EE482/682: DSP APPLICATIONS

CNNS AND DEEP COMPUTER VISION

Géron Chapter 14

http://www.ee.unlv.edu/~b1morris/ee482

NOTE

- Slides follow Geron's Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow

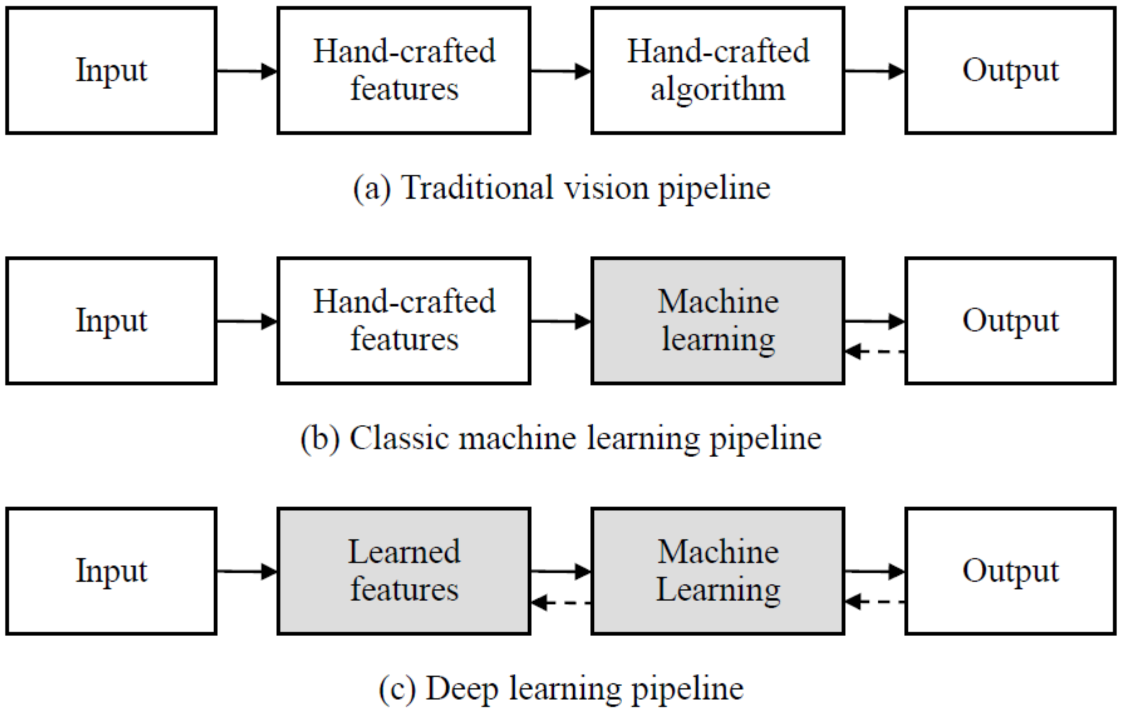
- Additional Deep Detections slides from [Object Detection with Deep Learning: A Review]

OUTLINE

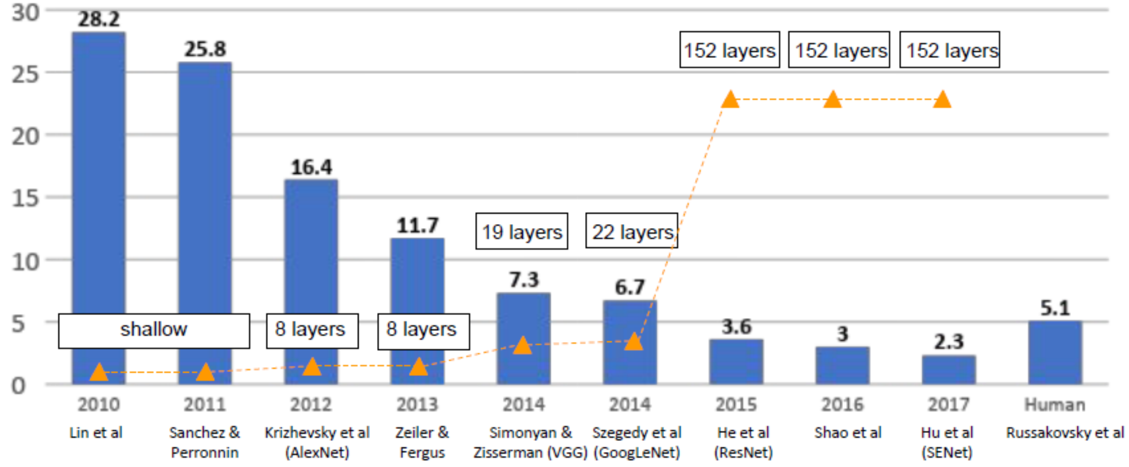
- Biological Inspiration
- Convolutional Layers
- Pooling Layers
- CNN Architectures
- Object Detection Survey
- Semantic Segmentation Survey

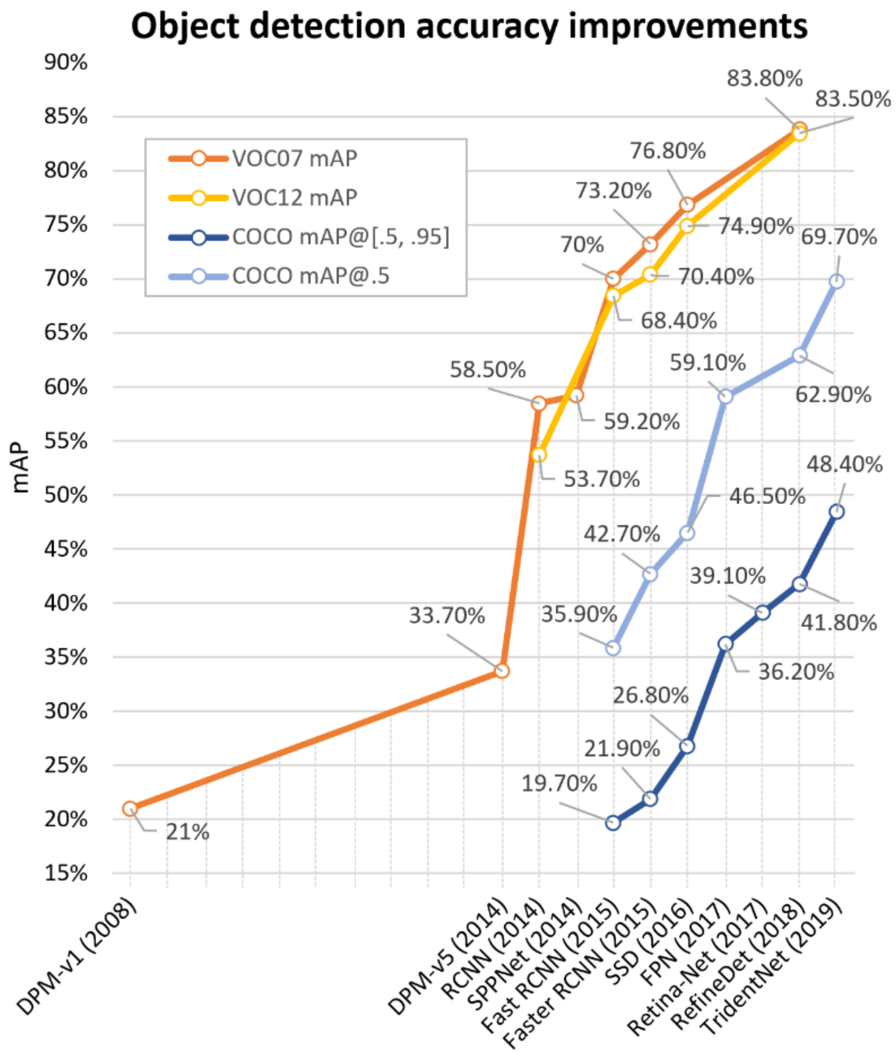
EVOLUTION OF COMPUTER VISION

- Classical vision
  - Hand-crafted features and algorithm based on expert knowledge
- Classical machine learning
  - Hand-crafted features (pre-processing) but ML for classification
- Deep learning
  - Both features and classification are learned
  - End-to-end training (from pixels to output)



DEEP CNN DOMINANCE IN CV



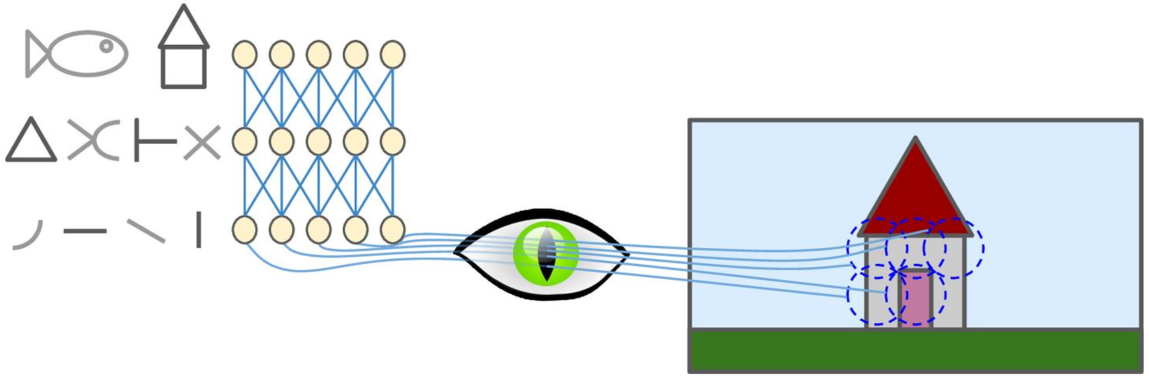


Li, Johnson, and Yeung, 2019

Zou et al., "Object Detection in 20 Years: A survey, 2019

ARCHITECTURE OF THE VISUAL CORTEX

- Modern CV is inspired by human vision (sensory modules)
- Hubel and Wiesel showed that neurons in the visual cortex had a small local receptive field
  - Only reacted to stimuli in a limited region of visual field (blue dashed circles)
- Lower-level neurons with simple pattern response (e.g. lines of specific orientation)
- Higher-level neurons with larger receptive field and combination of lower-level patterns
  - Neurons at higher-levels only connected to few at lower-level

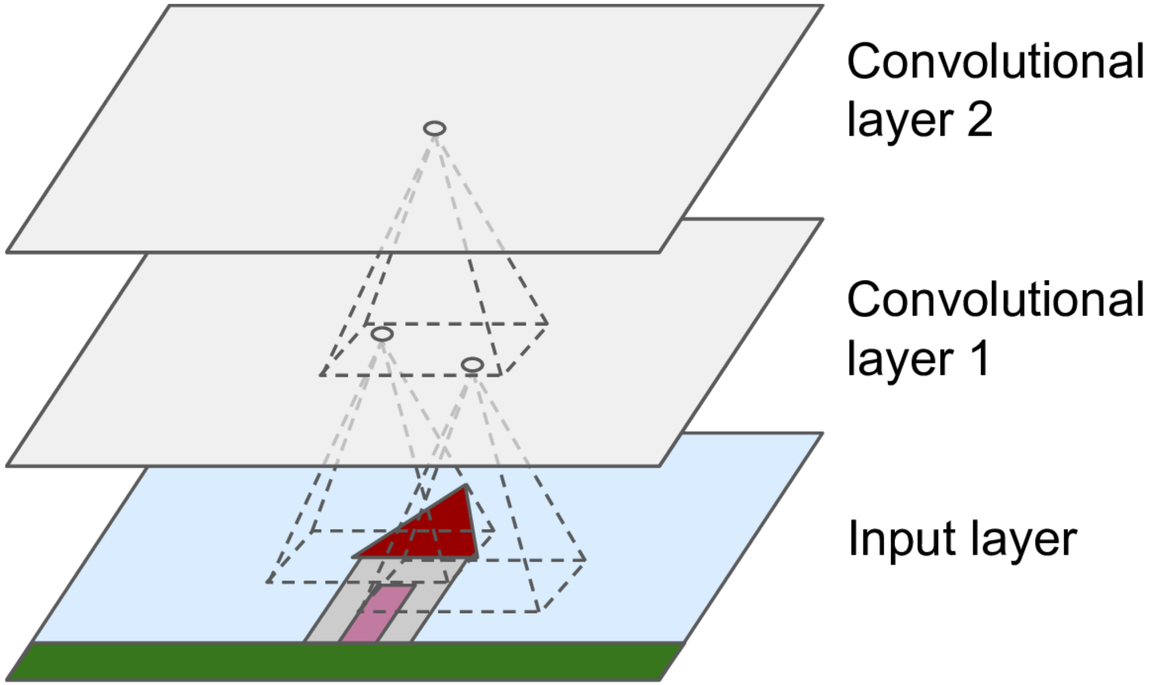


CONVOLUTIONAL NEURAL NETWORK

- Stacked neuron architecture enables detection of complex patterns in any area of the visual field → convolutional neural networks (CNNs)
- Led to LeNet-5 architecture by Yann LeCunn for handwritten number recognition (MNIST)
  - Fully connected layers and sigmoid activations
  - Convolutional layers and pooling layers
- Why not fully connected layers for images?
  - Even small images have large number of pixels resulting in huge networks
  - CNNs solve this with partial connected layers and weight sharing

