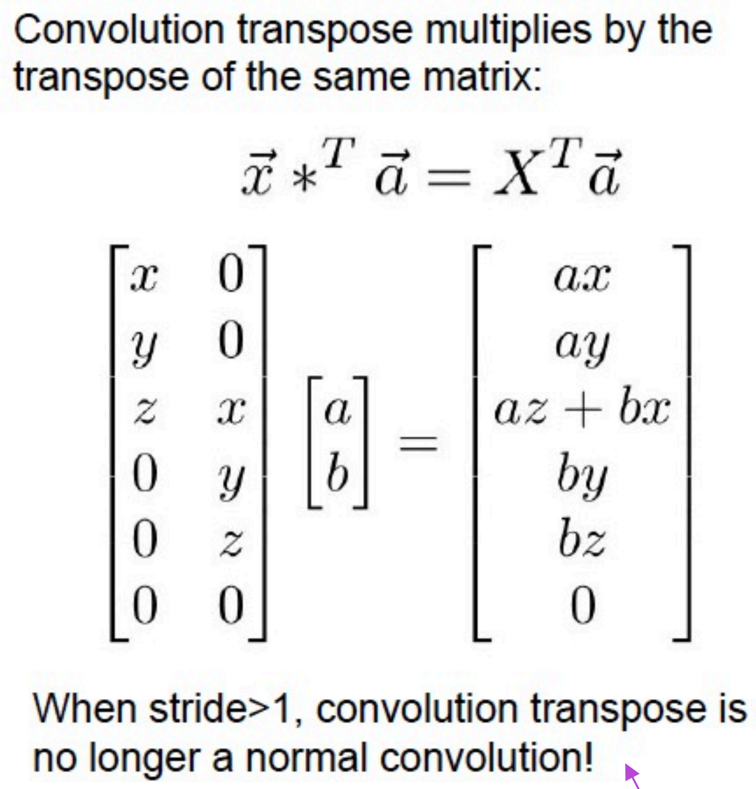


We can express convolution in terms of a matrix multiplication

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

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

Example: 1D conv, kernel size=3, <u>stride=2</u>, 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!

From http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture11.pdf

Actually it is a convolution, need to pad zero in between “a” and “b”

DSP explanation of transpose convolution

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

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

No zero filling
Stride 2

X corresponding to filter [x,y,z]
X^T corresponds to reversed filter [z,y,x]

Insert zeros before convolution

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

Stride K: upsampling by factor of K, insert K-1 zeros in between every original samples
This is exactly how we perform interpolation in DSP!
Interpolation filter is [z,y,x]

Applications of autoencoders

- Output \= input:
  - Denoising
  - Image completion (filling missing parts)
  - Super resolution (output dimension larger than input)
  - Segmentation
  - Visual Saliency detection
- Output = input
  - For unsupervised learning: to learn features that can represent an image with reduced dimension
  - For compression: use the quantized latent features to represent an image
- Autoencoder loss depends on the underlying application
- Using adversarial loss can help to make the output have the same distribution as the target output (beyond this class)

Pop Quizzes

- What is a fully convolutional network (FCN)? What are they used for?
- What is an auto-encoder?
- What is the benefit of using down sampling in the encoder?
- How do we upsample in the decoder?

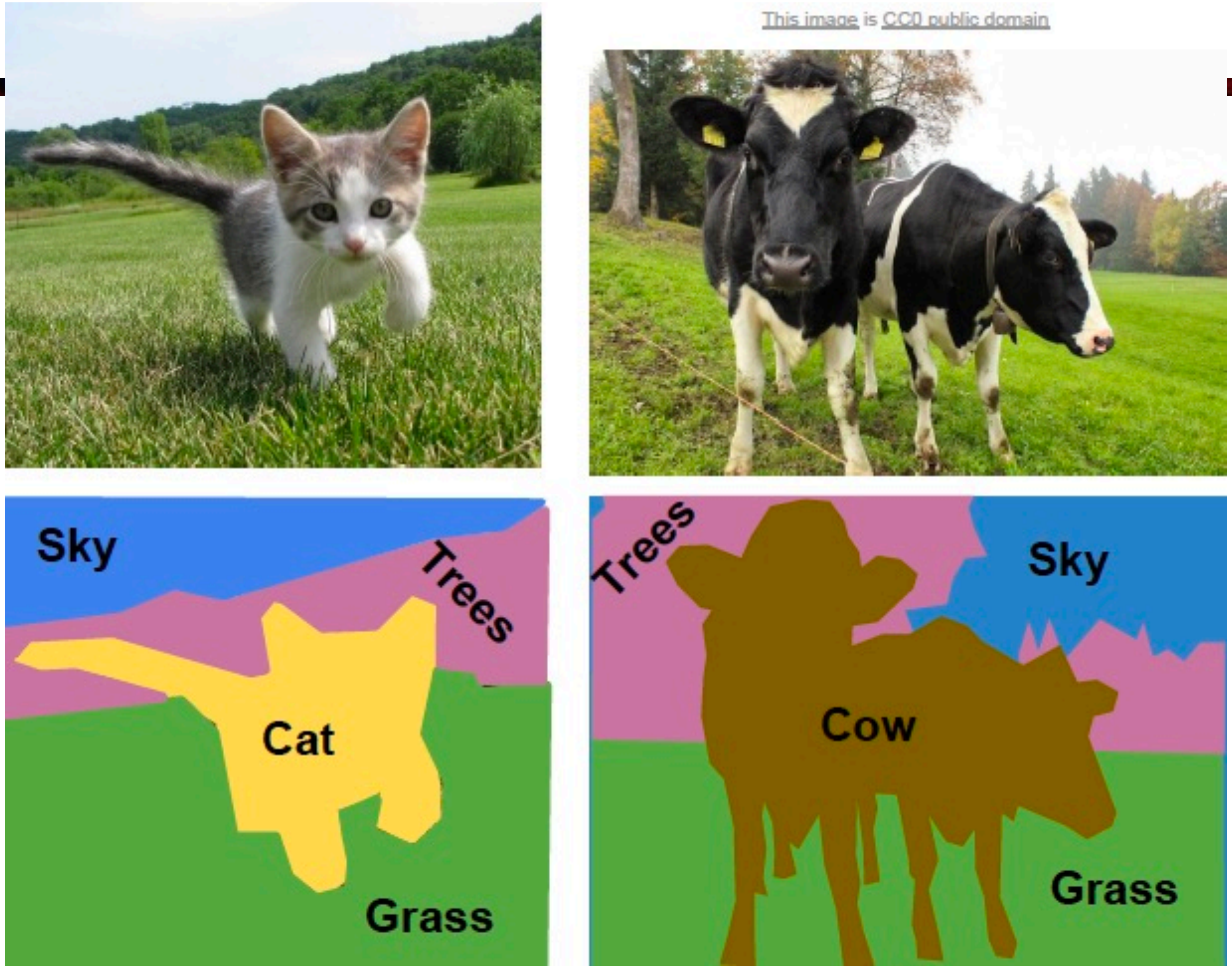
Pop Quizzes

- What is a fully convolutional network (FCN)? What are they used for?
  - Only convolutional layers
  - Used for mapping an input image to an output image
- What is an auto-encoder?
  - Encoder generate multi-channel downsampled features
  - Decoder reconstruct an image by upsample the features and additional convolution
  - Down-sampling / Up-sampling is not necessary
- What is the benefit of using down sampling in the encoder?
  - Enlarge receptive field with fewer layers to enable more efficient gathering of global information
  - Produce more compact latent features
- How do we upsample in the decoder?
  - Using transposed convolution, with fixed or learnable interpolation filters

Outline (Part II)

- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

Semantic Segmentation


This image is CC0 public domain

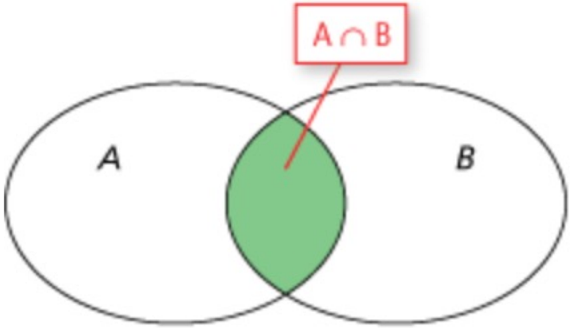
Each pixel is classified into one object class. Same type of object has the same color

[From http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture11.pdf](http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture11.pdf)

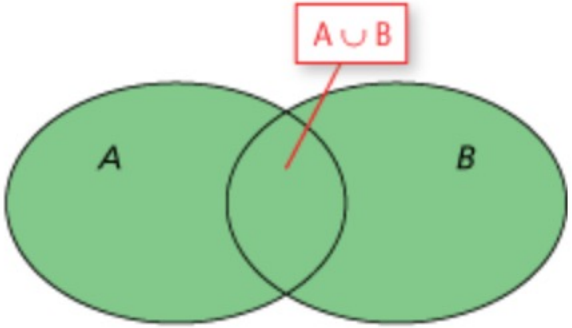
Loss function for segmentation

- Semantic segmentation:
  - Label each pixel as one of the classes
  - Treat as multi-class classification at each pixel
    - Generate multiple segmentation maps as output, one probability map for each class. The probabilities for all classes at each pixel sum to 1
  - Loss:
    - Sum of categorical cross entropy over all pixels for each image, and over all training images
    - DICE = Intersection over union (Non-differentiable)
    - Soft DICE: defined in terms of predicted probability

DICE Loss for Evaluating Binary Segmentation



Intersection of *A* and *B*



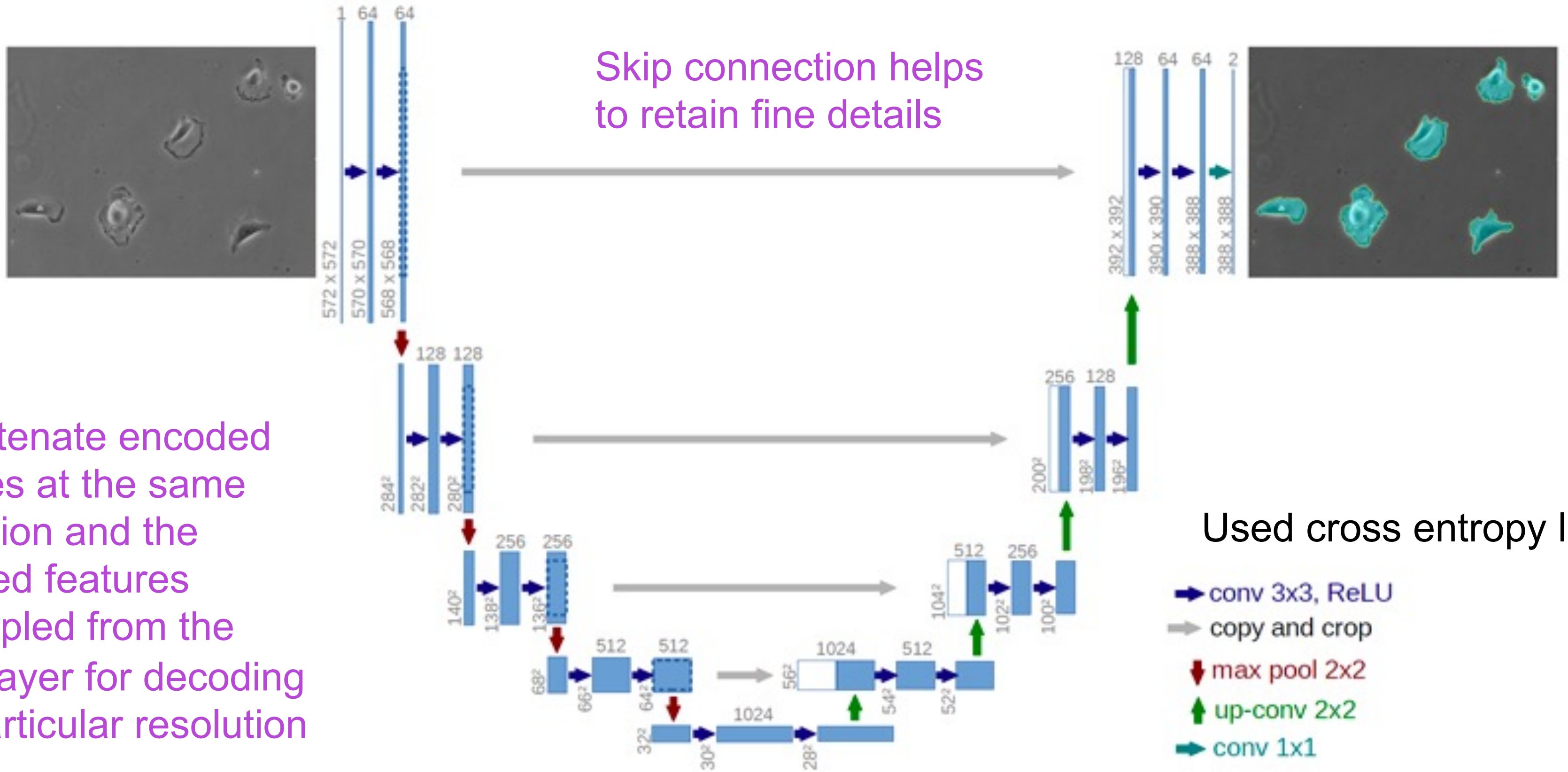
Union of *A* and *B*

A: Ground truth foreground,
g_i=1

B: predicted foreground,
p_i: probability of pixel i is foreground

- DICE = $\frac{\text{area of } A \cap B}{\text{area of } A \cup B}$ (Intersection over union or IoU, evaluating after thresholding the probability, non-differentiable)
- Soft DICE = $\frac{2 \sum_i p_i g_i}{\sum_i p_i^2 + \sum_i g_i^2}$ (differentiable)
  - (numerator ~ intersection, denominator ~ union)
  - maximize Soft DICE. Loss = - Soft DICE
- Overcomes the difficulty of using cross entropy loss when the foreground object is much smaller than the background

Multi-resolution auto encoder: U-Net



Skip connection helps to retain fine details

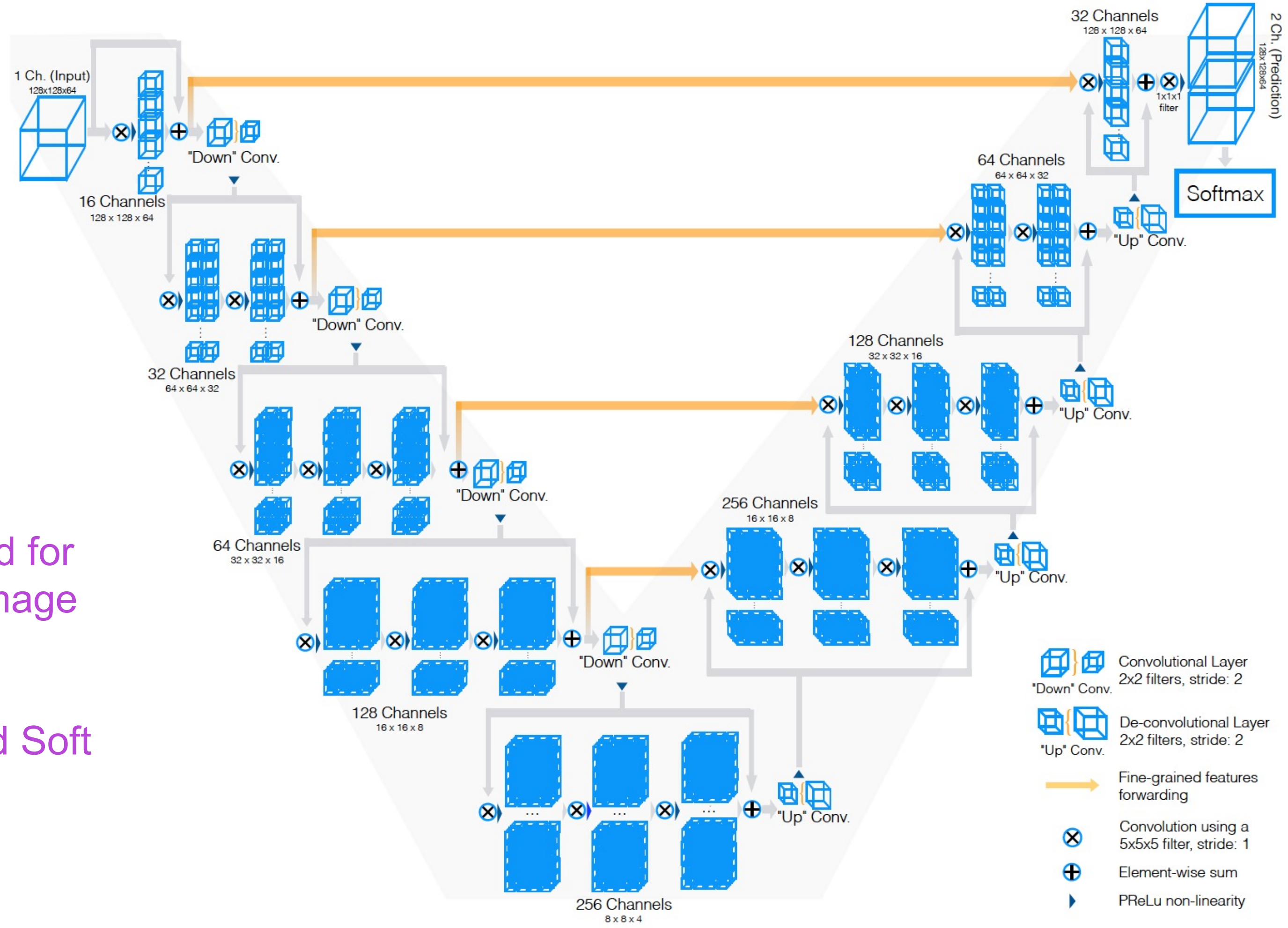
Concatenate encoded features at the same resolution and the decoded features upsampled from the lower layer for decoding at a particular resolution

Used cross entropy loss

From: https://lmb.informatik.uni-freiburg.de/research/funded_projects/bioss_deeplearning/unet.png

Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham. https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/. Watch the video!