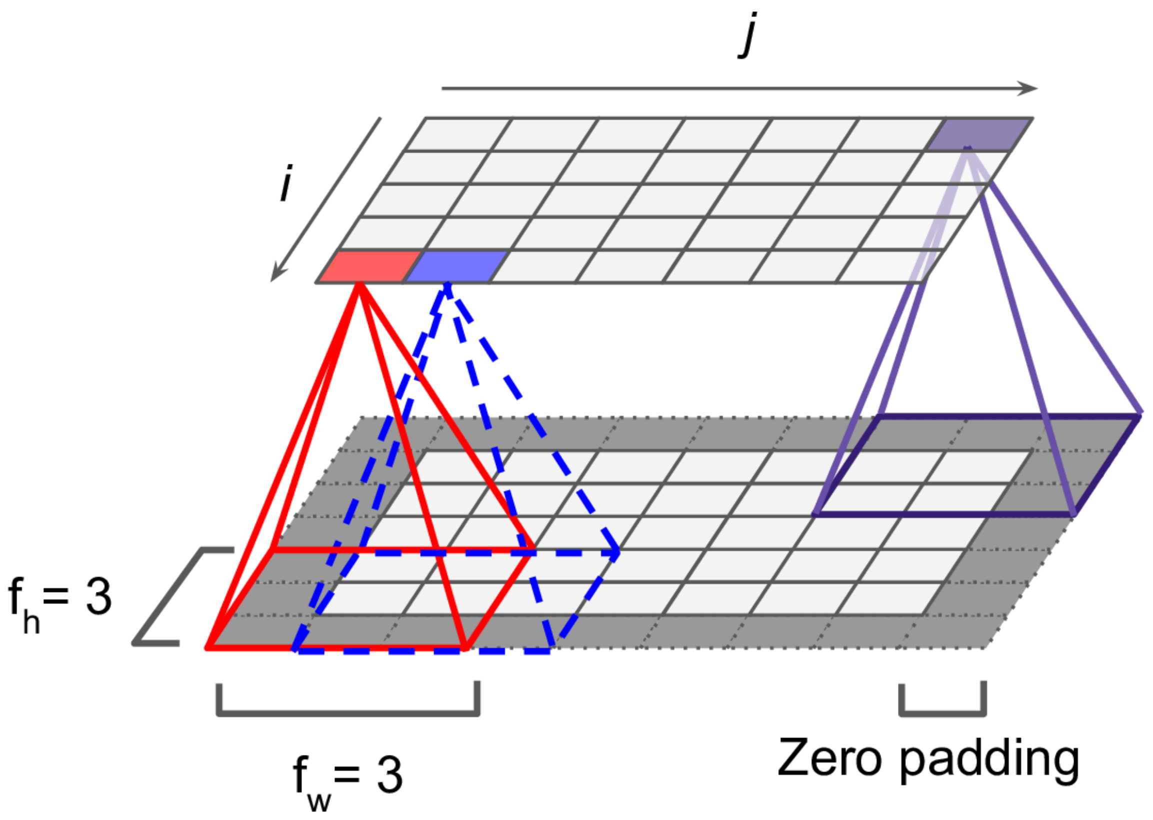
CONVOLUTIONAL LAYERS

- Neurons in the first layer are not connected to every single pixel in input image
  - Connected to receptive field
  - Stacked receptive field approach
- Hierarchical structure
  - First layer – small low-level features
  - Higher-levels – assemble lower-level features into higher-level features
  - Structure is common in real-world images



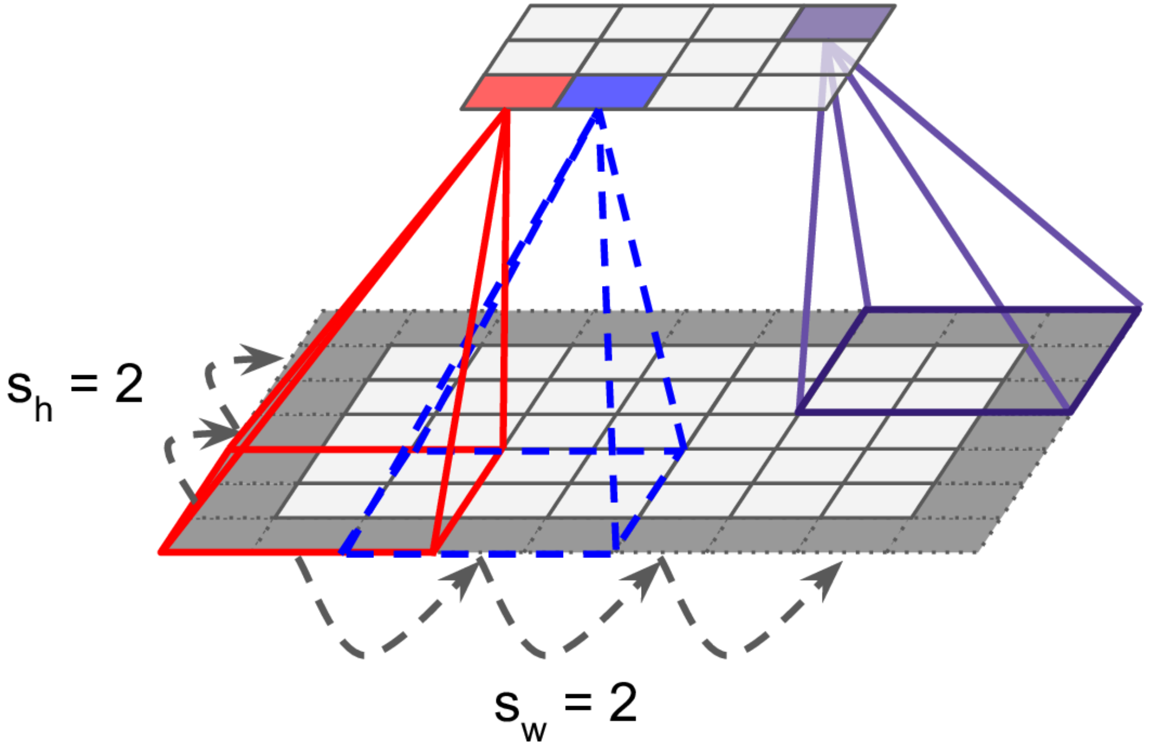
CONVOLUTIONAL LAYER CONNECTIONS

- Note: the actual operation performed is cross-correlation (no-flipping)
- Neuron (row, column) (i, j) is connected to neurons in previous layer within receptive field
  - Row $[i, i + f_h - 1]$
    - f_h - height of receptive field
  - Column $[j, j + f_w - 1]$
    - f_w - width of receptive field
  - Note: this is a causal filter though shown as symmetric
- Zero padding used to keep output/input layers of same size



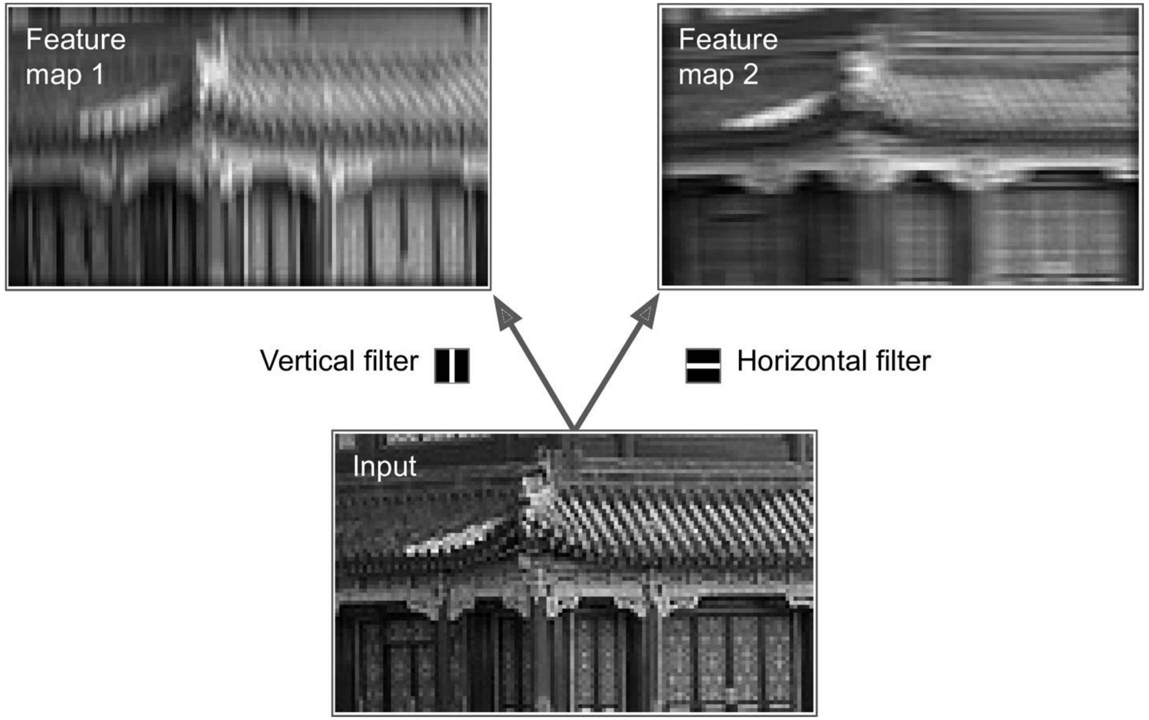
CONVOLUTIONAL LAYERS STRIDE

- Stride can be used to connect a large input layer to smaller output layer
- Change the spacing the of the receptive field
- Dramatically reduce model computational complexity (squared)
  - Height and width stride can be different

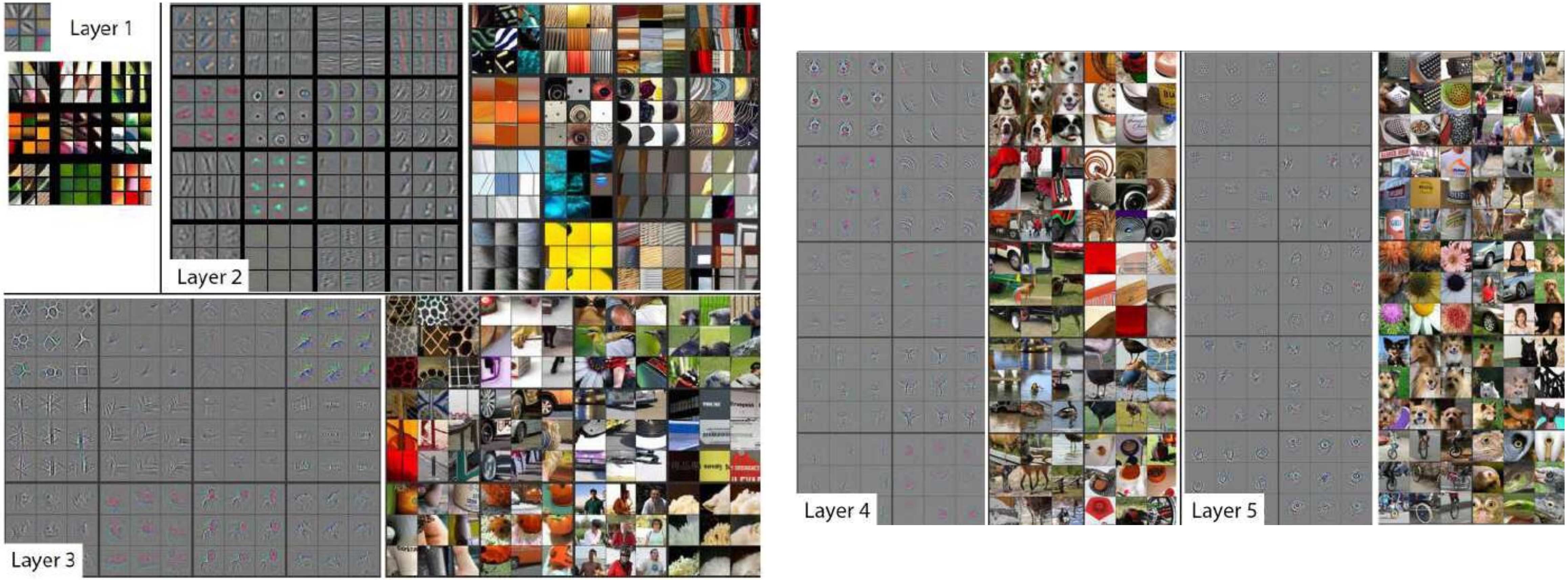


FILTERS

- Filters = convolutional kernels
- Size of the kernel is the receptive field for the neuron
- Feature map – output of the "convolution" operation
  - Highlights areas in an image that activate the filter most
- For CNNs, the filters are not defined manually!
  - Learn most useful filters for a task
  - Higher layers will learn to combine into more complex patterns



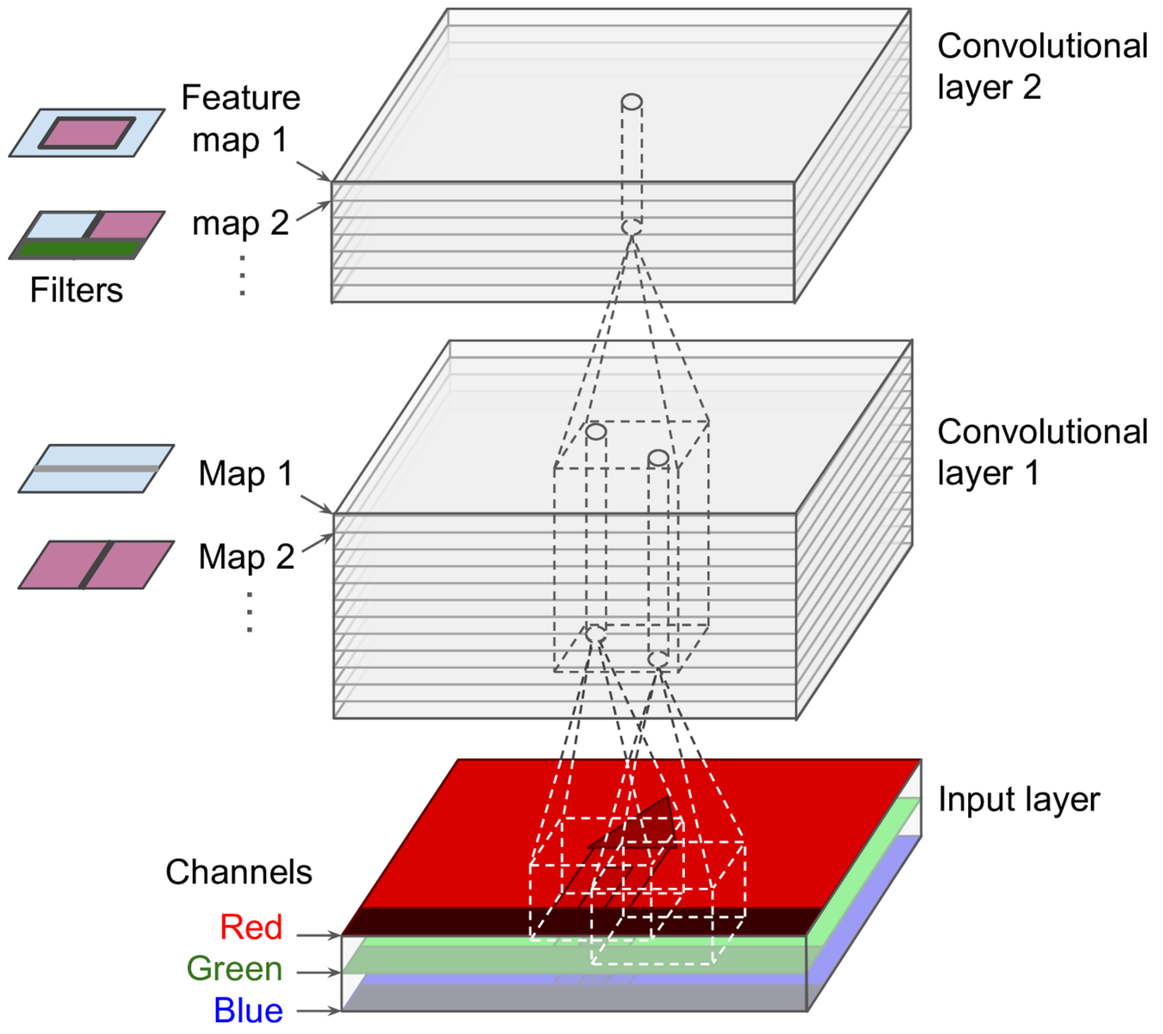
VISUALIZING WEIGHTS AND FEATURES



See Szeliski 2e, Ch 5.4.5

STACKING MULTIPLE FEATURE MAPS I

- Each convolution layer has multiple filters
  - Stacked 3D output (1 feature map for each filter)
- Each neuron in a feature map shares the same parameters (weights and bias)
- Neurons in different feature maps use different parameters
- Neuron's receptive field applies to all feature maps of previous layer
- Note input images often have multiple sublayers (channels)



STACKING MULTIPLE FEATURE MAPS II

- Output of a neuron in a convolutional layer

$$z_{i,j,k} = b_k + \sum_{u=0}^{f_h-1} \sum_{v=0}^{f_w-1} \sum_{k=0}^{f_{n'}-1} x_{i',j',k'} \times w_{u,v,k',k}$$

$$\begin{cases} i' = i \times s_h + u \\ j' = j \times s_w + v \end{cases}$$

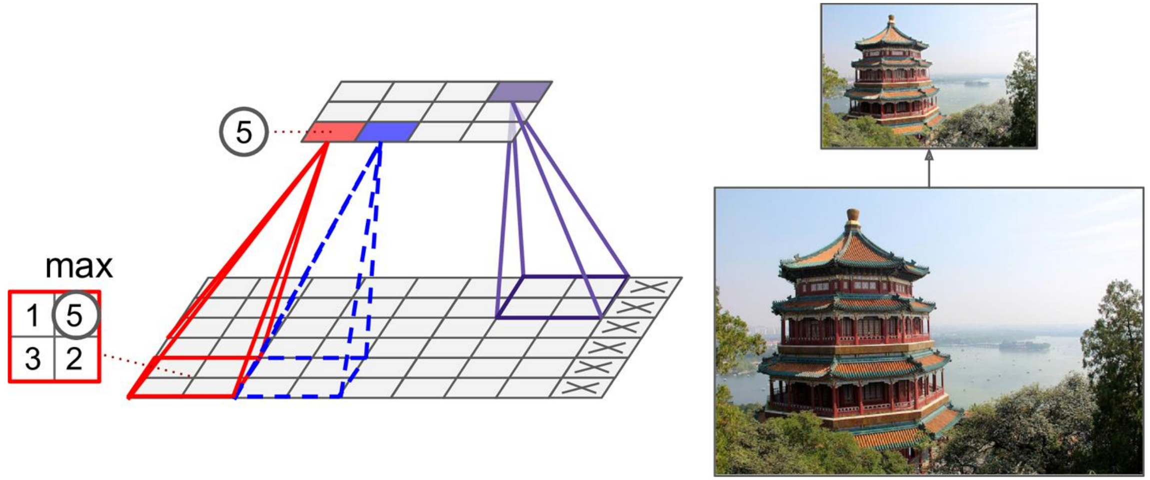
- $z_{i,j,k}$ - output of neuron in row i, column, j, in feature map k of the convolutional layer l
- b_k - bias term for feature map k (in layer l)
  - Tweaks overall brightness of feature map k
- s_h, s_w - vertical and horizontal strides
- f_h, f_w - height and width of receptive field (kernel)
- $f_{n'}$ - number of feature maps in previous (lower layer)
- $x_{i',j',k'}$ - output of neuron located in layer $l-1$, row i', column j', feature map k
- $w_{u,v,k',k}$ - connection weight between any neuron in feature map k of the layer l and its input located at row u, column v (relative to the neuron's receptive field), and feature map k'

MEMORY REQUIREMENTS

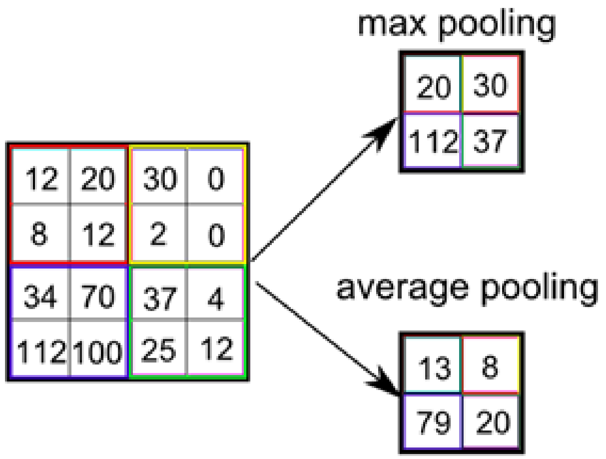
- Though much smaller the fully connected networks, CNNs still use significant amount of RAM
- During training, the reverse pass of backpropagation requires all the intermediate values computed during the forward pass
  - Need to have enough for all layers in the network
  - Forward pass can release memory after each layer is computed (only two consecutive layers required)
- Out-of-memory error
  - Reduce mini-batch size, increase stride, remove layers, change precision (16-bit vs 32-bit floats or use int), or distribute the CNN across devices

POOLING LAYERS

- Subsample input in order to reduce computational load, memory usage, and number of parameters (reduce risk of overfitting)
- Aggregate over the receptive field
  - Aggregate functions such as max (most popular) or mean
  - Max tends to work better by preserving only the strongest feature → cleaner signal, more invariance, less compute
- Stride gives downsampling
- Pooling kernel size can be even

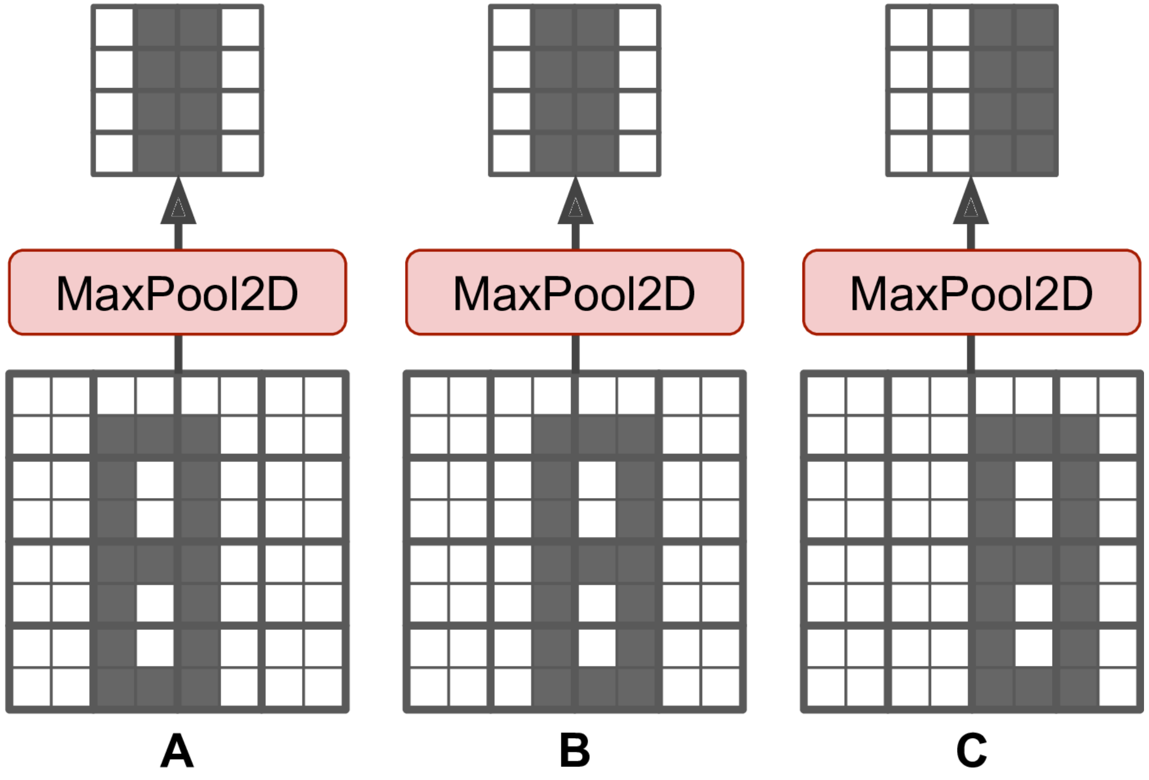


Max pooling layers (2x2 kernel, stride=2, no padding)