V-Net for Volumetric Image Segmentation



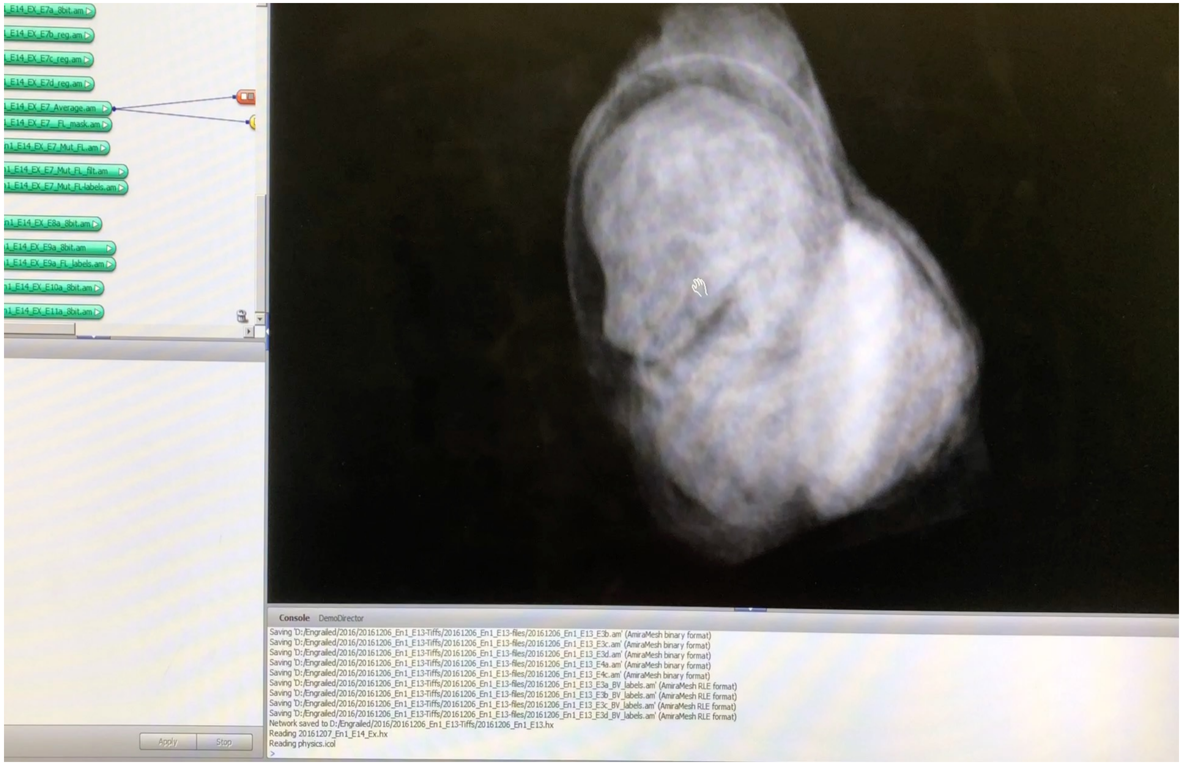
Often used for medical image volume

Introduced Soft DICE loss

Milletari, Fausto, Nassir Navab, and Seyed-Ahmad Ahmadi. "V-net: Fully convolutional neural networks for volumetric medical image segmentation." *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016. https://arxiv.org/pdf/1606.04797.pdf

Work at NYU Video Lab: Segmentation of High Frequency Ultrasound Images of Mouse Embryos

- Mouse embryo development is a good animal model for studying human brain development.
- High frequency ultrasound (HFU) can image embryos in vivo. *But image quality is poor and pose of the embryo varies greatly.*
- Accurate segmentation of the brain ventricles (BVs) and embryo body from 3D HFU image is essential for assessing the development of mouse embryos and impact of gene mutation.

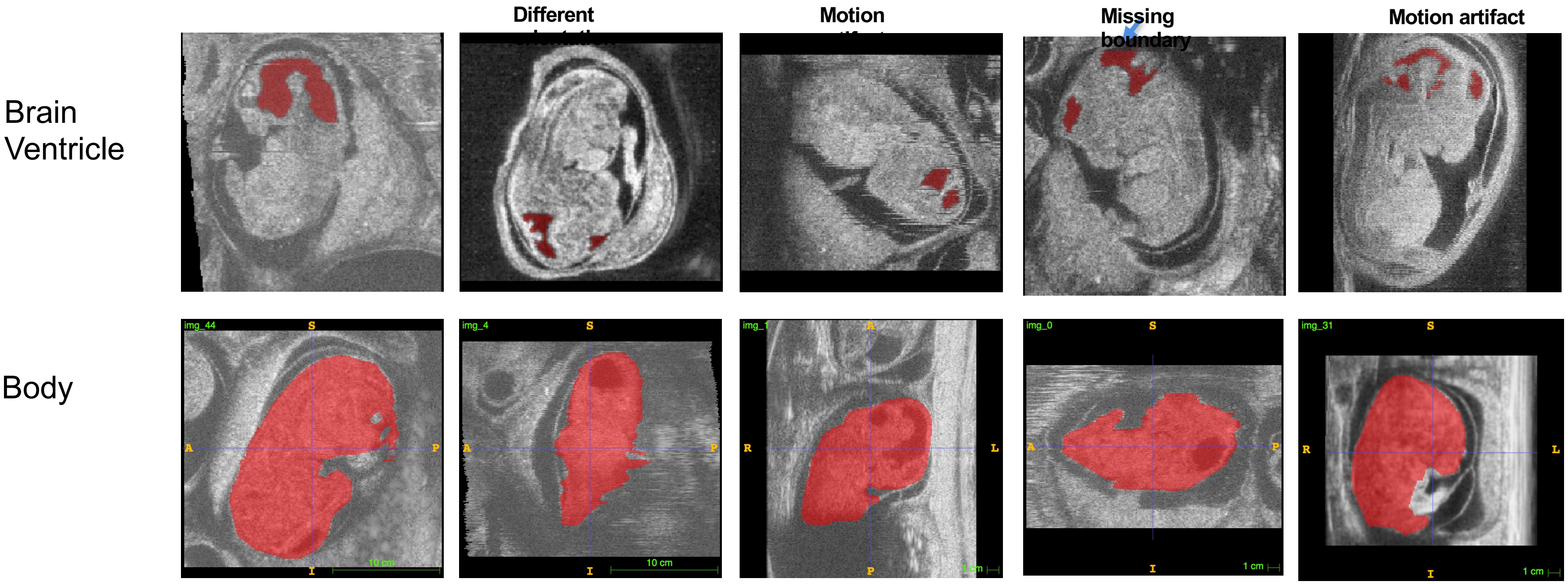


```
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3b.am' (AmiraMesh binary format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3c.am' (AmiraMesh binary format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3d.am' (AmiraMesh binary format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E4a.am' (AmiraMesh binary format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E4c.am' (AmiraMesh binary format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3a_BV_labels.am' (AmiraMesh RLE format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3b_BV_labels.am' (AmiraMesh RLE format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3c_BV_labels.am' (AmiraMesh RLE format)
Saving 'D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13-files/20161206_En1_E13_E3d_BV_labels.am' (AmiraMesh RLE format)
Network saved to D:/Engrailed/2016/20161206_En1_E13-Tiffs/20161206_En1_E13.hx
Reading 20161207_En1_E14_Ex.hx
Reading physics.icol
>
```

Joint work with Riverside Research and NYU SOM. Supported by NIH R01. Video lab students: Ziming Qiu (PhD 2022), Tongda Xu, Jack Langerman, Nitin Nair

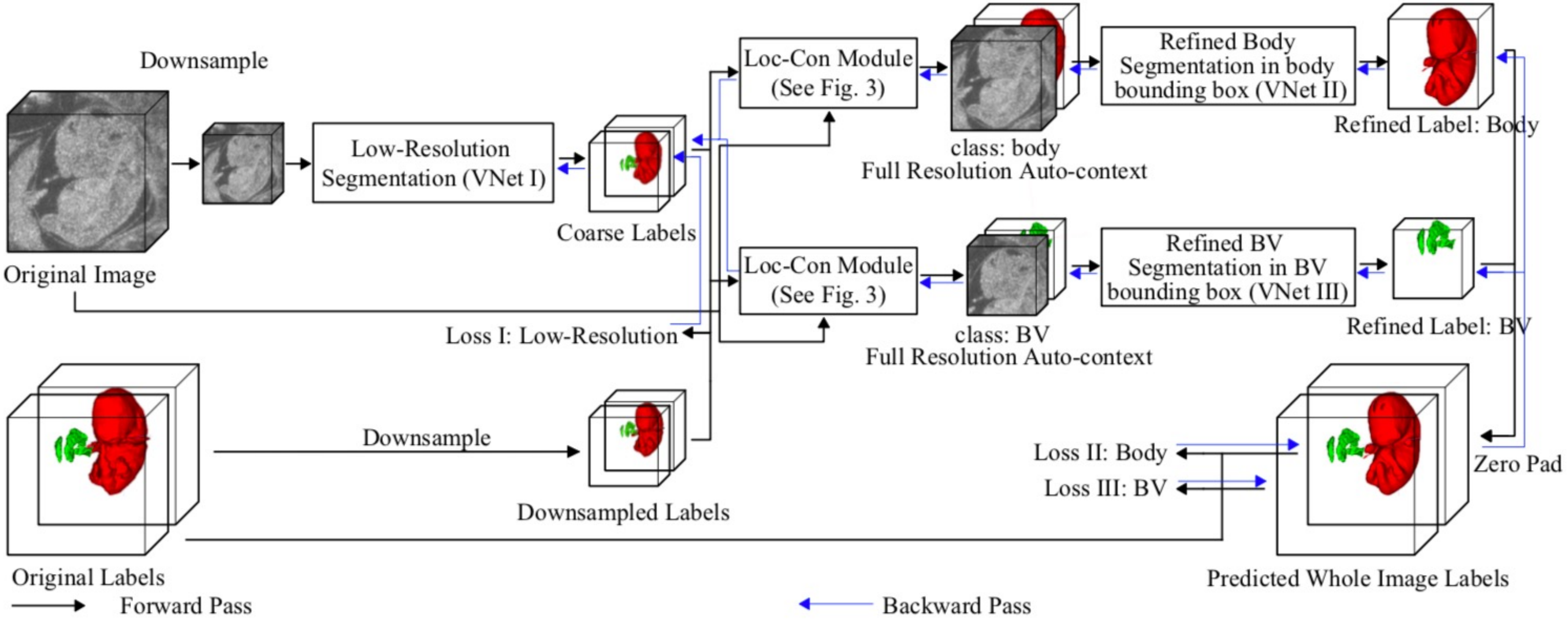
Challenges

- The variety of body posture, orientation and image contrast.
- The class imbalance between foreground and background.
- Presence of missing boundaries and motion artifacts.



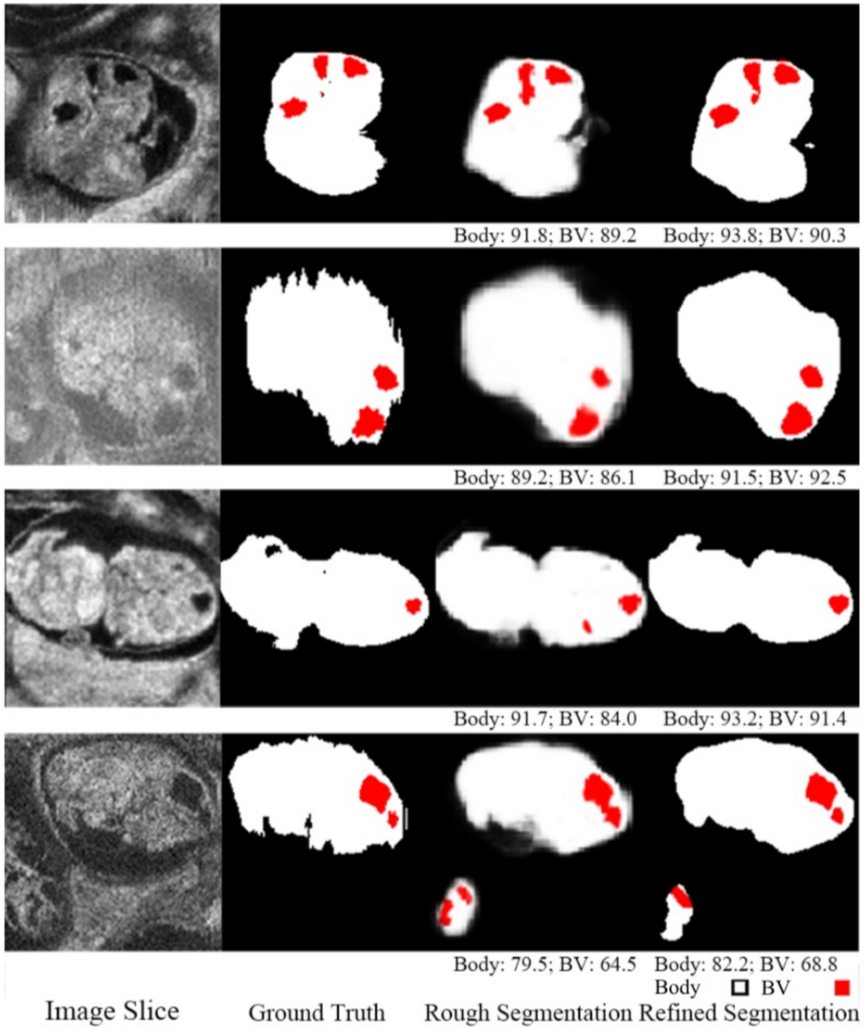
Deep-learning based segmentation

- ❑ Low resolution coarse segmentation to localize regions of interest for both BV and Body.
- ❑ Full resolution refined segmentation in each detected region of interest.
- ❑ Jointly trained end to end.



Downsample
Original Image
Low-Resolution Segmentation (VNet I)
Coarse Labels
Loc-Con Module (See Fig. 3)
class: body
Full Resolution Auto-context
Refined Body Segmentation in body bounding box (VNet II)
Refined Label: Body
Loc-Con Module (See Fig. 3)
class: BV
Full Resolution Auto-context
Refined BV Segmentation in BV bounding box (VNet III)
Refined Label: BV
Loss I: Low-Resolution
Original Labels
Downsample
Downsampled Labels
Loss II: Body
Loss III: BV
Zero Pad
Predicted Whole Image Labels
Forward Pass
Backward Pass

Fig. 2: Diagram of overall pipeline.



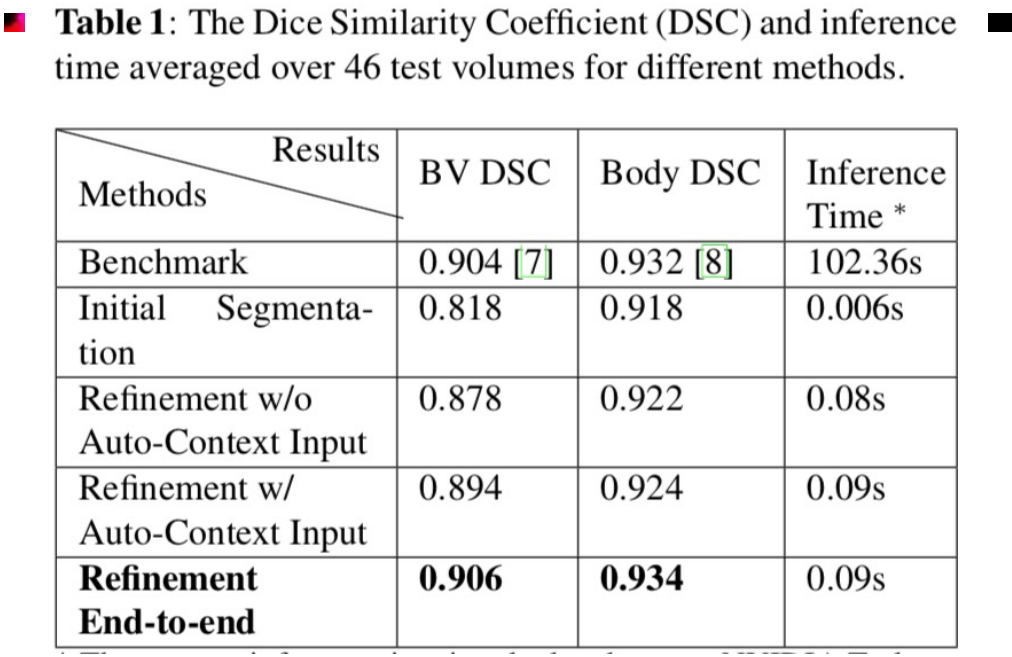


Table 1: The Dice Similarity Coefficient (DSC) and inference time averaged over 46 test volumes for different methods.

Results Methods	BV DSC	Body DSC	Inference Time *
Benchmark	0.904 [7]	0.932 [8]	102.36s
Initial Segmentation	0.818	0.918	0.006s
Refinement w/o Auto-Context Input	0.878	0.922	0.08s
Refinement w/ Auto-Context Input	0.894	0.924	0.09s
Refinement End-to-end	**0.906**	**0.934**	0.09s

Tongda Xu*, Ziming Qiu*, William Das, Chuiyu Wang, Jack Langerman, Nitin Nair, Orlando Aristizabal, Jonathan Mamou, Daniel H. Turnbull, Jeffrey A. Ketterling, Yao Wang, "Deep Mouse: An End-to-end Auto-context Refinement Framework for Brain Ventricle & Body Segmentation in Embryonic Mice Ultrasound Volumes," in 2020 IEEE 17th ISBI 2020, * equal contribution.

Pop Quizzes

- What loss functions to use for segmentation
- How does U-Net differ from the generic auto-encoder
- Why are the benefits of U-Net structure?

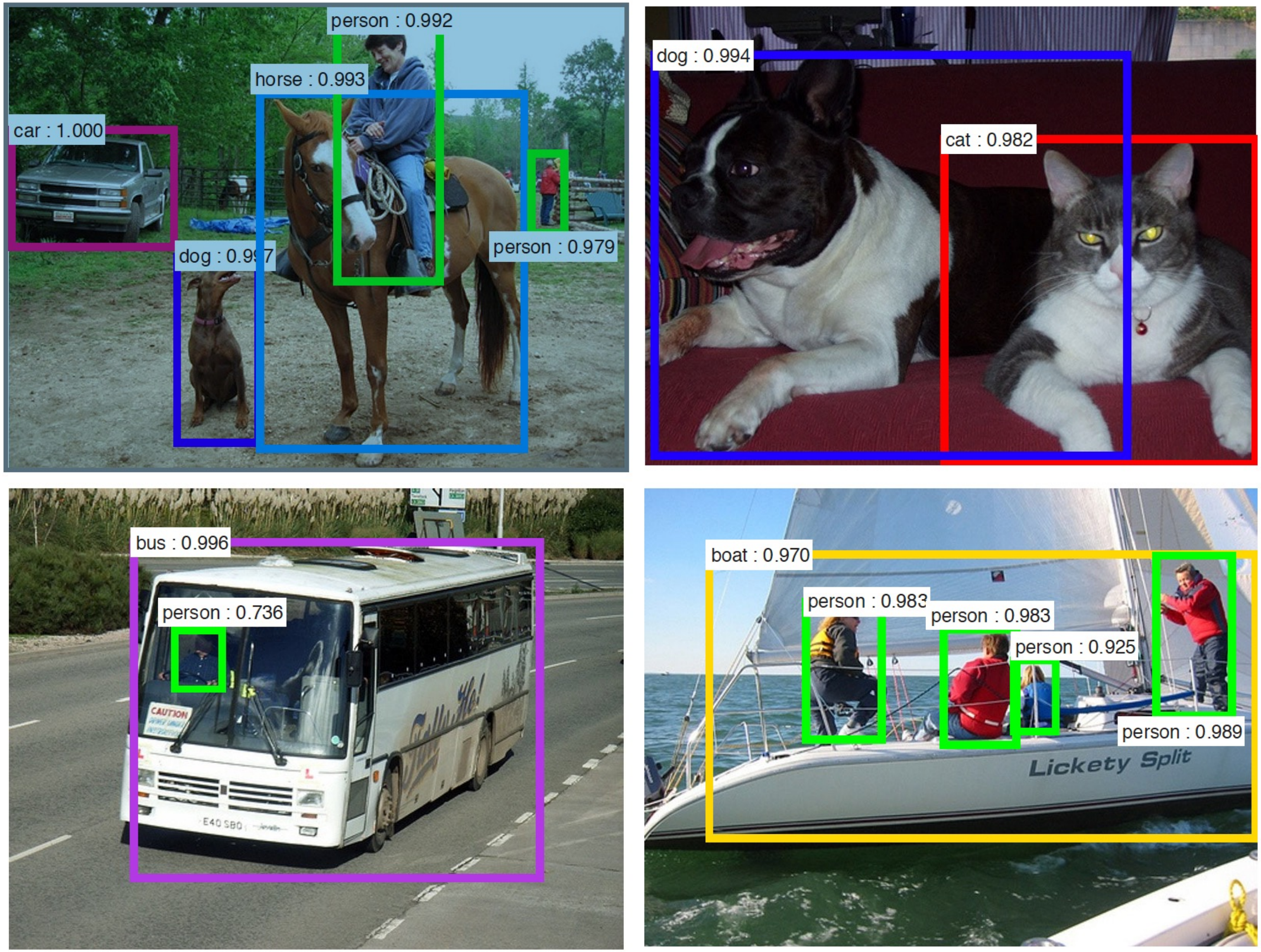
Pop Quizzes

- What loss functions to use for segmentation
  - Treat as classification problem at each pixel, sum the Cross entropy at all pixels
  - Soft DICE: similar to intersection over union
- How does U-NET differ from the generic auto-encoder
  - Has skip connection
  - Skip connection ensures high resolution information are preserved
- What are the Benefits of U-Net structure?
  - Multi-resolution processing to ensure both global and local information are considered

Outline (Part II)

- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

Object Localization and Classification



From: Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91-99. 2015.

Faster R-CNN

- A backbone network generate features
- A region proposal network (RPN) goes through each location in the feature maps, examines multiple anchor boxes and classify it as Object or not and furthermore predict deviation of the actual object box location and size from the anchor
- Overlapping proposals go through non-maximum suppression to select boxes with high “object-like” score
- Features in remaining boxes further go through a classification and regression layer (Post-RPN), to classify the object and further refine the box location and size

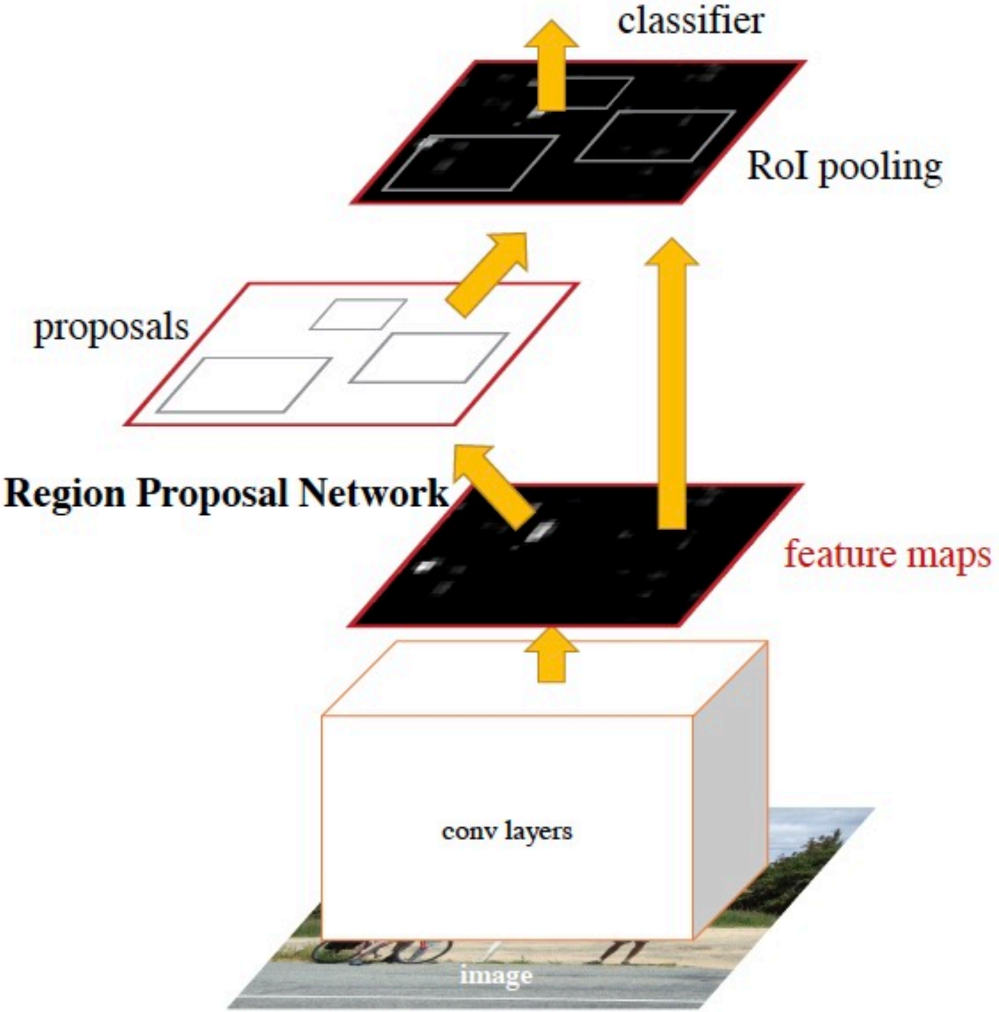


Figure from Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91-99. 2015.

Region Proposal Network

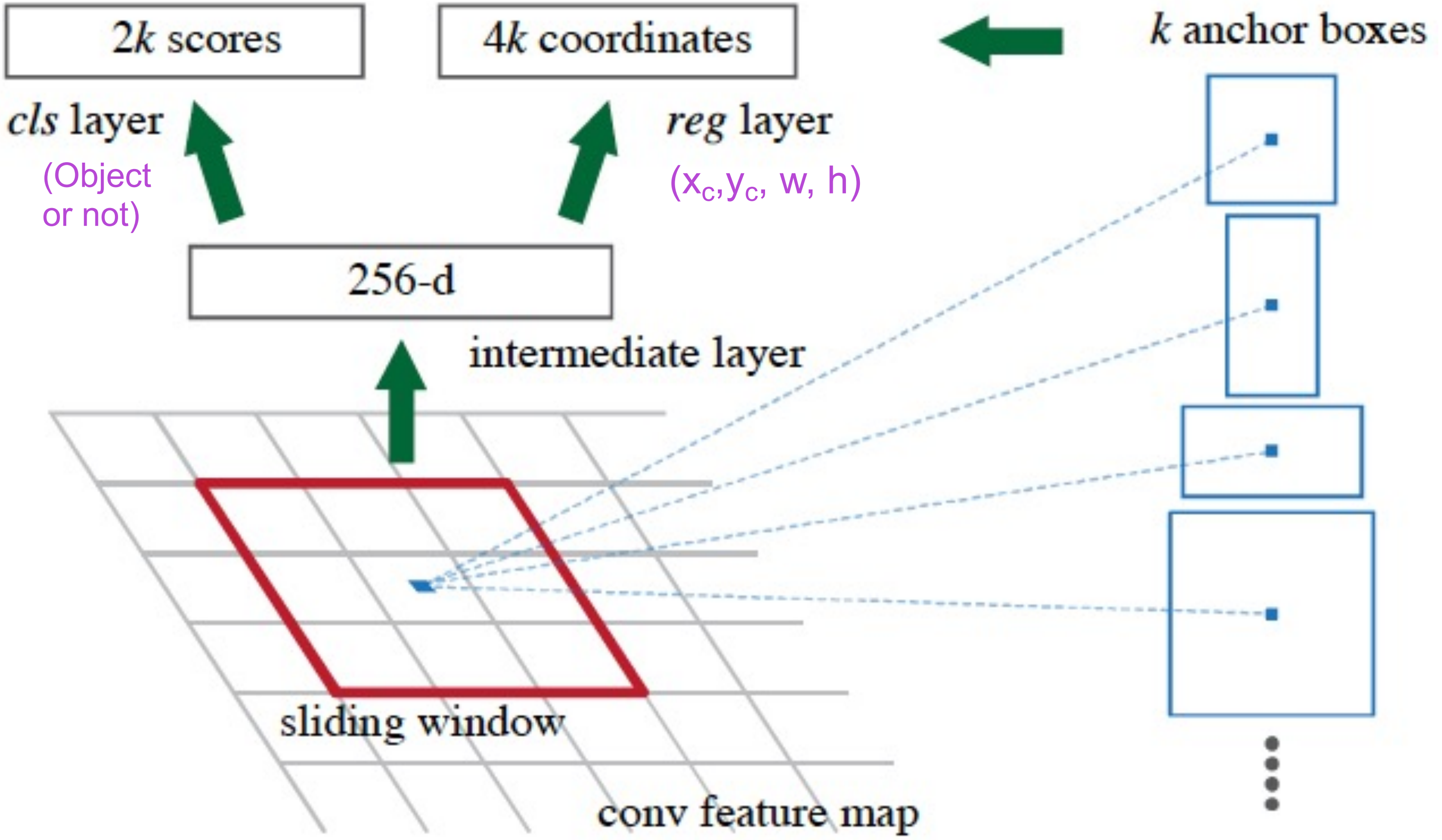
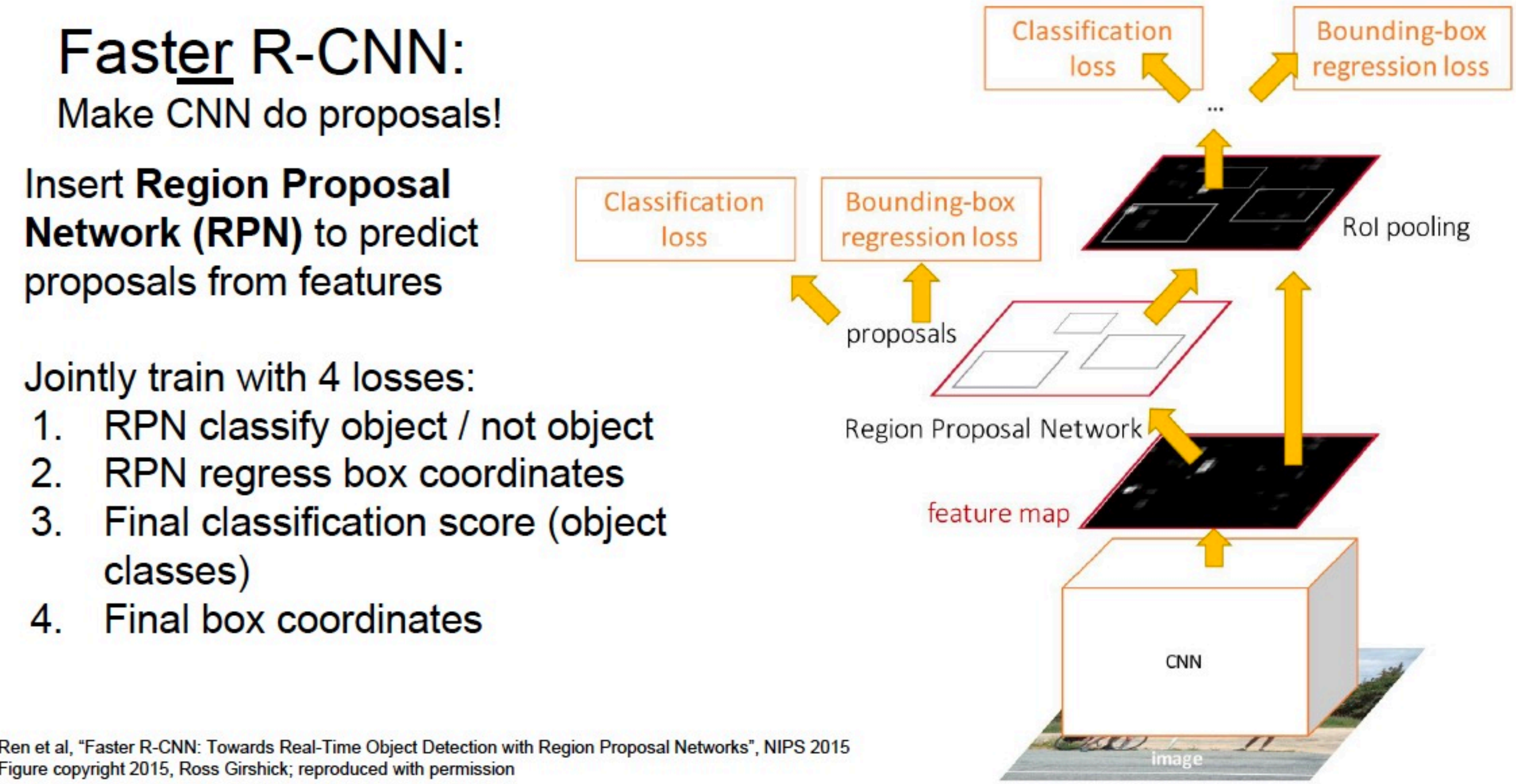


Figure from Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91-99. 2015.