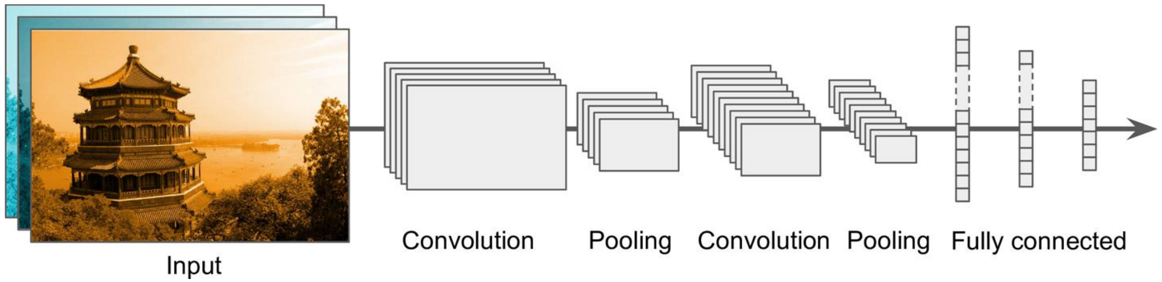
POOLING LAYERS INVARIANCE

- Introduces some level of invariance to small translations
  - Small image shifts result in same response
  - Additionally small invariance to rotation and scale with max pool
- Max pool every few CNN layers for invariance at larger scale
  - Useful when task should be invariant (e.g. image classification)
- Drawbacks
  - Destructive – 2x2, stride 2 drops 75% of input values
  - Invariance not always desirable (e.g. semantic segmentation should have equivariance)



CNN ARCHITECTURES

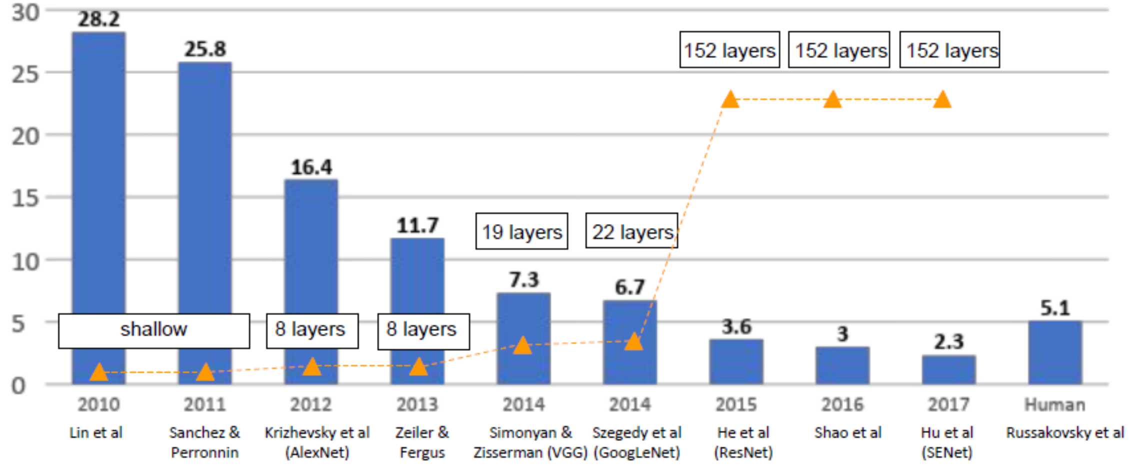
- Typical CNN architecture
  - Stack a few convolutional layers (each followed by ReLU layer for non-linearity)
  - Pooling layer
  - Repeat Conv + ReLU + Pool
  - Top layers are regular feedforward neural network which is usually fully connected layers (+ReLUs)
  - Final layer outputs the prediction (e.g. softmax for class probabilities)



- Input kernel can be larger since generally only 3 sublayers (RGB channels)
- Conv layers use stacked small 3x3 kernels since it is more computationally efficient and perform better than larger
- Number of filters increases at higher layers
  - Few low-level patterns, but more ways to combine
  - Double #filters after pooling (stride 2)
- Flatten conv output before fully connected dense layer
  - Add dropout to avoid overfitting

ILSVRC IMAGENET CHALLENGE

- Variants of basic CNN architecture have been developed
- Benchmark with ImageNet Challenge
  - Large scale with 1M images and 1000 classes
  - Much more complicated than any benchmark at the time (~2010)
- Dramatic drop in top-five error from 26% to 2.3% in 6 years
  - Bigger is better



LENET-5

- Network of Yann LeCun (1998) [NYU] designed for handwritten digit recognition (MNIST)
- Images normalized at input
- No padding → smaller size each layer
- Average pool has learnable coefficient and bias term
- Limited C3-S2 map connections
- Output square Euclidean distance
  - Similar cross-entropy

Table 14-1. LeNet-5 architecture

Layer	Type	Maps	Size	Kernel size	Stride	Activation
Out	Fully connected	–	10	–	–	RBF
F6	Fully connected	–	84	–	–	tanh
C5	Convolution	120	1 × 1	5 × 5	1	tanh
S4	Avg pooling	16	5 × 5	2 × 2	2	tanh
C3	Convolution	16	10 × 10	5 × 5	1	tanh
S2	Avg pooling	6	14 × 14	2 × 2	2	tanh
C1	Convolution	6	28 × 28	5 × 5	1	tanh
In	Input	1	32 × 32	–	–	–

http://yann.lecun.com/exdb/lenet/index.html

ALEXNET

- 2013 ImageNet winner
  - 17% top-5 error rate (26% for 2nd place)
  - Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton [U Toronto]
- Similar to LeNet-5 but larger and deeper
- First to stack convolutional layers directly on top of one another (no pooling in between)
- To reduce overfitting
  - 50% dropout of layers F9 and F10
  - Data augmentation
- Local response normalization used to inhibit neighboring feature maps
  - Encourage different feature maps to specialize, push neighbors apart, and improve generalization

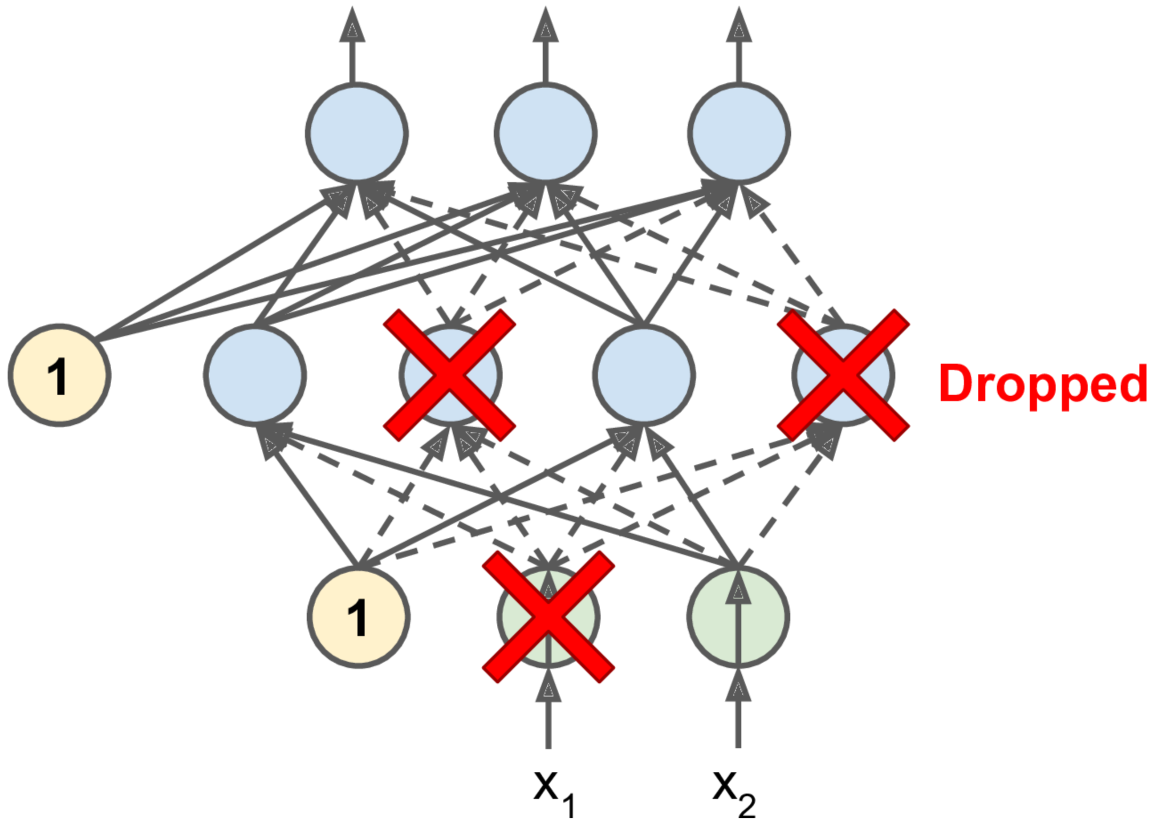
Table 14-2. AlexNet architecture

Layer	Type	Maps	Size	Kernel size	Stride	Padding	Activation
Out	Fully connected	–	1,000	–	–	–	Softmax
F10	Fully connected	–	4,096	–	–	–	ReLU
F9	Fully connected	–	4,096	–	–	–	ReLU
S8	Max pooling	256	6 × 6	3 × 3	2	valid	–
C7	Convolution	256	13 × 13	3 × 3	1	same	ReLU
C6	Convolution	384	13 × 13	3 × 3	1	same	ReLU
C5	Convolution	384	13 × 13	3 × 3	1	same	ReLU
S4	Max pooling	256	13 × 13	3 × 3	2	valid	–
C3	Convolution	256	27 × 27	5 × 5	1	same	ReLU
S2	Max pooling	96	27 × 27	3 × 3	2	valid	–
C1	Convolution	96	55 × 55	11 × 11	4	valid	ReLU
In	Input	3 (RGB)	227 × 227	–	–	–	–

ZF Net is an AlexNet variant with tweaked hyperparameters

DROPOUT

- Popular technique from Hinton 2012 and Srivastava et al. 2014
  - 1-2% accuracy boost (even SOTA)
- At each training step, a neuron has a probability of being ignored (dropped out)
  - Neuron can be active during next training step
- Dropout rate generally between 10-50%
  - 20-30% for recurrent neural networks
  - 40-50% for CNNs
- Forces networks to diversify
  - Neurons cannot co-adapt with neighbors
  - Cannot rely only an a few input neurons
  - Less sensitive to slights changes in input
  - ~Average of many networks



DATA AUGMENTATION

- Artificially increase training dataset size by generating realistic variants of training instances
  - Ideally, shouldn't be able to distinguish real from augmented example
- Reduces overfitting (regularization technique)
- Common augmentations
  - Small shifts, rotation, resize (scaling)
  - Horizontal flip – orientation invariance
  - Vary contrast – lighting condition invariance