Training of Faster R-CNN



Faster R-CNN:

Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features

Jointly train with 4 losses:

1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates

Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

From http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture11.pdf

Sample Results

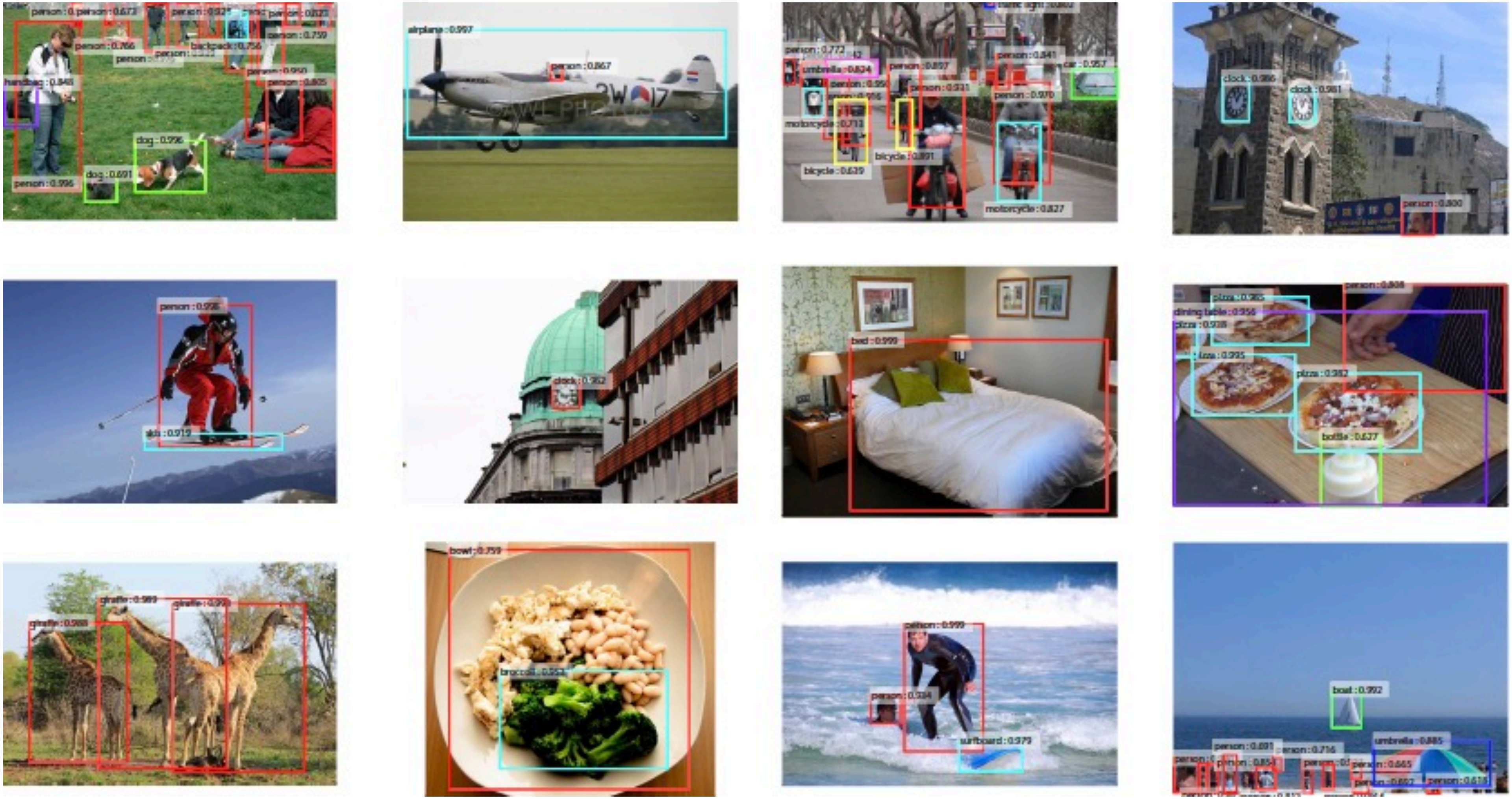


Figure from Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91-99. 2015.

Do we really need the second stage?

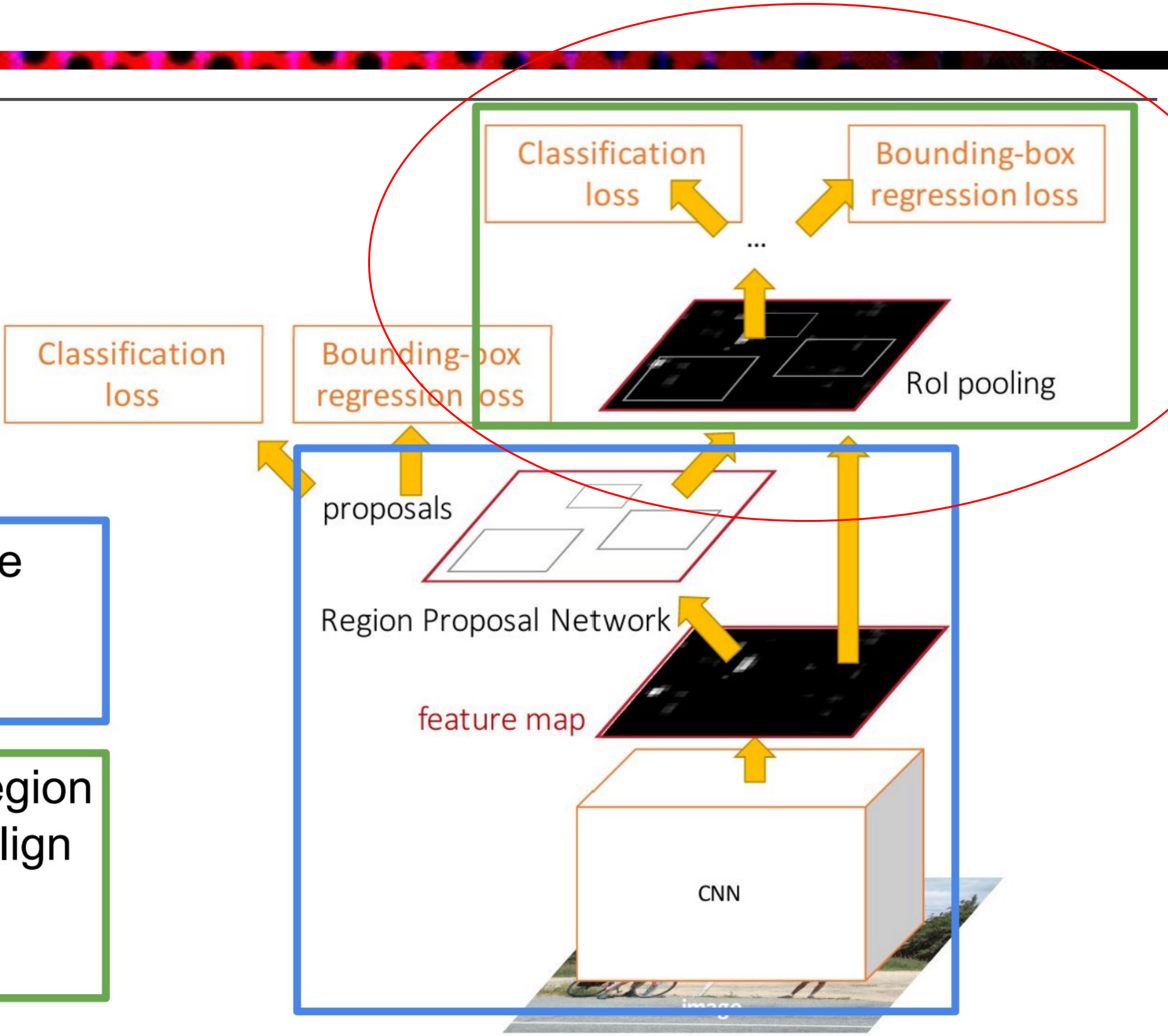
Faster R-CNN:

Make CNN do proposals!

Faster R-CNN is a
Two-stage object detector

First stage: Run once per image
- Backbone network
- Region proposal network

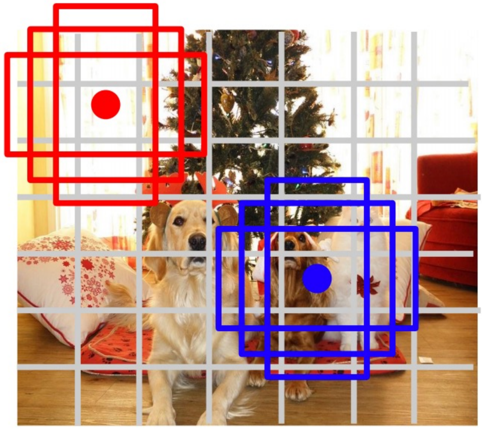
Second stage: Run once per region
- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset



From: http://cs231n.stanford.edu/slides/2020/lecture_12.pdf

Single Stage Object Detectors: YOLO / SSD / RetinaNet

Input image
3 x H x W



Divide image into grid
7 x 7

Image a set of **base boxes** centered at each grid cell
Here B = 3

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers:
(dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)
- Looks a lot like RPN, but category-specific!

Output:
7 x 7 x (5 * B + C)

Redmon et al, "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016
Liu et al, "SSD: Single-Shot MultiBox Detector", ECCV 2016
Lin et al, "Focal Loss for Dense Object Detection", ICCV 2017

From: http://cs231n.stanford.edu/slides/2020/lecture_12.pdf

- Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779-788. 2016.
- Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "SSD: Single shot multibox detector." In *European conference on computer vision*, pp. 21-37. Springer, Cham, 2016. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779-788. 2016.

Object Detection: Lost of Variables …

Backbone Network
VGG16
ResNet-101
Inception V2
Inception V3
Inception
ResNet
MobileNet

"Meta-Architecture"
Two-stage: Faster R-CNN
Single-stage: YOLO / SSD
Hybrid: R-FCN

Image Size
# Region Proposals
…

Takeaways
Faster R-CNN is slower but more accurate

SSD is much faster but not as accurate

Bigger / Deeper backbones work better

Huang et al, "Speed/accuracy trade-offs for modern convolutional object detectors", CVPR 2017
Zou et al, "Object Detection in 20 Years: A Survey", arXiv 2019

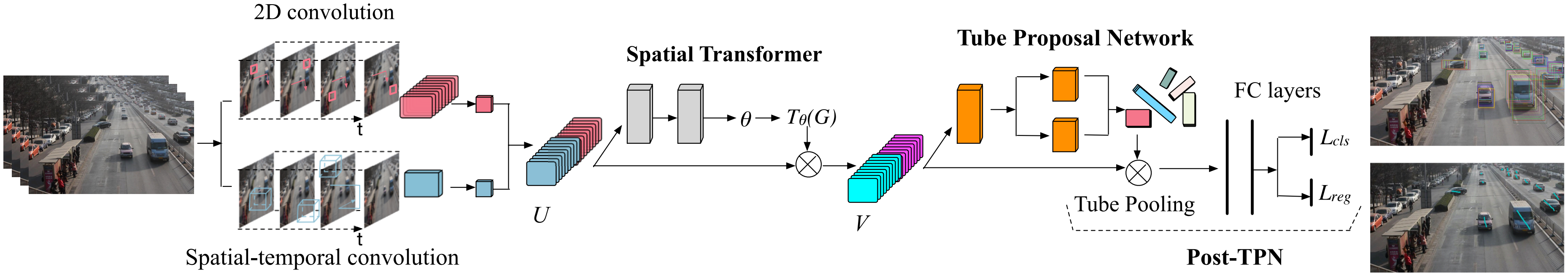
R-FCN: Dai et al, "R-FCN: Object Detection via Region-based Fully Convolutional Networks", NIPS 2016
Inception-V2: Ioffe and Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", ICML 2015
Inception V3: Szegedy et al, "Rethinking the Inception Architecture for Computer Vision", arXiv 2016
Inception ResNet: Szegedy et al, "Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning", arXiv 2016
MobileNet: Howard et al, "Efficient Convolutional Neural Networks for Mobile Vision Applications", arXiv 2017

From: http://cs231n.stanford.edu/slides/2020/lecture_12.pdf

Work at NYU Video Lab: Robust Vehicle Tracking at Urban Intersections

- Vehicle detection and tracking at very congested urban intersections
  - Traffic accidence analysis based on vehicle trajectory data
  - Joint project with NYU Center for Urban Science + Progress (CUSP)
- Challenges
  - Low resolution NYC Dept of Transportation surveillance videos
  - Severe occlusion in dense traffic
  - Vanishing point (non-bird eye view) viewing angles
  - Shadows and illumination changes
- Developing a deep learning network that can simultaneously detect and track a video object
  - Detect bounding tubes that cover moving objects in short video segments
  - Extension of faster region-CNN, which detects bounding boxes in individual frames
- Video Lab Student: Chenge Li (Ph.D. 2019)

TrackNet Model Overview



Feature Extraction

We found using C3D alone is not as good as using both C3D and VGG

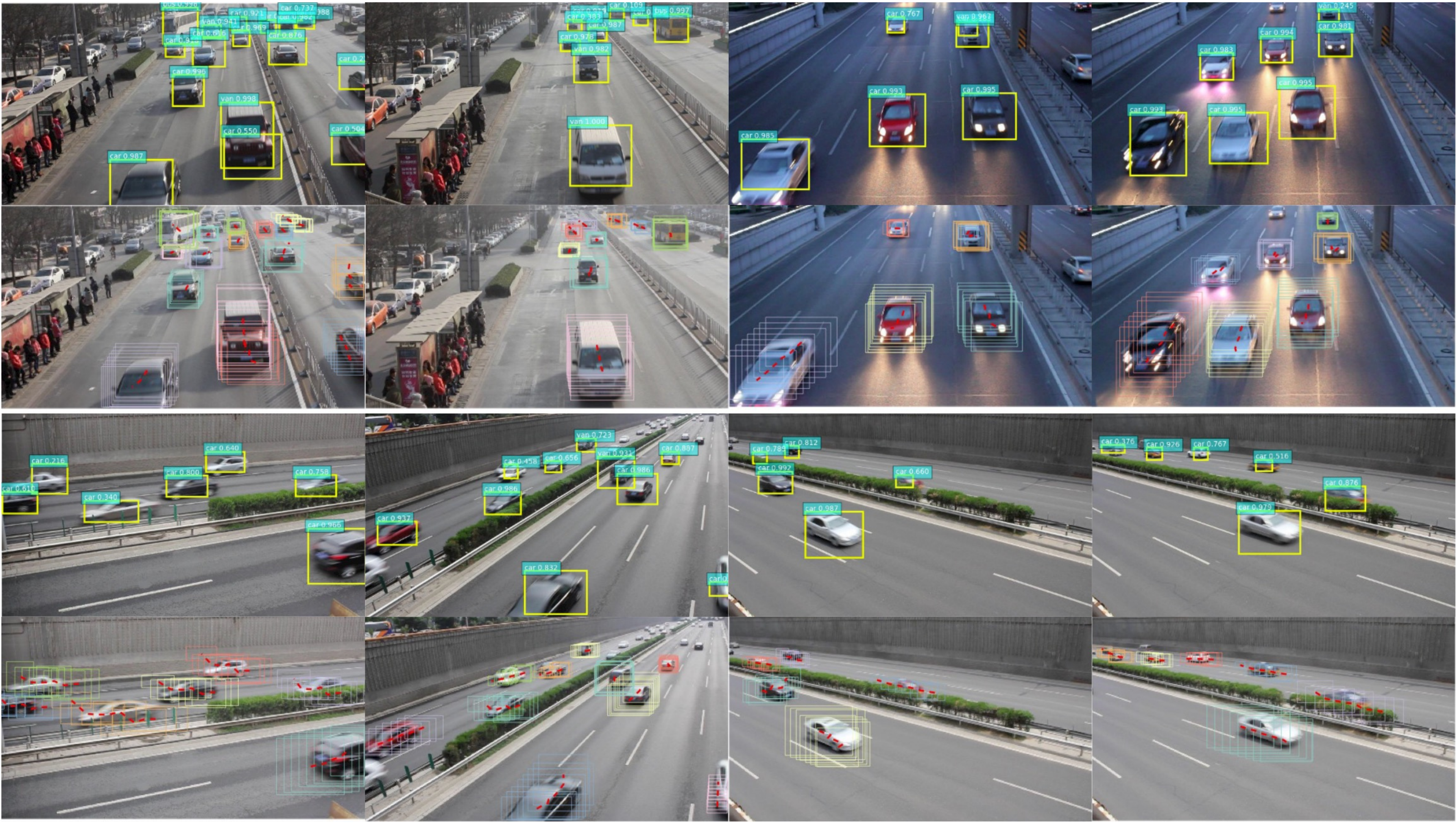
Transform feature maps so that traffic videos from different view angles have similar feature representation

At each candidate location, test multiple anchor tubes: generate a “objectness score” and the offset from the original anchor tube location and shape