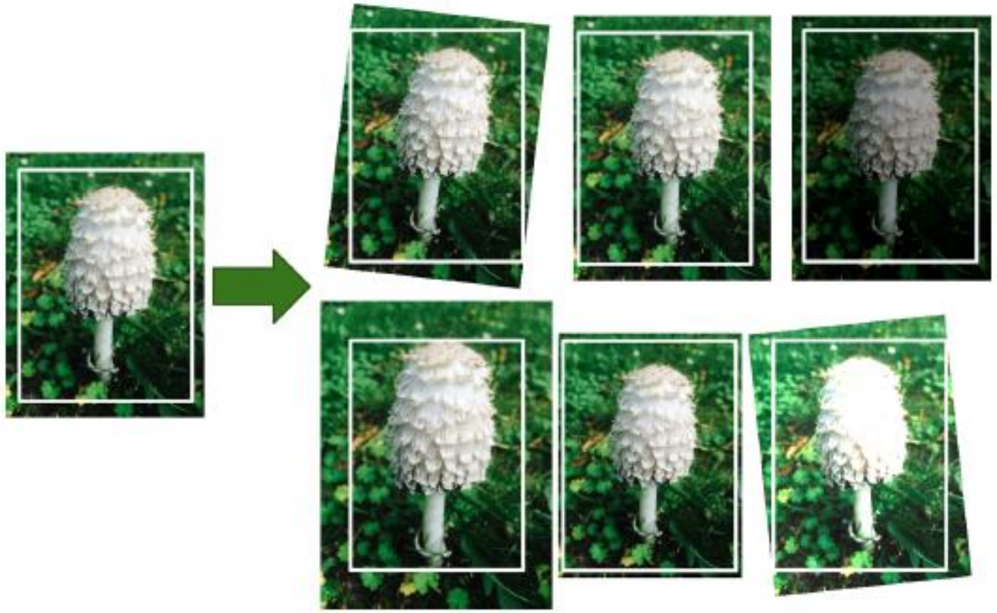
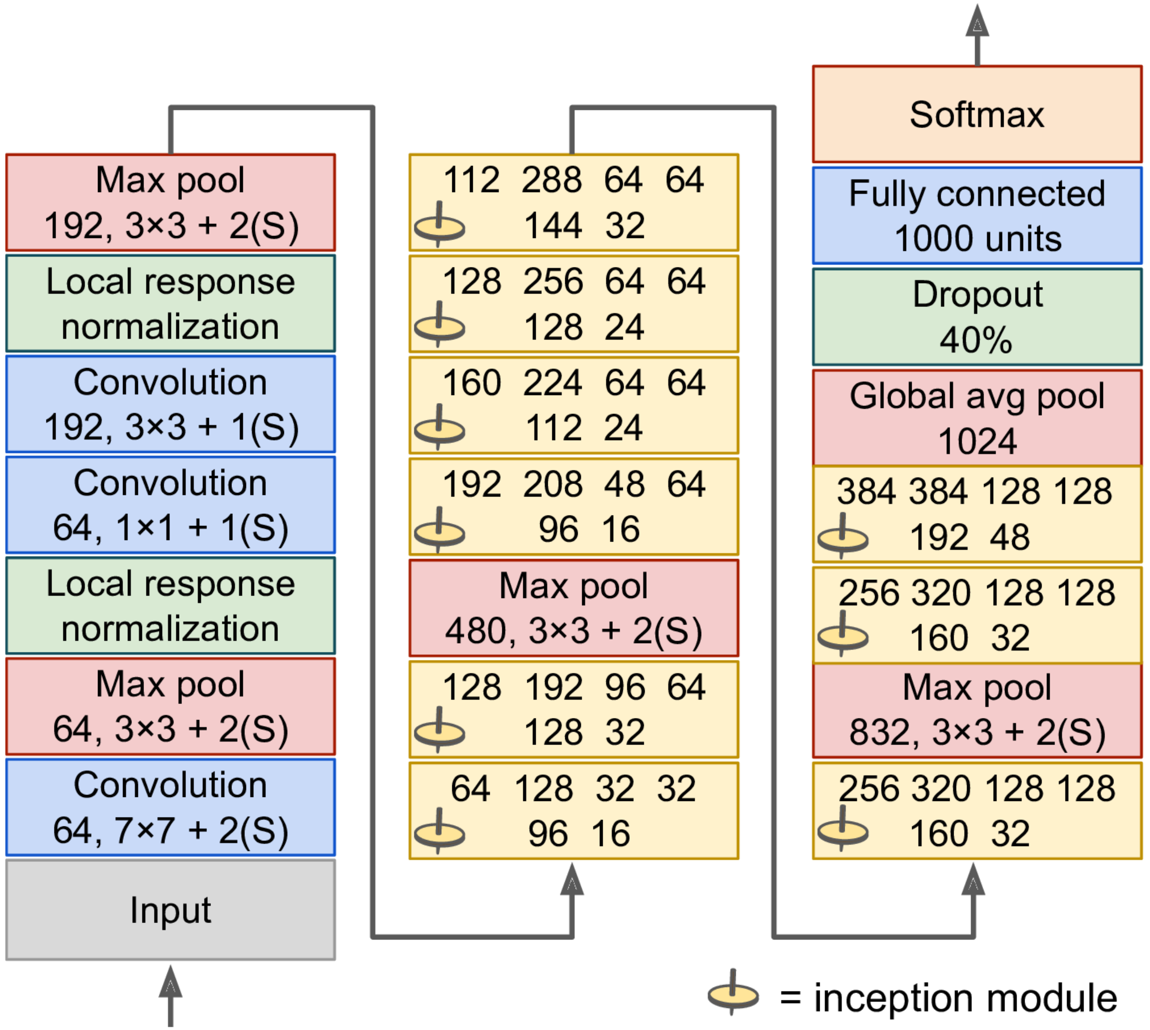


Figure 14-12. Generating new training instances from existing ones

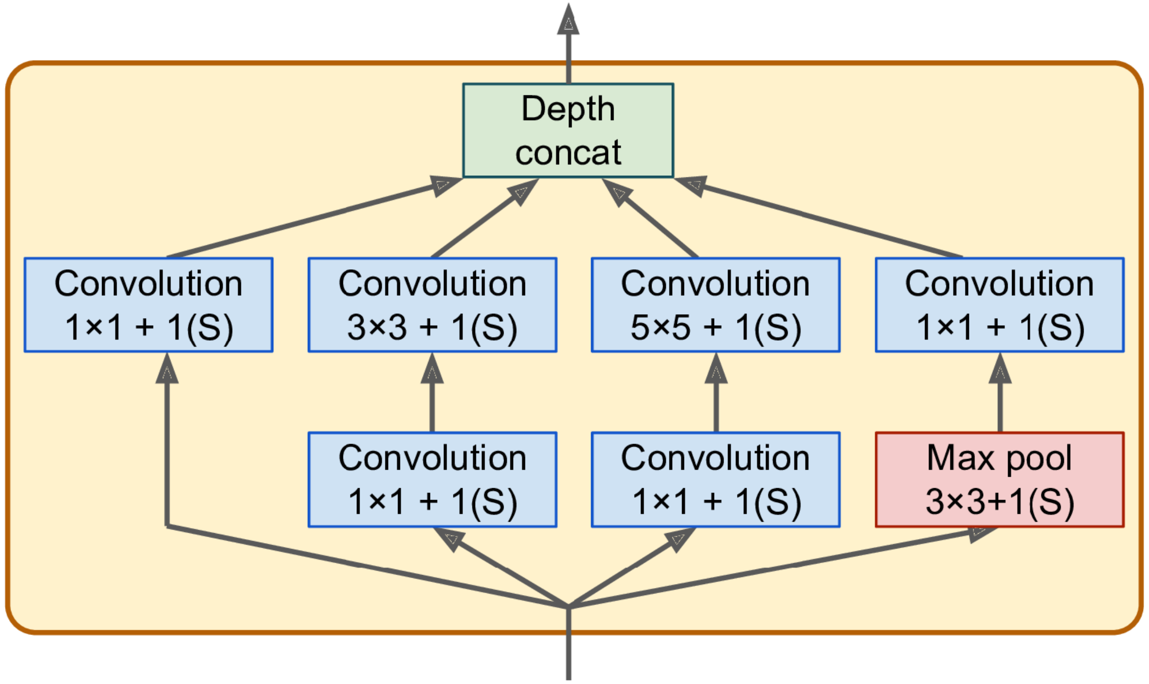
GOOGLENET (INCEPTION)

- 2014 ILSVRC Winner
  - <7% top-5 error rate
  - Christian Szegedy et al. [Google]
  - Current versions Inception-v3 and Inception-v4 (GoogLeNet + ResNet)
- Much deeper architecture than previous CNN (large stack)
  - Much fewer parameters (6M vs. 60M AlexNet)
- Inception layers for parameter efficiency
- Use of 1x1 convolutions as a bottleneck layers
- Local response normalization to learn a wide variety of features
- Classification task with multiple (max) pool to reduce size (avg. final 7x7 map)
  - No need for multiple fully connected (FC) layers to save parameters



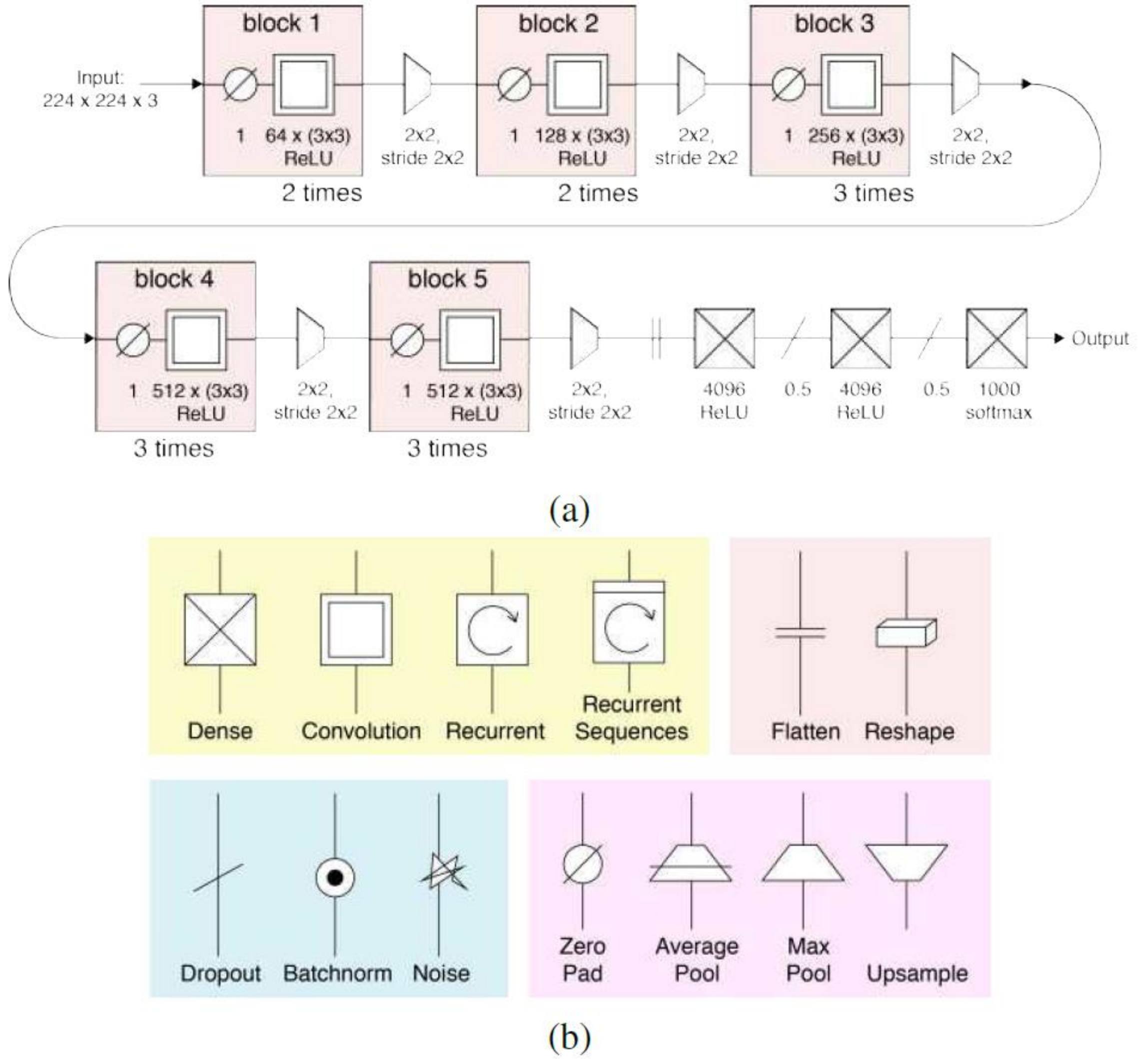
INCEPTION MODULE

- Parallel convolutions
  - 3x3+1(S) = 3x3 kernel, stride 1, “same” padding
  - All use ReLU activation
- 2nd convolution layer
  - Different kernel size for patterns at different scale
  - Stacked conv for more complex patterns than single linear convolution
- Depth concat
  - All layers have the same outputs size
  - Stack 2nd layer outputs depthwise
- 1x1 bottleneck layers
  - Fewer output than input dimension
  - Fewer parameters, faster training, improved generalization
  - Not spatial but depth patterns



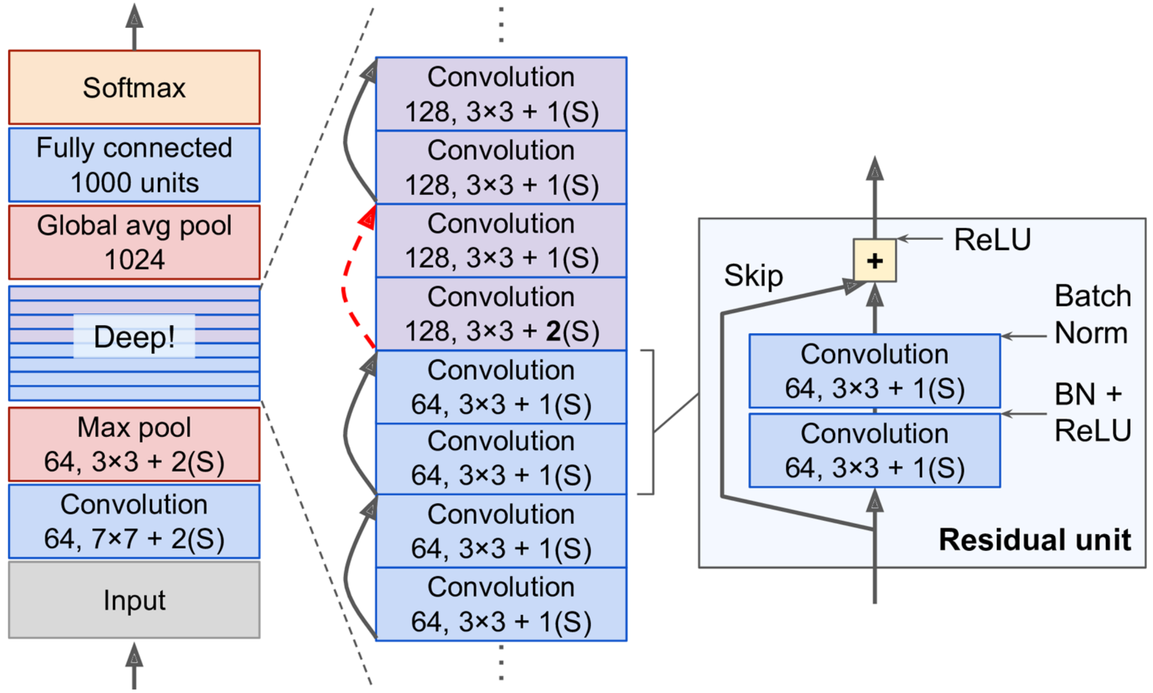
VGGNET

- 2014 ILSVRC runner-up
  - Simonyan and Zisserman [Oxford]
- Classical architecture
  - Stacked 2-3 conv + pool layers
  - Variants of 16 or 19 conv layers
  - 3 FC classification layers
- Used many 3x3 filters



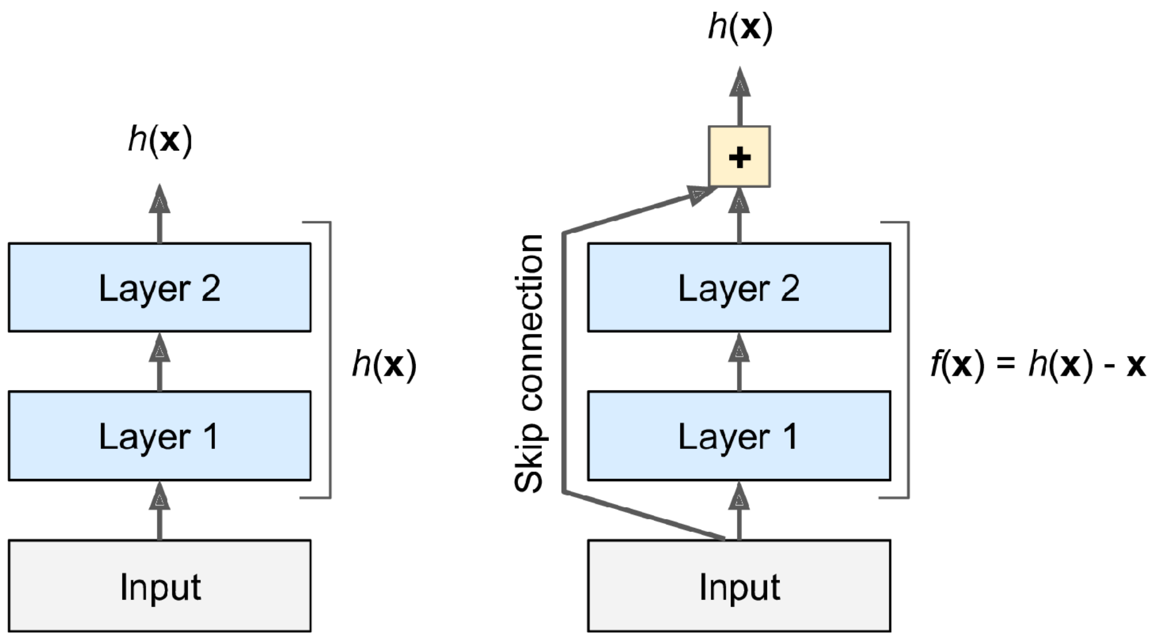
RESNET

- 2015 ILSVRC winner
  - <3.6% top-5 error rate
  - Kaiming He et. Al [Microsoft]
- Deeper with fewer parameters
  - 152 layer winner
  - Variants of 34, 50, and 101 layers
- Skip (shortcut) connections
  - Signal passed into up one layer and a further layers ahead
  - Build network on residual units (RUs)
- Batch normalization (pg 338)
  - Better gradient conditioning (vanishing gradient)
  - Standardize inputs then rescales and offsets
  - Acts as a regularizer (e.g. no need for dropout)



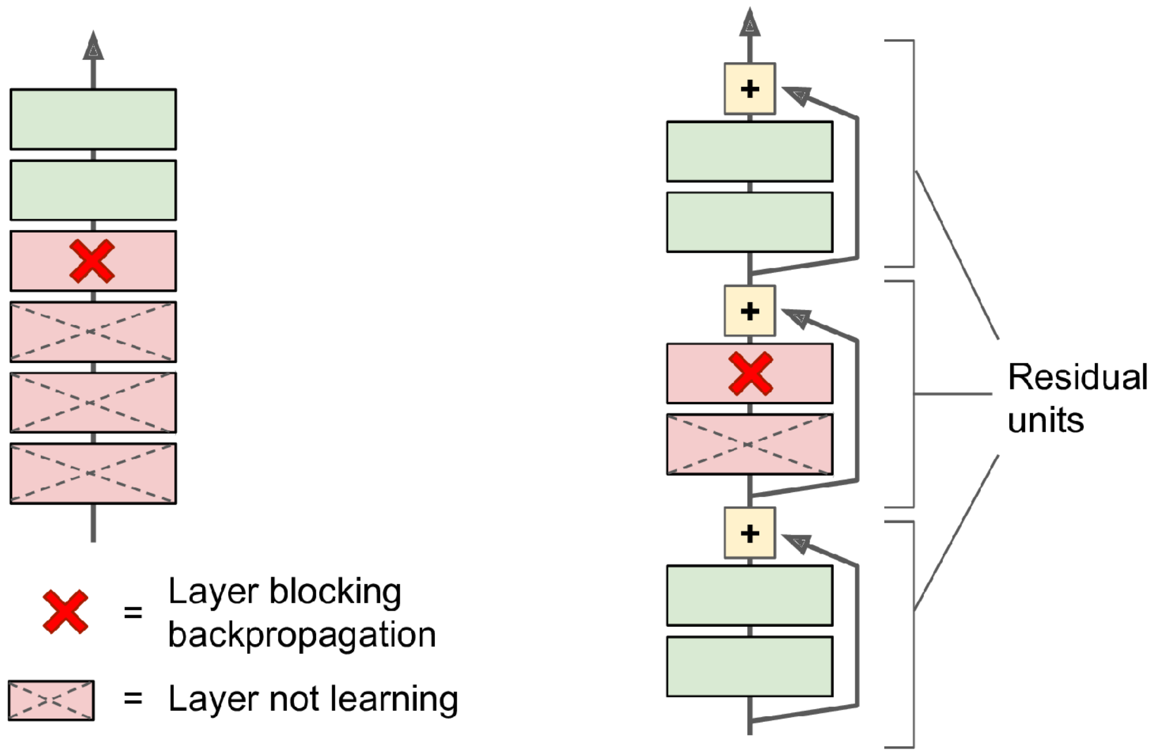
RESIDUAL LEARNING I

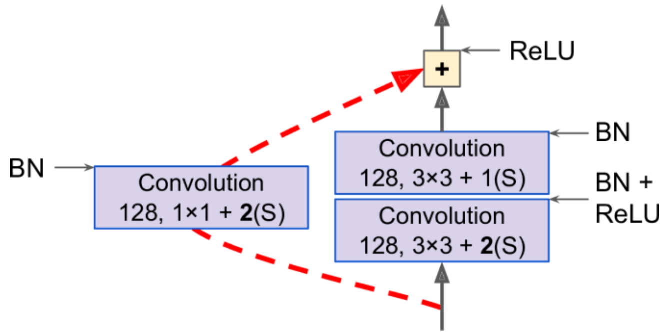
- Signal feeding layer is also added to the output of a layer higher in the stack
- Instead of modeling function $h(x)$, it models $f(x) = h(x) - x$
- Faster weight update (0 initialization)
  - Regular networks output 0
  - Skip connection copies input (identity function)



RESIDUAL LEARNING II

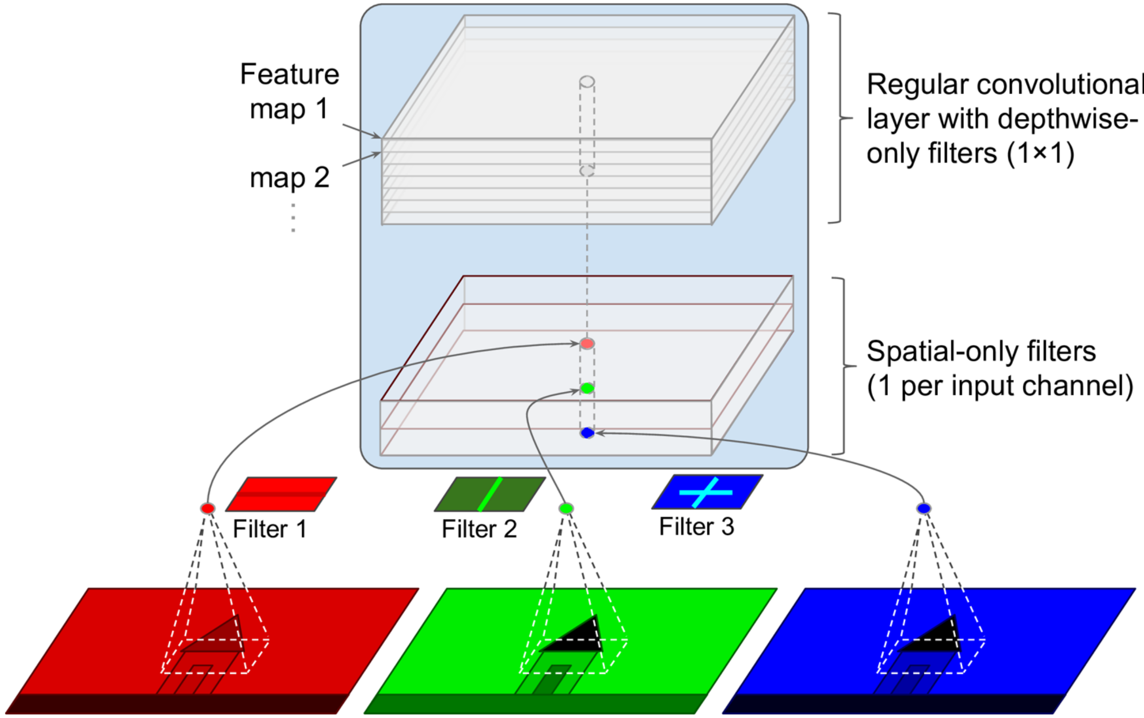
- Skip connection bypass layer blocking
  - Input signal can propagate to higher levels
  - Can train layers even if lower layers have not started learning yet
- Feature map size and depth change
  - Skip connection prevents direct addition after resize
  - 1x1 convolution, stride 2, and output matched kernel size