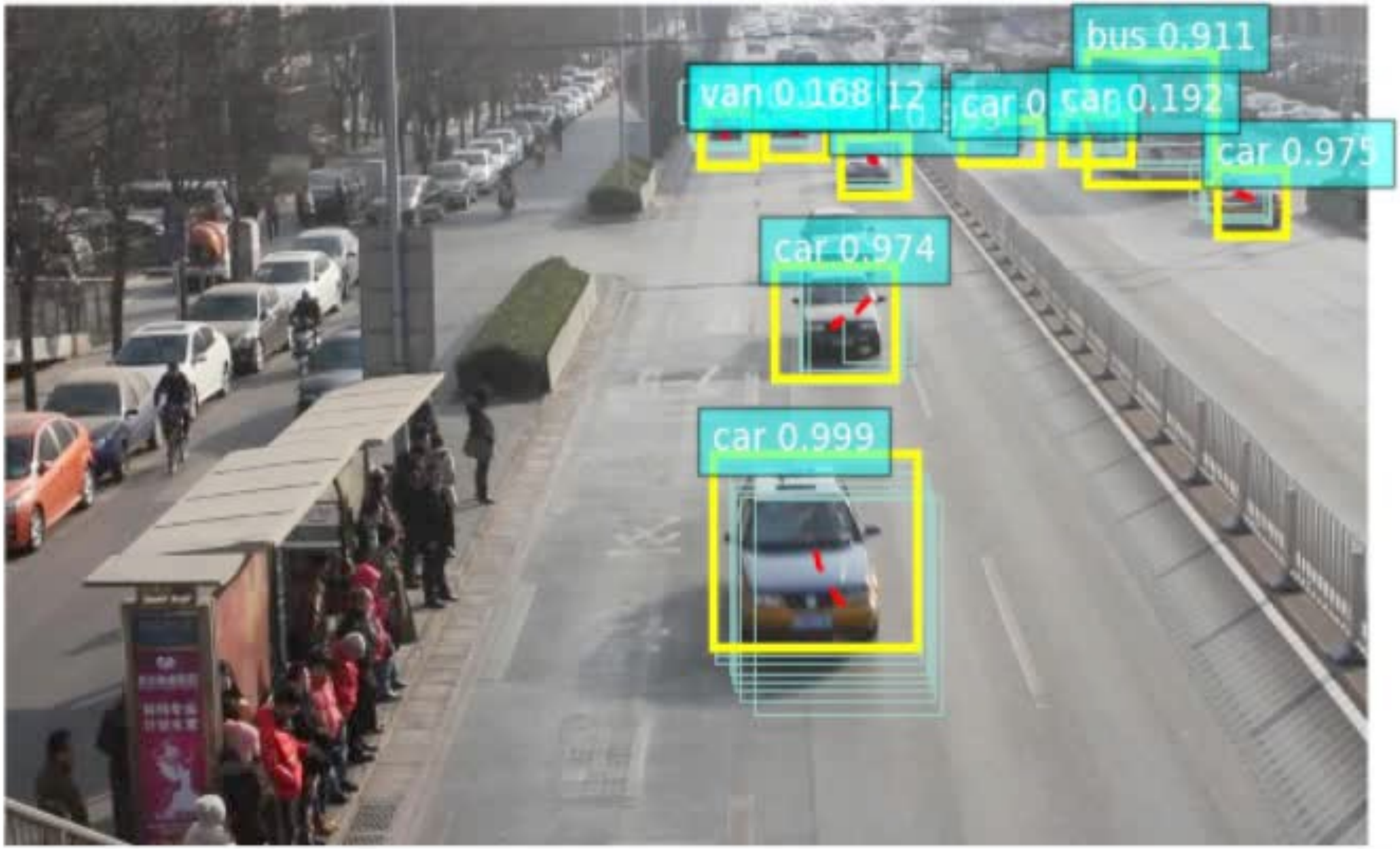
For each proposal tube with high “objectness score”, further refine the tube parameters, and classify it into one of predefined object category

Chenge Li, Gregory Dobler, Xin Feng, Yao Wang "**TrackNet: TrackNet: Simultaneous Detection and Tracking of Multiple Objects**", https://arxiv.org/abs/1902.01466

Sample Results: Snap Shot



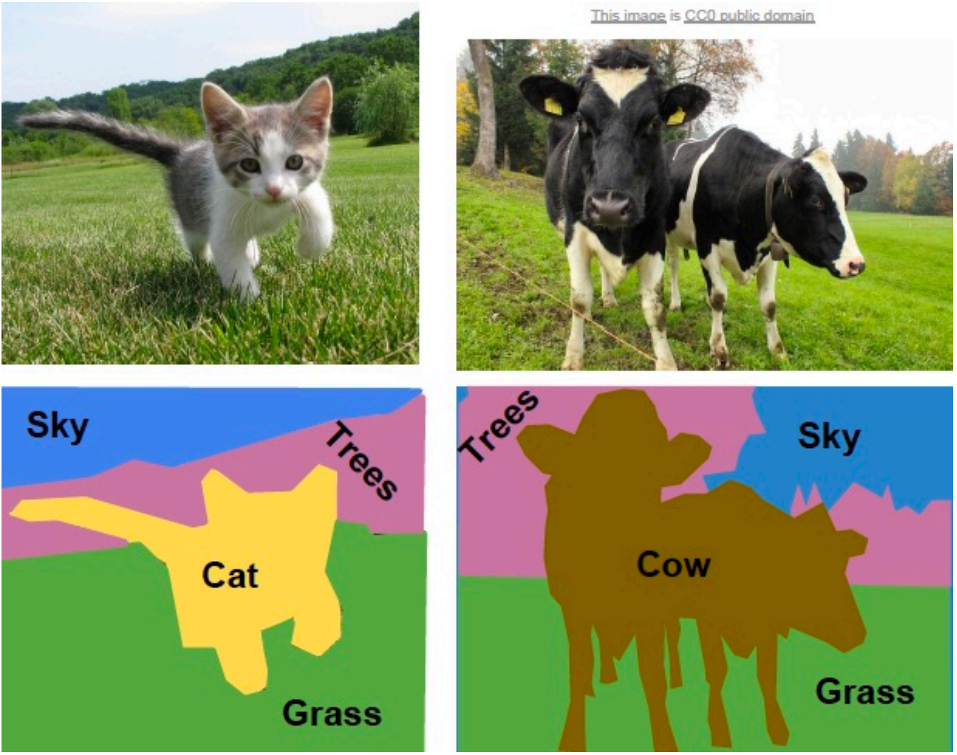
Sample Results: Video

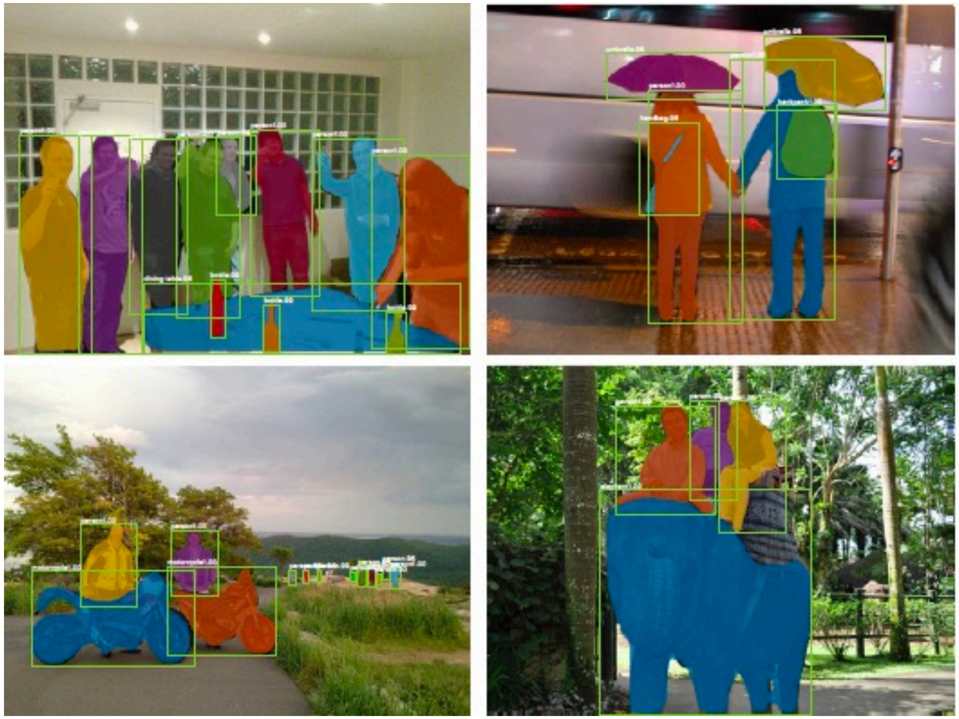


Outline (Part II)

- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- Interpretation of trained models

Semantic vs. Instance Segmentation


This image is CC0 public domain

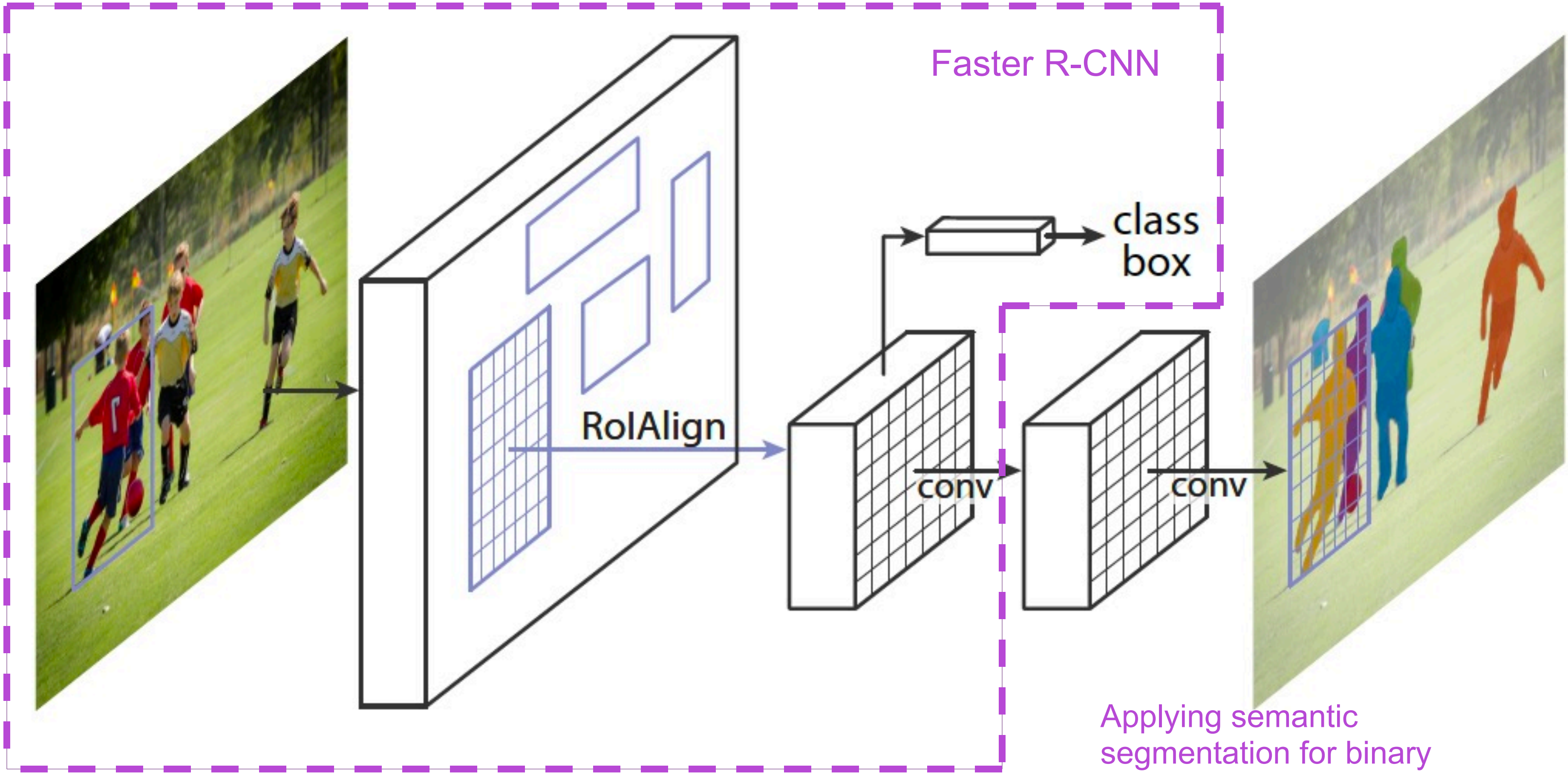


Semantic segmentation: Each pixel is labeled into one object class. The two cows have the same color!

Instance segmentation: Each instance of the same type of object is given a different color!

Image From: He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In *Proceedings of the IEEE international conference on computer vision*, pp. 2961-2969. 2017. https://arxiv.org/pdf/1703.06870.pdf

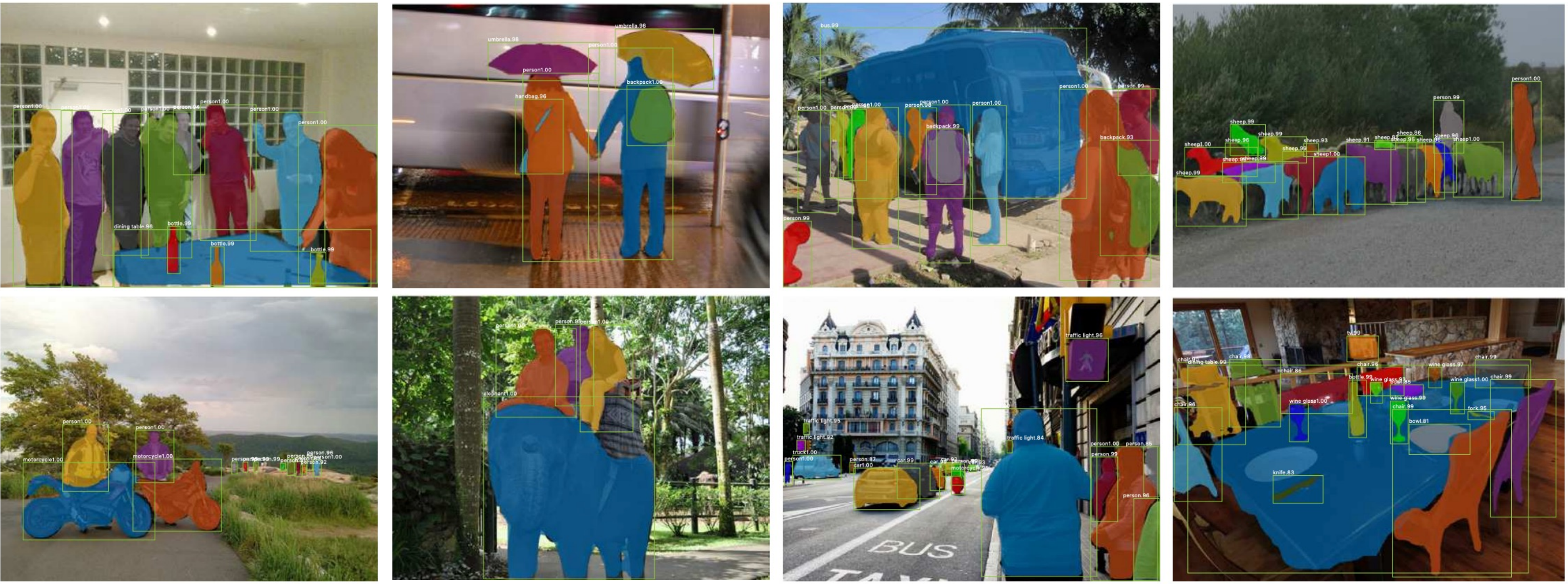
Mask R-CNN: Running a segmentation network for each detected object