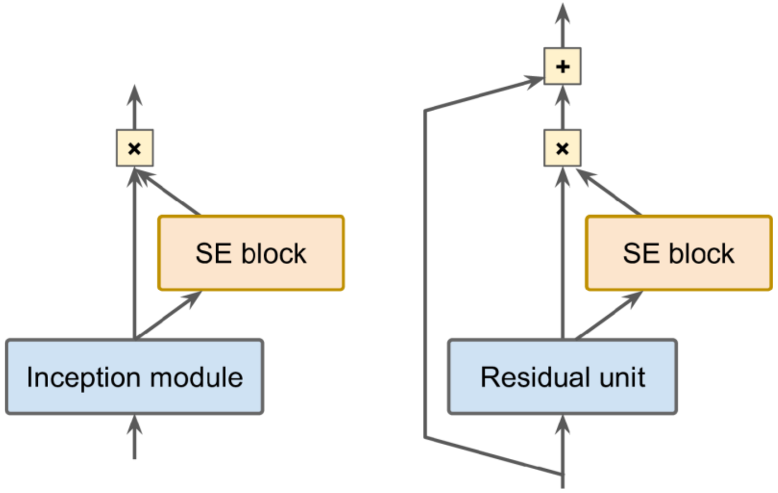
XCEPTION

- GoogLeNet variant
  - Combines GoogLeNet + ResNet
  - Inception modules replaced with depthwise separable convolution layer
  - Chollet 2016 (Keras author)
- Separable convolution layer
  - Separate spatial and depth
  - 1 spatial filter per input channel
  - Use on layers with many feature channels (not on input/early layers)
  - Fewer parameters, less memory, fewer computations, and generally perform better

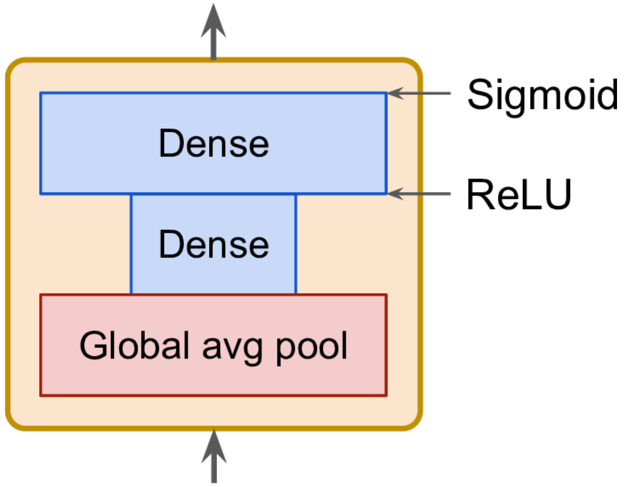


SENET

- 2018 ILSVRC winner
  - Squeeze-and-Excitation Network
  - 2.25% top-5 error rate
  - Built on Inception (SE-Inception) and ResNets (SE-ResNet)

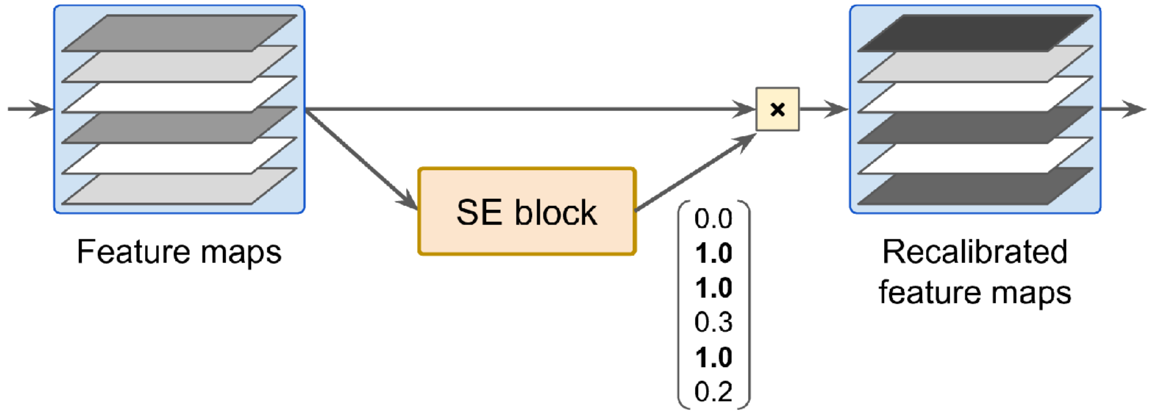


- SE block
  - Global average pool: mean of each feature map
  - "Squeeze" (bottleneck)
    - Dramatically reduce number of maps for low dimensional embedding of feature distribution
    - Force SE block to learn general representations of feature combinations
  - Output: recalibration vector (boost normally co-occurring features)



SE BLOCK

- Analyze output of attached unit to learn features that are usually most active together (depth search)
- Recognizes features that respond together (mouth, nose, eyes) and boosts features that are missing/low response (e.g. eyes)
- Recalibration steps solves ambiguity when feature is confused with something else



PRETRAINED MODELS AND TRANSFER LEARNING

- Don't implement models from scratch by hand, use existing implementations
  - Known as backbone network
- Models pretrained on ImageNet
  - Good general features
  - Models expect specific size and pre-processing (e.g. normalization)
- Only requires a few lines of code

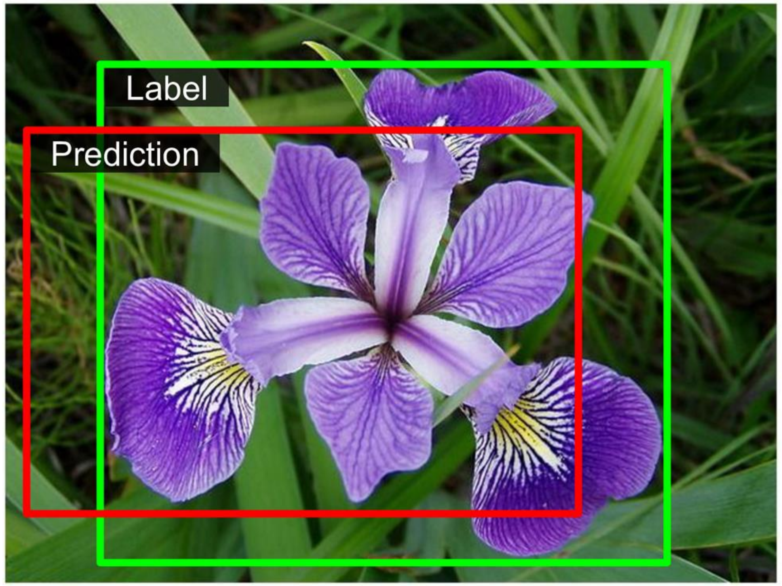
- Transfer learning
  - Utilize strong backbone and adjust last layers for a specific task
- Useful when not working with ImageNet classes (all the time) and with limited training data
- Initialize network with ImageNet weights and only train higher layers (e.g. classification or minimal conv)

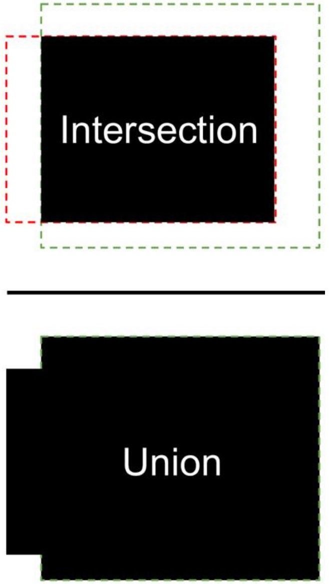
REMINDER: RECOGNITION TASKS

- Recognition/Classification
- Object Detection
- Semantic Segmentation

CLASSIFICATION AND LOCALIZATION

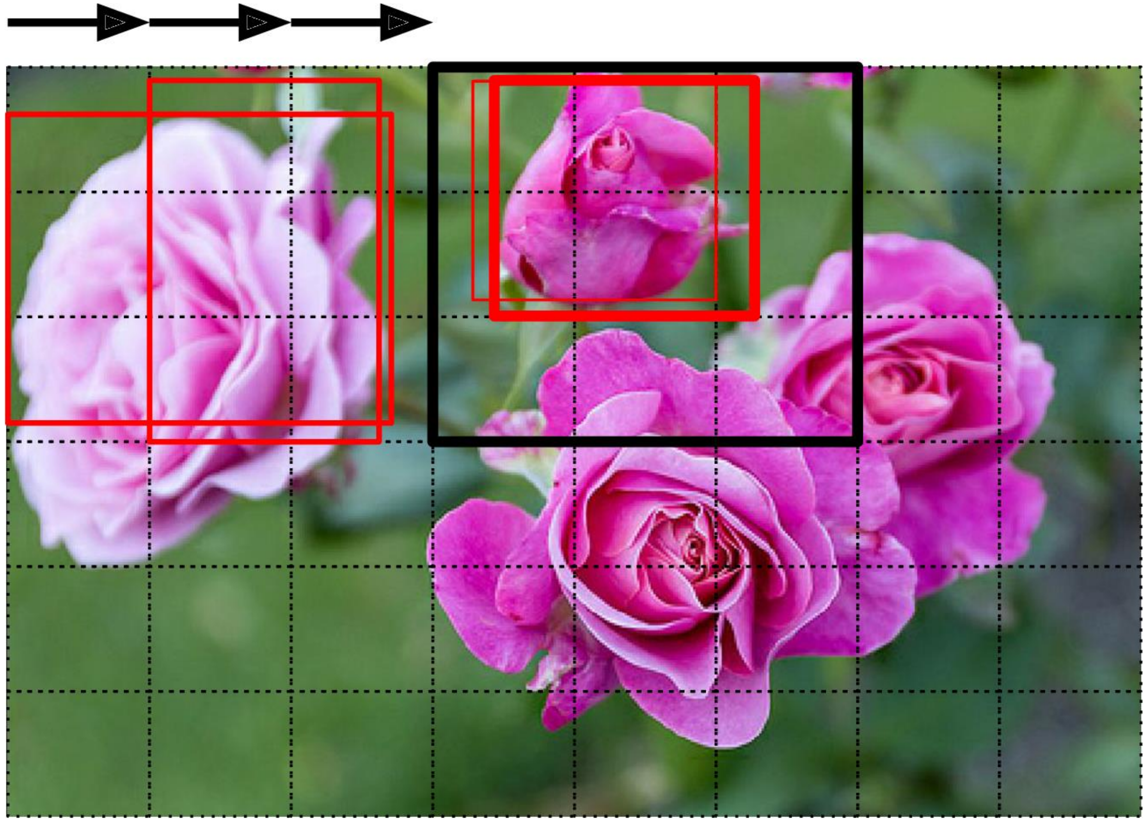
- Classification – identify the image class
- Localization – provide a bounding box for the image class
  - Expressed as a regression task [x, y, w, h]
  - Assumption of a single object per image
  - Much of the work is in labeling the data with bounding boxes
    - Many tools exist (e.g. VGG Image Annotator, LabelImg, OpenLabeler, ImgLab, LabelBox, Suervisely)
  - Evaluated with intersection over union (IoU) the overlap