Applying semantic segmentation for binary classes in each detected ROI

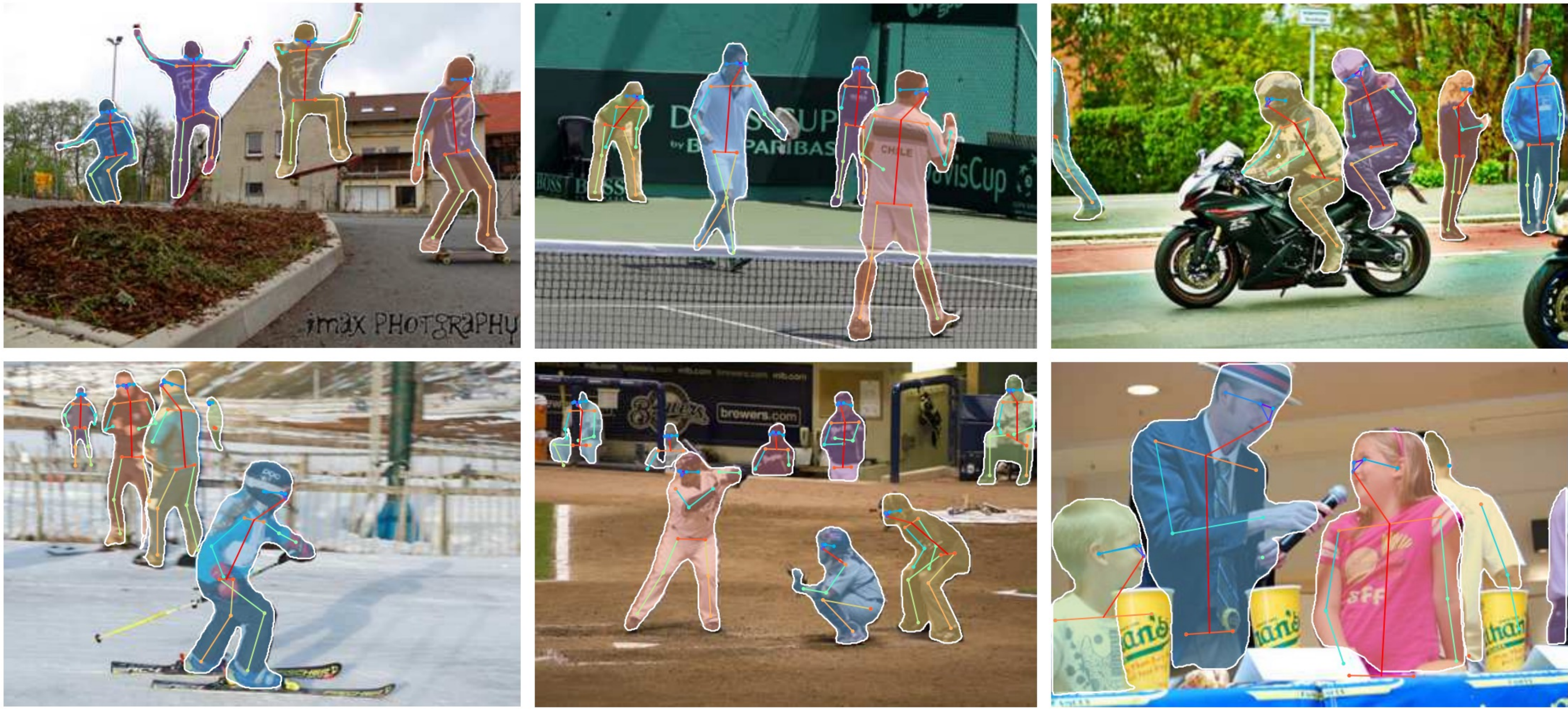
From: He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In *Proceedings of the IEEE international conference on computer vision*, pp. 2961-2969. 2017. https://arxiv.org/pdf/1703.06870.pdf

Mask-RCNN: Sample Results



From: He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In *Proceedings of the IEEE international conference on computer vision*, pp. 2961-2969. 2017. https://arxiv.org/pdf/1703.06870.pdf

Mask RCNN: Also estimate the human pose



From: He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In *Proceedings of the IEEE international conference on computer vision*, pp. 2961-2969. 2017. https://arxiv.org/pdf/1703.06870.pdf

Pop Quizzes

- How does faster R-CNN work?
- How does YOLO work?
- How does instance segmentation work?

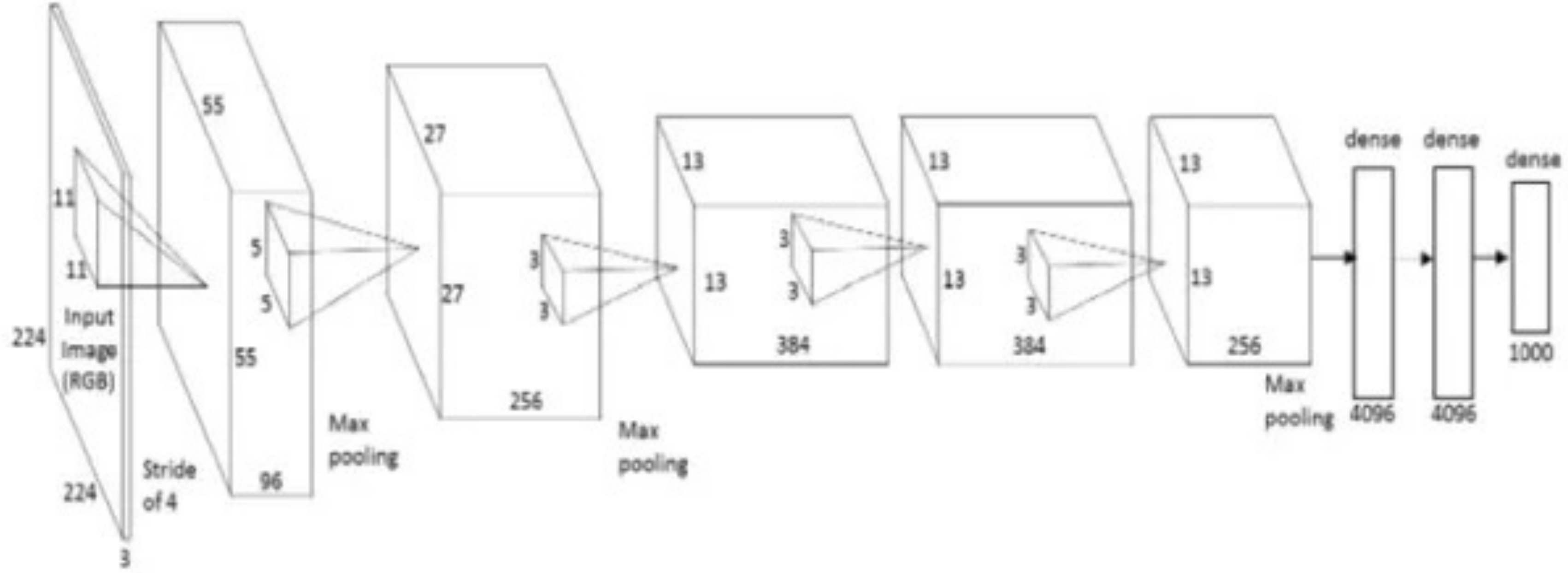
Pop Quizzes

- How does faster R-CNN work?
  - Backbone: Generate features
  - Region proposal network: for each possible location, evaluate multiple possible proposals by regressing the box shape and classify between object/non-object
  - Region refinement and classification: for each proposal with high object scores, further refine location and classify among many object classes
- How does YOLO /SSD work?
  - One pass: for each possible location and box shape, directly classify and regress the box shape
  - Less accurate
- How does instance segmentation work (Mask-RCNN)?
  - Further apply segmentation on each detected object region

Outline (Part II)

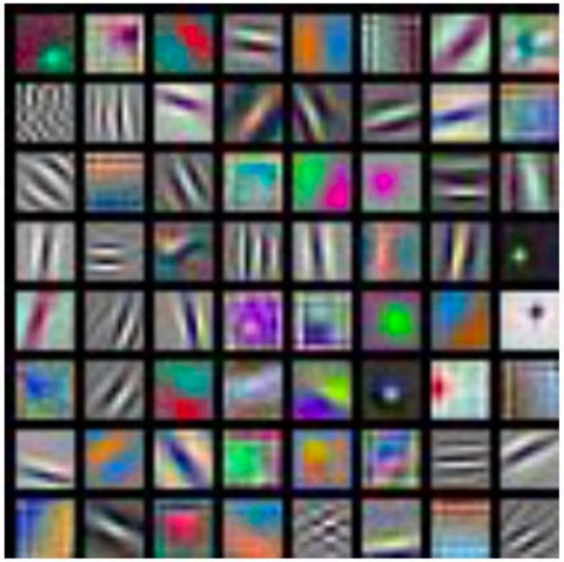
- Neural Nets and Conv Nets and Model Training (Review)
- Some important extensions of conv. layers
- Popular classification models and transfer learning
- Image to image autoencoder
  - Denoising
- Semantic Segmentation using Multiresolution Autoencoder
- Object detection and classification
- Instance segmentation
- ➡ Interpretation of trained models

What is going on inside ConvNet?



- What filters are used in each layer?
- What features are extracted after each layer?
- What regions/features the network uses to make classification decisions?

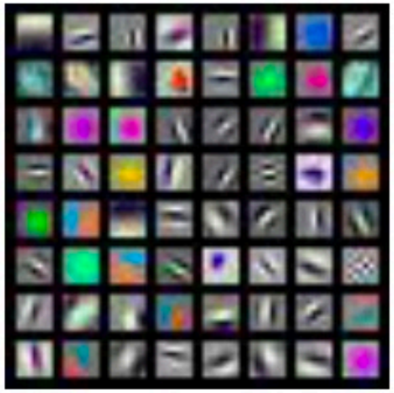
First layer: Visualize Filters



AlexNet:
64 x 3 x 11 x 11

ResNet-18:
64 x 3 x 7 x 7

ResNet-101:
64 x 3 x 7 x 7



DenseNet-121:
64 x 3 x 7 x 7

From: http://cs231n.stanford.edu/slides/2020/lecture_13.pdf

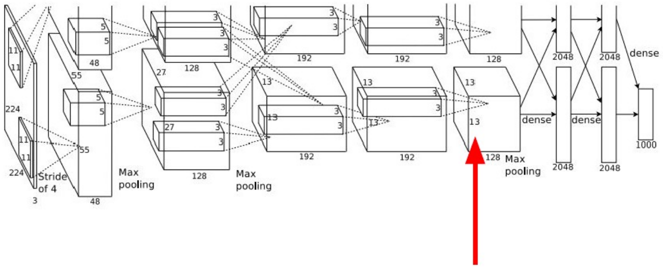
Since first layer input is 3 color channels, we can visualize the filters in color.
The filters in different networks are similar: extract edges in different directions and other basic patterns, different color transitions.

Filters at other layers?

- Much harder to interpret
- Multi-channel input
- Instead look at what features / patterns generate high response in each output channel at each layer

Features Detected at Intermediate Layers

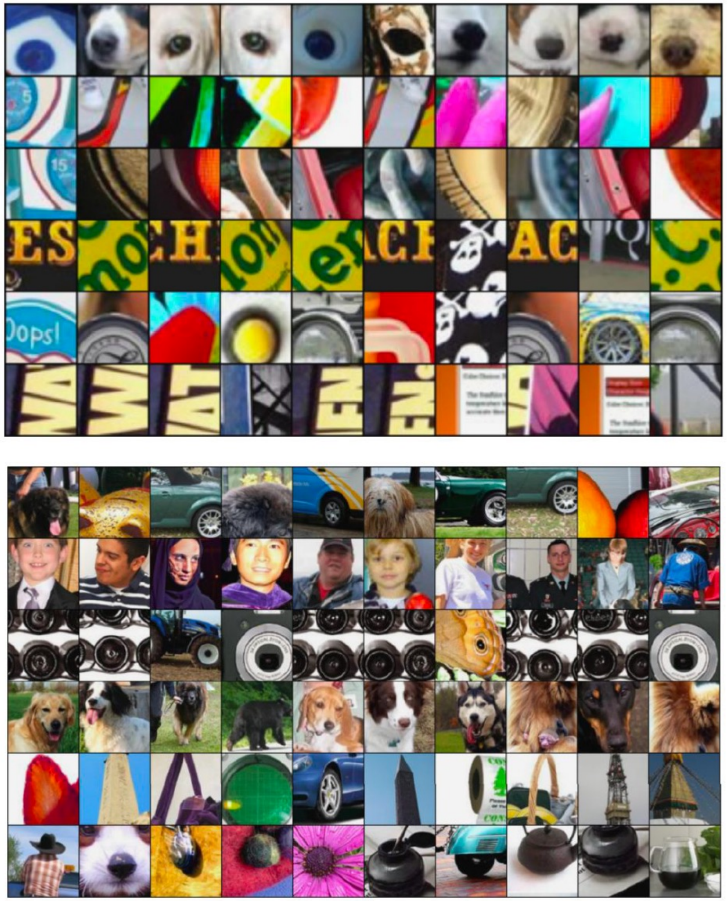
Maximally Activating Patches



Pick a layer and a channel; e.g. conv5 is 128 x 13 x 13, pick channel 17/128

Run many images through the network, record values of chosen channel

Visualize image patches that correspond to maximal activations



Springenberg et al, “Striving for Simplicity: The All Convolutional Net”, ICLR Workshop 2015
Figure copyright Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller, 201
reproduced with permission.

From: http://cs231n.stanford.edu/slides/2020/lecture_13.pdf

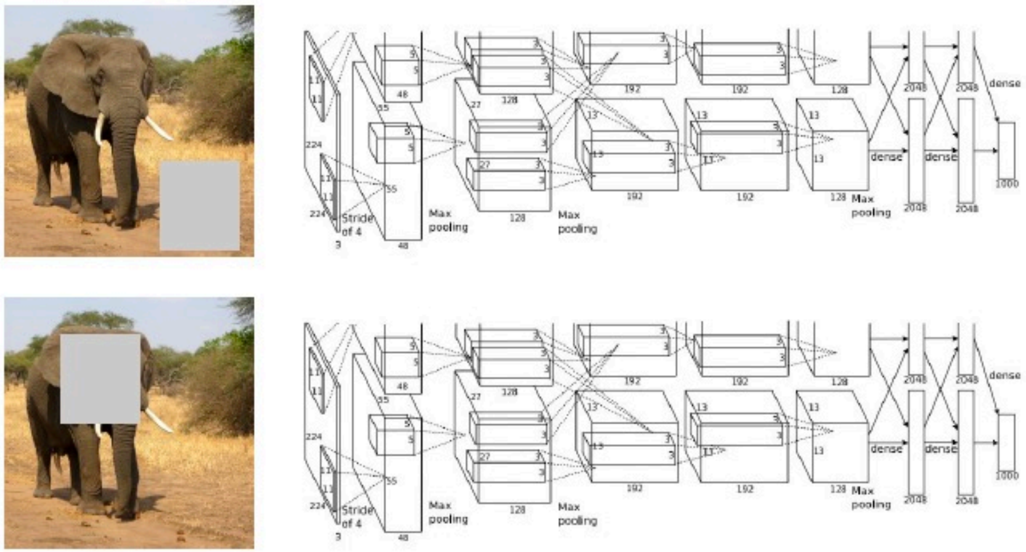
Different row corresponds to different feature channels

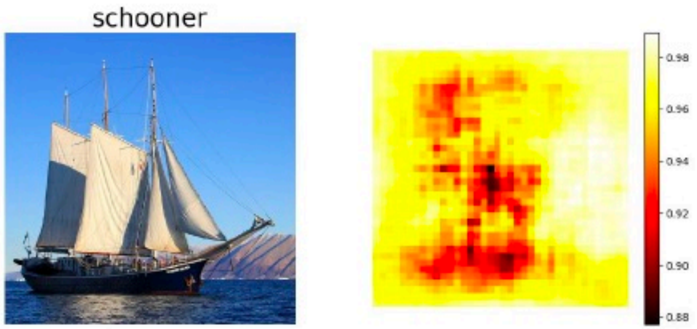
Which pixels contribute to the classification decision?

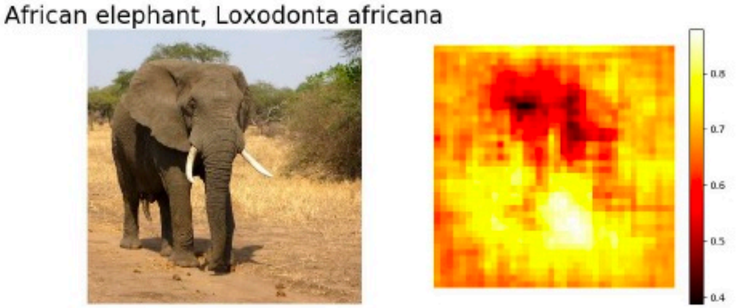
- Described by a saliency map
- How to derive the saliency map?

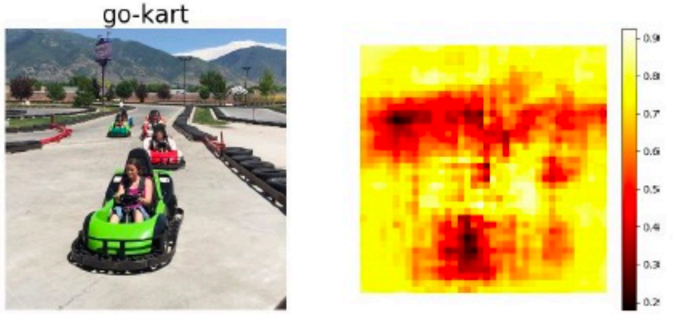
Which pixels matter: Saliency via Occlusion

Mask part of the image before feeding to CNN, check how much predicted probabilities change









Zeiler and Fergus, "Visualizing and Understanding Convolutional Networks", ECCV 2014

Boat image is CC0 public domain
Elephant image is CC0 public domain
Go-Karts image is CC0 public domain

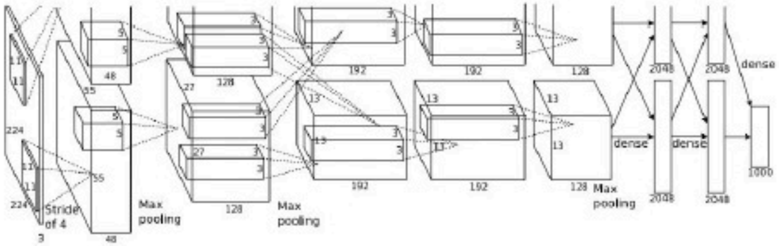
From: http://cs231n.stanford.edu/slides/2020/lecture_13.pdf

Negative change indicates this pixel is very important for recognizing the class

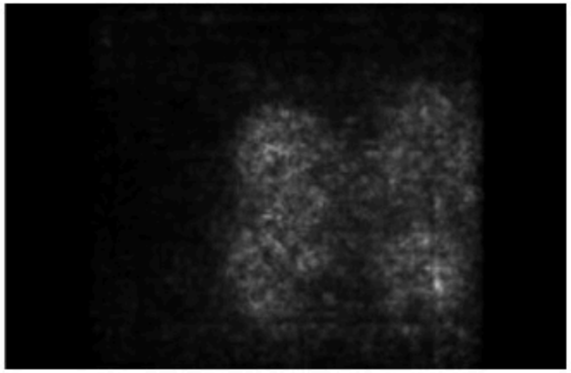
Very slow: has to run the model with one pixel occluded at a time, for all pixels

Which pixels matter: Saliency via Backprop

Forward pass: Compute probabilities



Dog

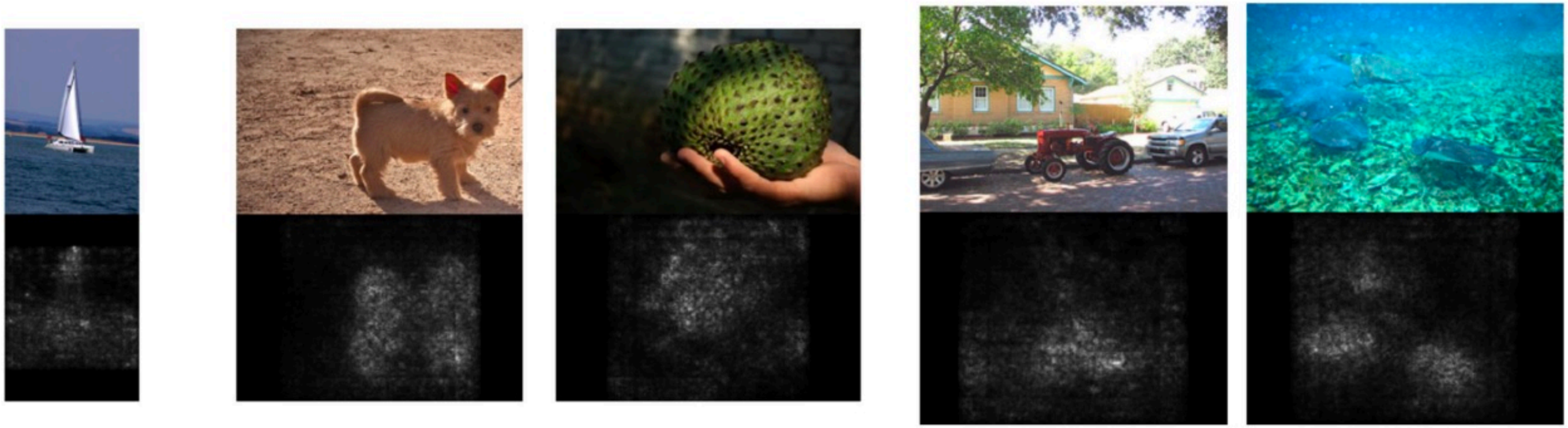


Compute gradient of (unnormalized) class score with respect to image pixels, take absolute value and max over RGB channels

Simonyan, Vedaldi, and Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR Workshop 2014.
Figures copyright Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, 2014; reproduced with permission.

From: http://cs231n.stanford.edu/slides/2020/lecture_13.pdf

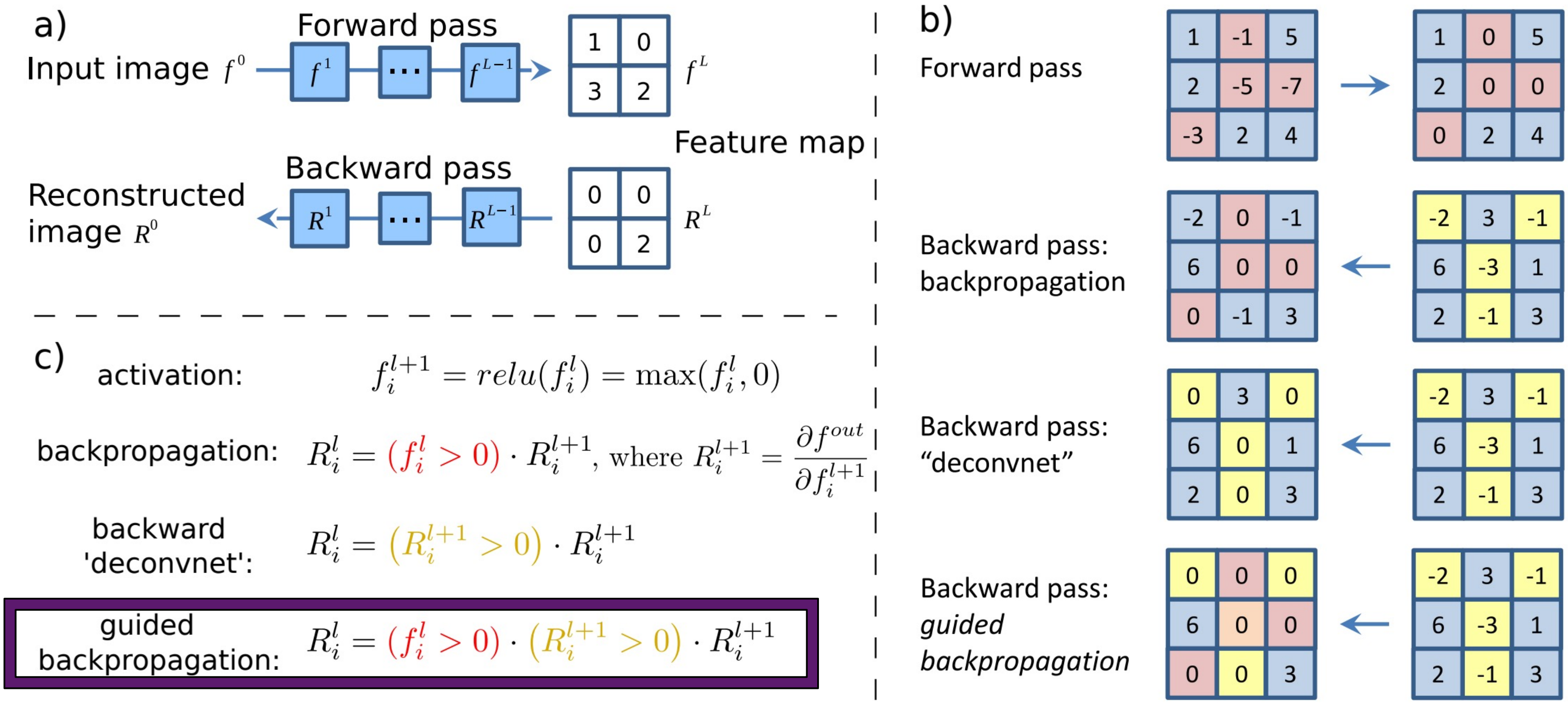
Sample Saliency Maps by Computing Gradients with Respect to Input



Simonyan, Vedaldi, and Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”, ICLR Workshop 2014.

Generally, the results are noisy.

Saliency via Guided Backprop



$$f^L = \begin{matrix} 1 & 0 \\ 3 & 2 \end{matrix} \qquad R^L = \begin{matrix} 0 & 0 \\ 0 & 2 \end{matrix}$$

Forward pass

$$\begin{matrix} 1 & -1 & 5 \\ 2 & -5 & -7 \\ -3 & 2 & 4 \end{matrix} \rightarrow \begin{matrix} 1 & 0 & 5 \\ 2 & 0 & 0 \\ 0 & 2 & 4 \end{matrix}$$

Backward pass: backpropagation

$$\begin{matrix} -2 & 0 & -1 \\ 6 & 0 & 0 \\ 0 & -1 & 3 \end{matrix} \leftarrow \begin{matrix} -2 & 3 & -1 \\ 6 & -3 & 1 \\ 2 & -1 & 3 \end{matrix}$$

Backward pass: "deconvnet"

$$\begin{matrix} 0 & 3 & 0 \\ 6 & 0 & 1 \\ 2 & 0 & 3 \end{matrix} \leftarrow \begin{matrix} -2 & 3 & -1 \\ 6 & -3 & 1 \\ 2 & -1 & 3 \end{matrix}$$

Backward pass: *guided backpropagation*

$$\begin{matrix} 0 & 0 & 0 \\ 6 & 0 & 0 \\ 0 & 0 & 3 \end{matrix} \leftarrow \begin{matrix} -2 & 3 & -1 \\ 6 & -3 & 1 \\ 2 & -1 & 3 \end{matrix}$$

activation: $f_i^{l+1} = relu(f_i^l) = \max(f_i^l, 0)$

backpropagation: $R_i^l = (f_i^l > 0) \cdot R_i^{l+1}$, where $R_i^{l+1} = \dfrac{\partial f^{out}}{\partial f_i^{l+1}}$

backward 'deconvnet': $R_i^l = (R_i^{l+1} > 0) \cdot R_i^{l+1}$

guided backpropagation: $R_i^l = (f_i^l > 0) \cdot (R_i^{l+1} > 0) \cdot R_i^{l+1}$

Springenberg, Jost Tobias, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. "Striving for simplicity: The all convolutional net." *arXiv preprint arXiv:1412.6806* (2014).

Layer 6: Low level features

Layer 9: High level features

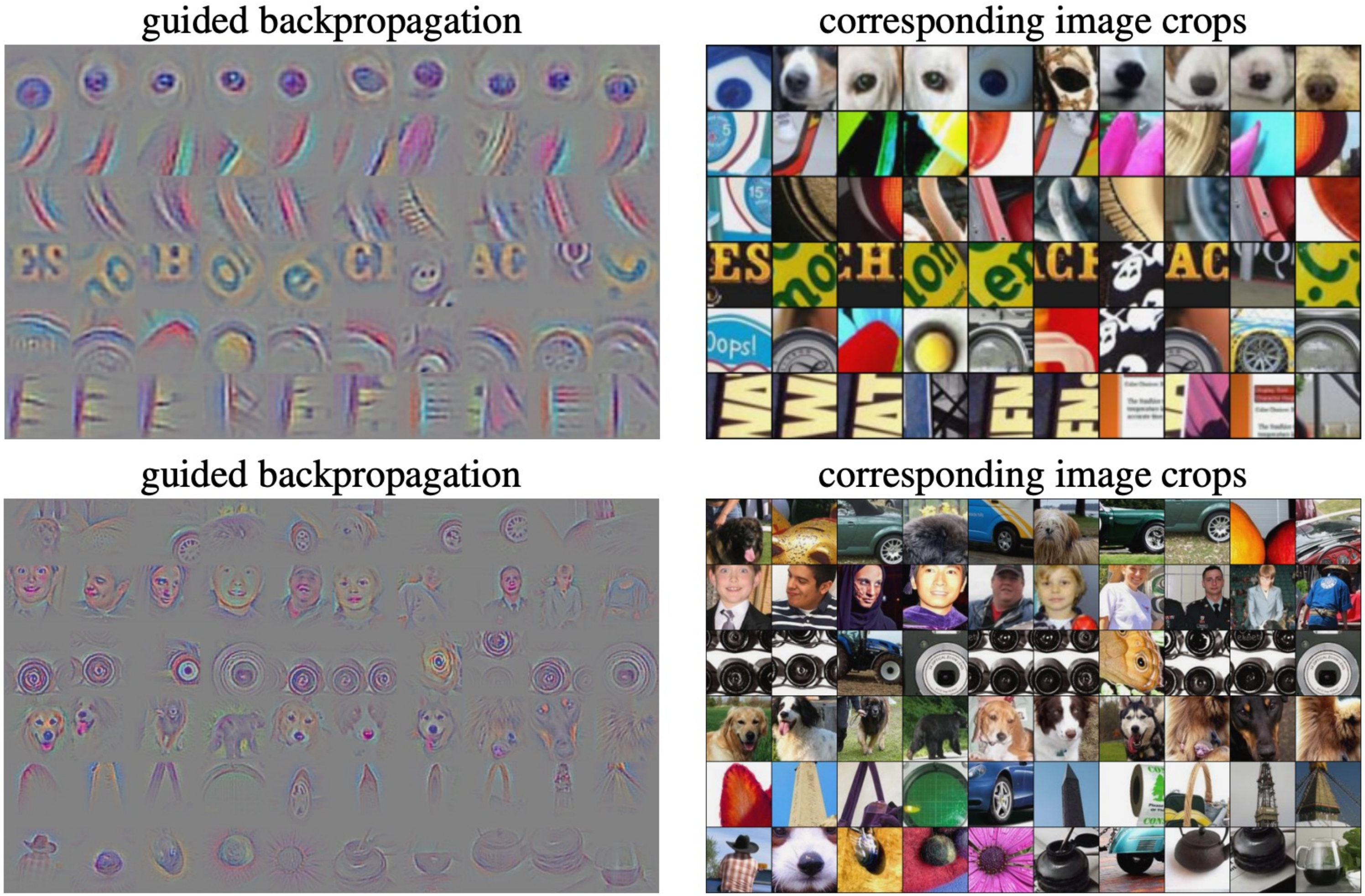


Figure 3: Visualization of patterns learned by the layer conv6 (top) and layer conv9 (bottom) of the network trained on ImageNet. Each row corresponds to one filter. The visualization using "guided backpropagation" is based on the top 10 image patches activating this filter taken from the ImageNet dataset. Note that image sizes are not preserved (in order to save space).

Springenberg, Jost Tobias, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. "Striving for simplicity: The all convolutional net." *arXiv preprint arXiv:1412.6806* (2014).

Saliency vis Class Activation Map (CAM)

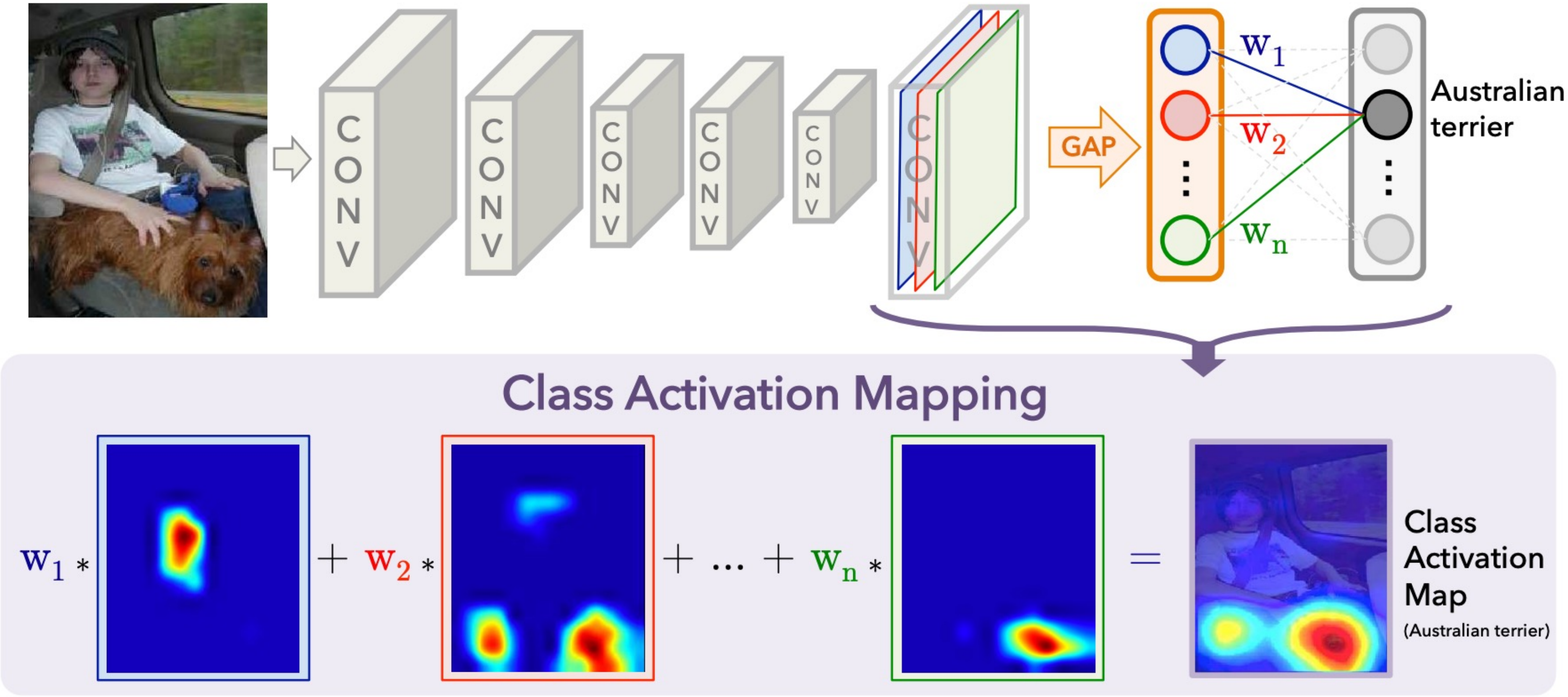


Figure 2. Class Activation Mapping: the predicted class score is mapped back to the previous convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions.

From: Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning deep features for discriminative localization." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921-2929. 2016. https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Zhou_Learning_Deep_Features_CVPR_2016_paper.pdf

Only applicable for the global average pooling of feature maps before one fully connected layer.

Sample Class Activation Maps (CAM)

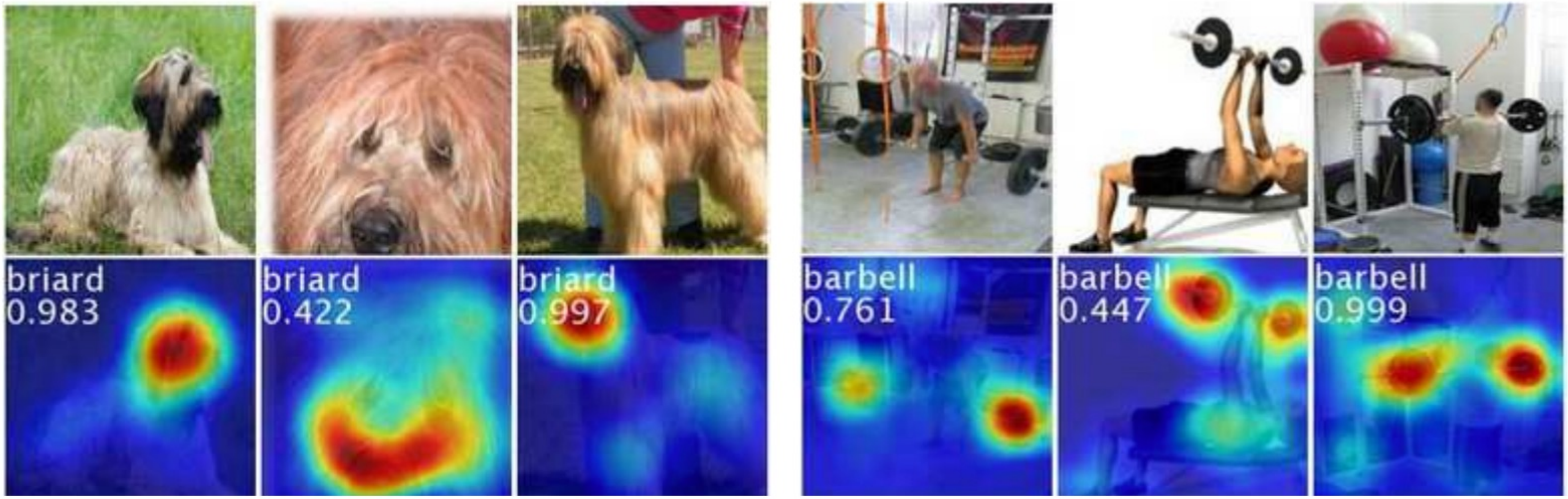


Figure 3. The CAMs of two classes from ILSVRC [21]. The maps highlight the discriminative image regions used for image classification, the head of the animal for *briard* and the plates in *barbell*.

From: Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning deep features for discriminative localization." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921-2929. 2016. https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Zhou_Learning_Deep_Features_CVPR_2016_paper.pdf

Grad-CAM, Guided Grad-CAM

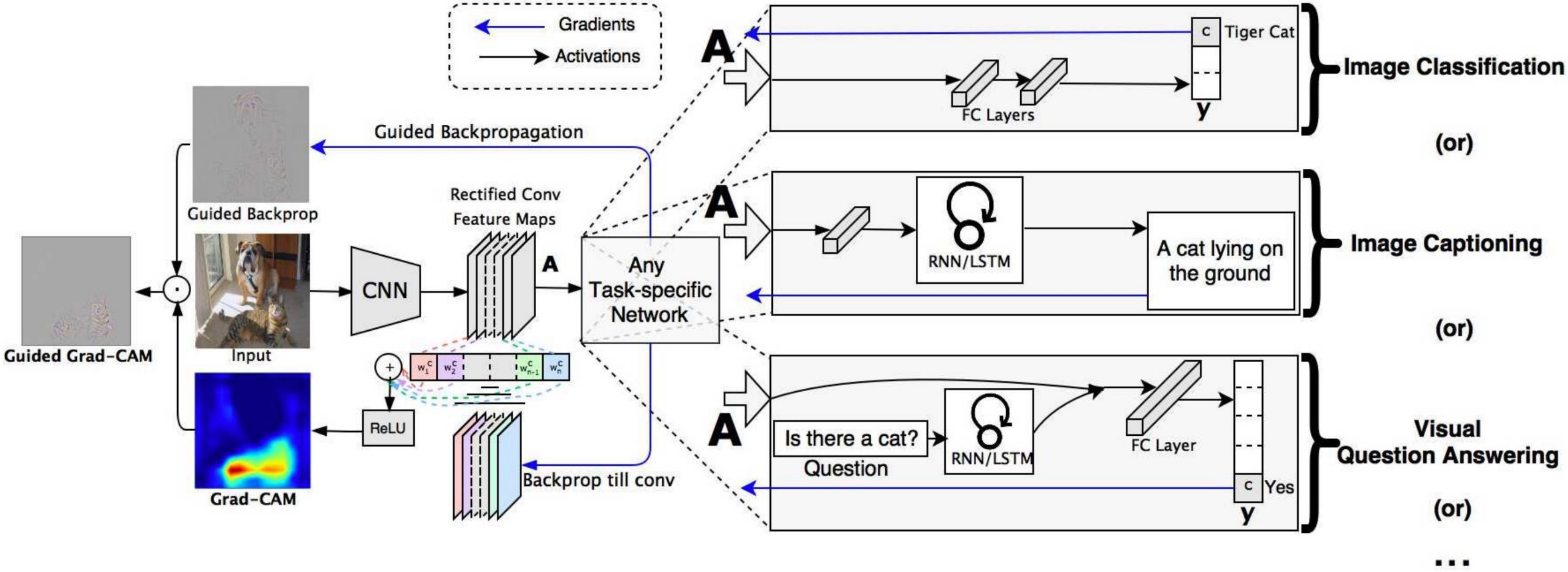
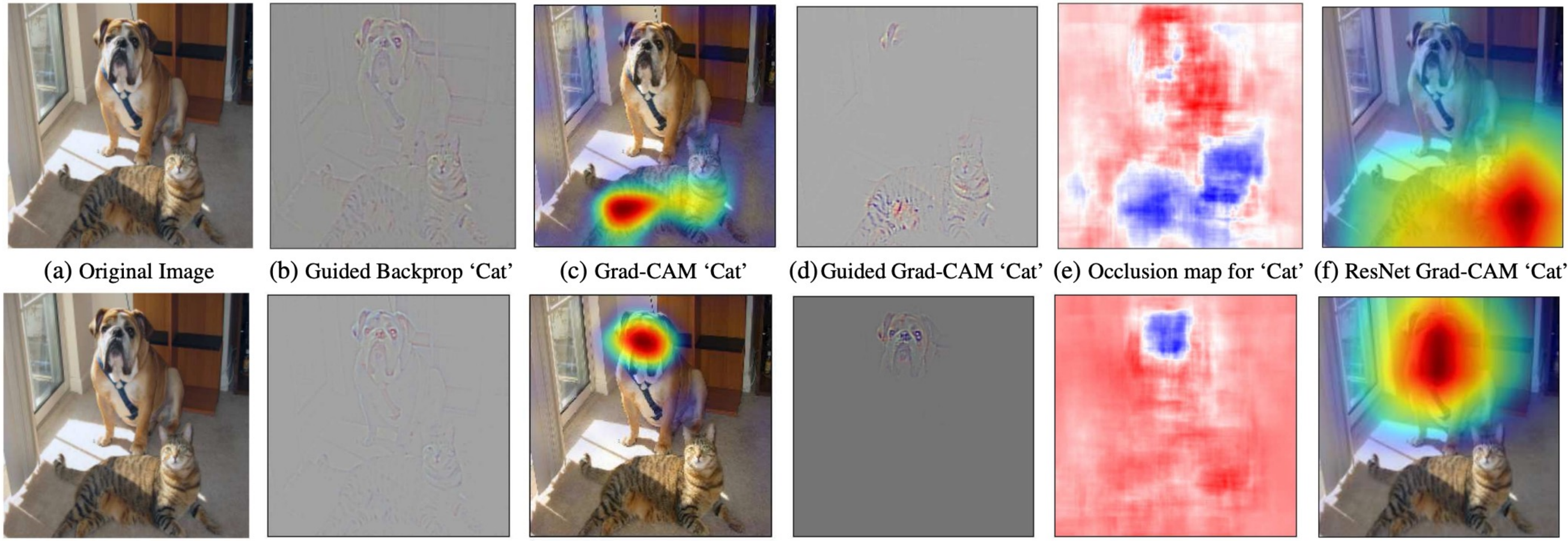


Figure 2: Grad-CAM overview: Given an image and a class of interest (*e.g.*, 'tiger cat' or any other type of differentiable output) as input, we forward propagate the image through the CNN part of the model and then through task-specific computations to obtain a raw score for the category. The gradients are set to zero for all classes except the desired class (tiger cat), which is set to 1. This signal is then backpropagated to the rectified convolutional feature maps of interest, which we combine to compute the coarse Grad-CAM localization (blue heatmap) which represents where the model has to look to make the particular decision. Finally, we pointwise multiply the heatmap with guided backpropagation to get Guided Grad-CAM visualizations which are both high-resolution and concept-specific.

Selvaraju, Ramprasaath R., Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. "Grad-cam: Visual explanations from deep networks via gradient-based localization." In *Proceedings of the IEEE international conference on computer vision*, pp. 618-626. 2017. https://openaccess.thecvf.com/content_ICCV_2017/papers/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.pdf

Applicable to more general architectures. CAM is a special case of Grad-CAM.



(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d) Guided Grad-CAM 'Cat' (e) Occlusion map for 'Cat' (f) ResNet Grad-CAM 'Cat'

(g) Original Image (h) Guided Backprop 'Dog' (i) Grad-CAM 'Dog' (j) Guided Grad-CAM 'Dog' (k) Occlusion map for 'Dog' (l) ResNet Grad-CAM 'Dog'

Figure 1: (a) Original image with a cat and a dog. (b-f) Support for the cat category according to various visualizations for VGG-16 and ResNet. (b) Guided Backpropagation [42 highlights all contributing features. (c, f) Grad-CAM (Ours): localizes class-discriminative regions, (d) Combining (b) and (c) gives Guided Grad-CAM, which gives high resolution class-discriminative visualizations. Interestingly, the localizations achieved by our Grad-CAM technique, (c) are very similar to results from occlusion sensitivity (e while being orders of magnitude cheaper to compute. (f, l) are Grad-CAM visualizations for ResNet-18 layer. Note that in (c, f, i, l), red regions corresponds to high score fo class, while in (e, k), blue corresponds to evidence for the class. Figure best viewed in color.

Selvaraju, Ramprasaath R., Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. "Grad-cam: Visual explanations from deep networks via gradient-based localization." In *Proceedings of the IEEE international conference on computer vision*, pp. 618-626. 2017. https://openaccess.thecvf.com/content_ICCV_2017/papers/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.pdf

Pop Quizes

- What is saliency map?
- What are different methods for visualizing saliency maps and their pros/cons?

Pop Quizzes

- What is saliency map?
  - A map indicating the contribution of different pixels for the classification/regression results (what part of the image leads to the classification label or regression value?)
- What are different methods for generating saliency maps and their pros/cons
  - Occlusion (too slow!)
  - Back prop gradient with respect to the input image (noisy!)
  - Guided back prop: Back prop gradient only if the gradient is positive and the activation is positive (less noisy but not very class specific)
  - Class activation map (CAM) (very good in locating the object, but only work for certain network architecture)
  - Guided Grad CAM (more general than CAM, most promising)

Summary

- Some important extensions of conv. layers
  - Residual connection
  - Dense connection
  - Dilated convolution
- Popular classification models and transfer learning
- Image to image autoencoder for denoising
  - Upsampling: learnable interpolation filters
- Semantic Segmentation using Multiresolution Autoencoder (U-Net)
  - Combine features at multiple resolutions
- Object detection and classification (faster R-CNN, YOLO)
  - Simultaneous region detection and labeling, two pass vs. single pass
- Instance segmentation
  - Detecting regions corresponding to different objects, followed by segmentation of detected objects
- Interpretation of trained models
  - Interpretation of filters and features at different layers
  - Saliency maps for classification

What you should know?

- Be able to answer all the quizzes

Recommended Readings

- For vision applications:
  - Stanford course by Feifei Li, et al: CS231n: Convolutional Neural Networks for Visual Recognition, http://cs231n.stanford.edu/
  - Objection detection and segmentation:
    - http://cs231n.stanford.edu/slides/2020/lecture_12.pdf
  - Popular network case studies:
    - http://cs231n.stanford.edu/slides/2020/lecture_9.pdf
  - Learning GPU and PyTorch and TensorFlow:
    - http://cs231n.stanford.edu/slides/2020/lecture_6.pdf
  - Video available for previous offerings:
    - https://www.youtube.com/playlist?list=PL3FW7Lu3i5JvHM8ljYj-zLfQRF3EO8sYvhttps://www.youtube.com/playlist?list=PL3FW7Lu3i5JvHM8ljYj-zLfQRF3EO8sYv

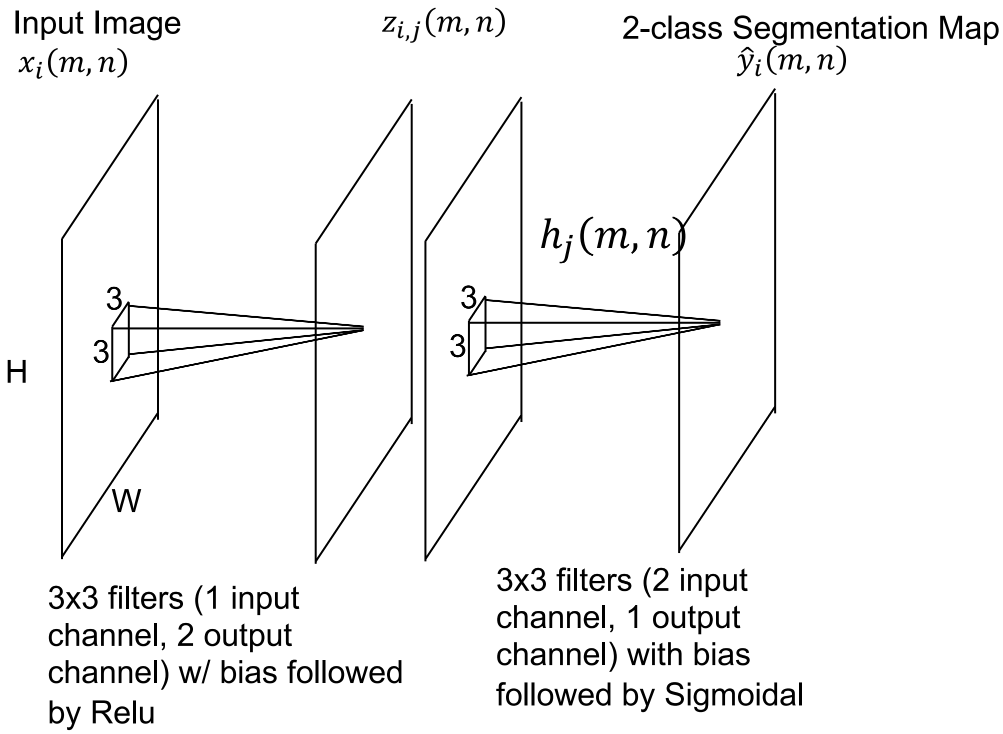
Written Assignment (1)

1. Consider the following ”very simple” segmentation network. Assuming we use SOFT Dice as the loss function.

1) Define the SOFT DICE loss function mathematically

2) Let the j-th feature map for the i-th input image after Layer 1 (before output) be described by $z_{i,j}(m,n)$. The filters in Layer 2 are denoted by $h_j(m,n)$ and the biases by b, where *j* is the index of the input feature map. Express the output $\hat{y}_i(m,n)$ as a function of $z_{i,j}(m,n), h_j(m,n), b$.

3) Derive the gradients with respect to the filter weights $h_j(m,n)$ and bias b of the last layer.



Input Image $x_i(m,n)$

$z_{i,j}(m,n)$

2-class Segmentation Map $\hat{y}_i(m,n)$

$h_j(m,n)$

3x3 filters (1 input channel, 2 output channel) w/ bias followed by Relu

3x3 filters (2 input channel, 1 output channel) with bias followed by Sigmoidal

Written Assignment (2)

2. Consider a network with 3 convolutional layers. 1) Suppose each layer uses 3x3 filters without dilation, what is the receptive field size of each output pixel? 2) Suppose the first layer uses 3x3 filter without dilation, and the second and the third layer each uses 3x3 filters with a dilation rate of 2. What is the receptive field of each output pixel? 3) Now suppose each layer uses 3x3 filters without dilation, but there is a 2x2 downsampling after each layer layer. What is the receptive field? Discuss the pros and cons of these methods.

3. Why does the skip connection facilitates gradient backpropagation?

4. Why does U-Net helps to exploit both global and local information in its decision? In the final out layer, the input consists of skipped connection from a high resolution layer and upsampled signals from a lower layer. Which one brings global information, which one brings local information?