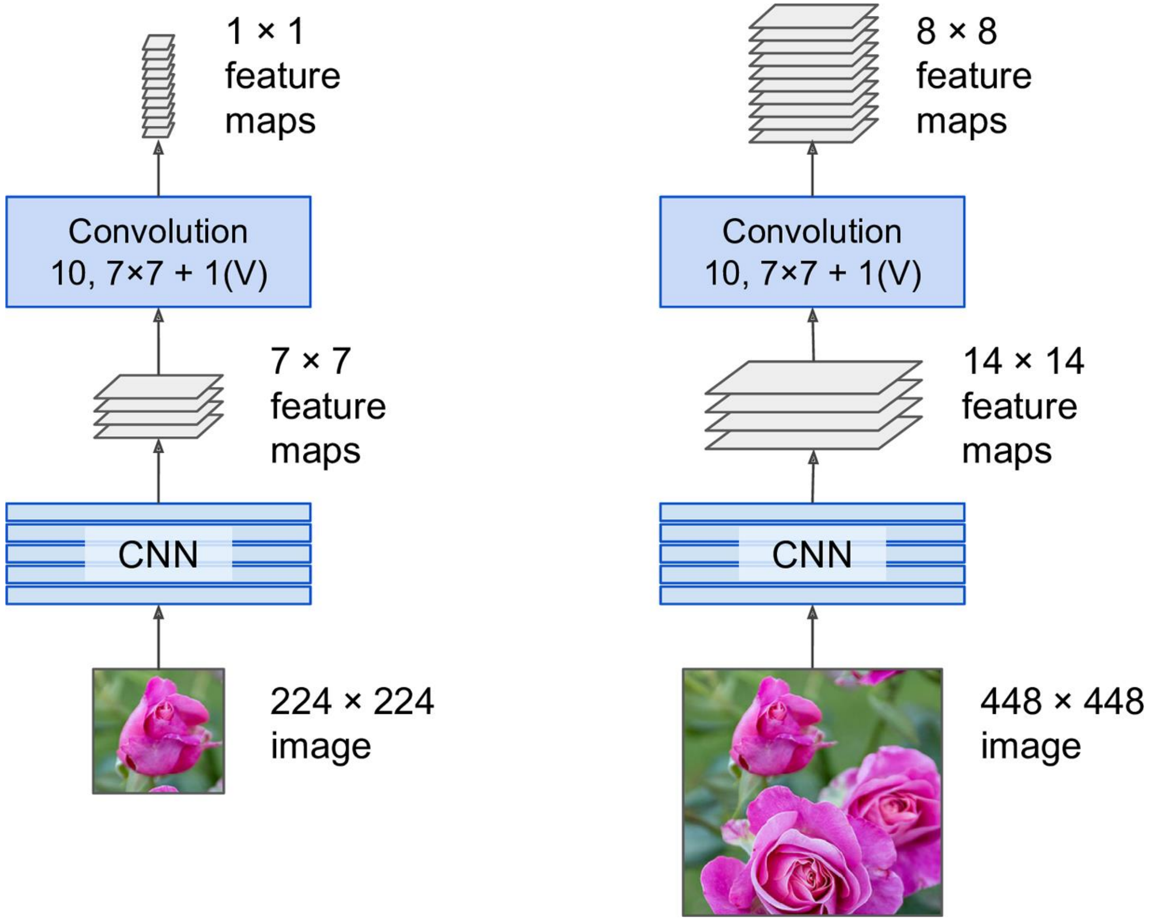
OBJECT DETECTION

- Task of classifying and localizing multiple objects in an image
- Early attempts used a sliding window
  - Run classification CNN over each window in the image
  - Need search at scale (multiple passes)
  - Get multiple responses to same object → NMS
    - Objectness score to remove responses
    - Merge responses with high IoU



FULLY CONVOLUTIONAL NETWORKS

- Introduced by Long CVPR 2015 for semantic segmentation
- Replace dense classification with convolutional layers
  - Same number of operations but with different output tensor shape
  - Allows processing input of any size (unlike dense layer with fixed input size)
- For larger image, equivalent to sliding CNN across image in blocks

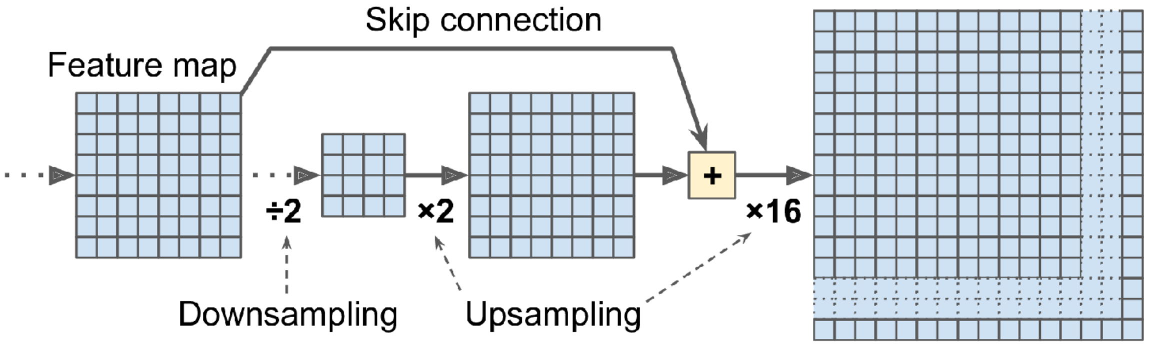


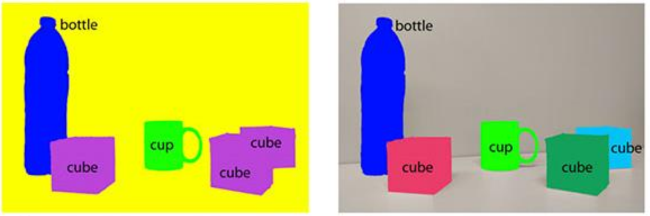
OBJECT DETECTION ARCHITECTURES

- Fast(er) R-CNN
  - Apply FCN approach with region proposals
  - Fast R-CNN uses Selective Search
  - Faster R-CNN uses a small region proposal network to predict bounding boxes
- YOLO (you only look once) – major shift in approach with a single CNN pass
  - Divide image into cells and predict 5 bounding boxes per cell
  - Predicts bbox offset rather than absolute location (smaller range)
  - Use of anchor boxes (bounding box priors) as prototypical object dimensions
  - Trained with images of different scale → detect different scale
- SSD (single shot detector)
  - Better accuracy than YOLO
  - Use of MultiBox with decreasing convolutional layers for detection scales
  - More bounding box predictions than YOLO

SEMANTIC SEGMENTATION

- Each pixel is classified according to the class of the object it belongs
  - Different objects of same class are not distinguished (panoptic segmentation)
- Traditional CNNs lose spatial resolution due to layer stride
  - Need to "upsample" coarse feature map
  - Use transposed convolutional layer
  - Add skip connections for better resolution
- Instance segmentation – each object is distinguished from each other
  - Mask R-CNN, Kaiming He 2017 as extension of Faster R-CNN to produce pixel mask for each bounding box





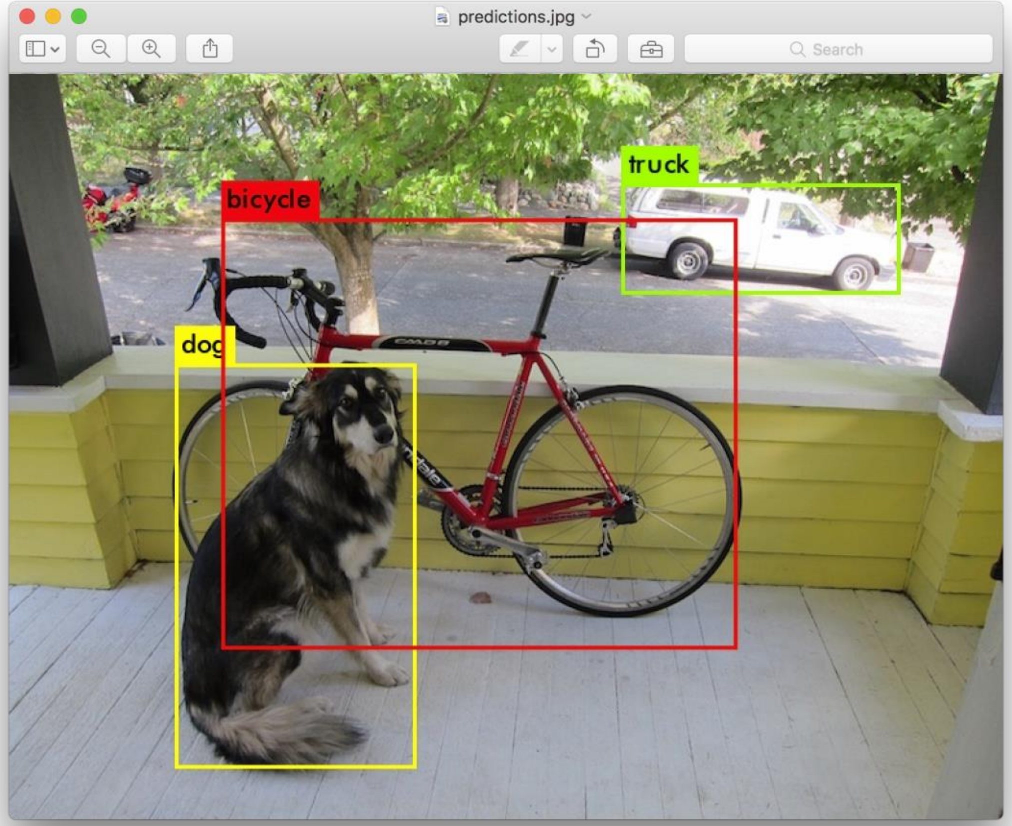
(c) Semantic segmentation (d) Instance segmentation

OBJECT DETECTION

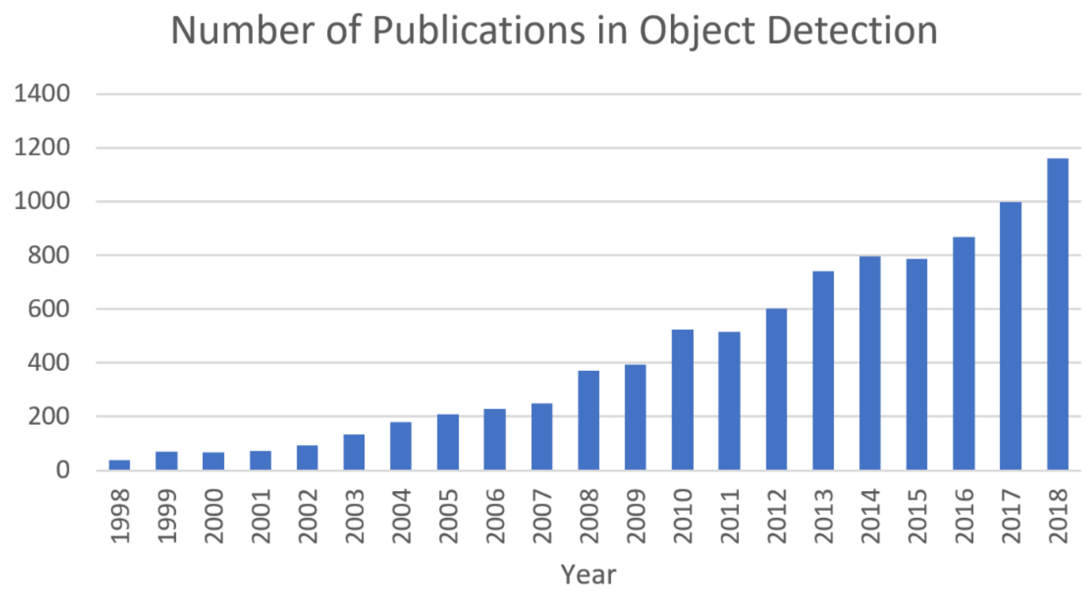
OBJECT DETECTION WITH DEEP LEARNING: A REVIEW
ZHAO, ZHENG, XU, AND WU, T-NNLS 2019

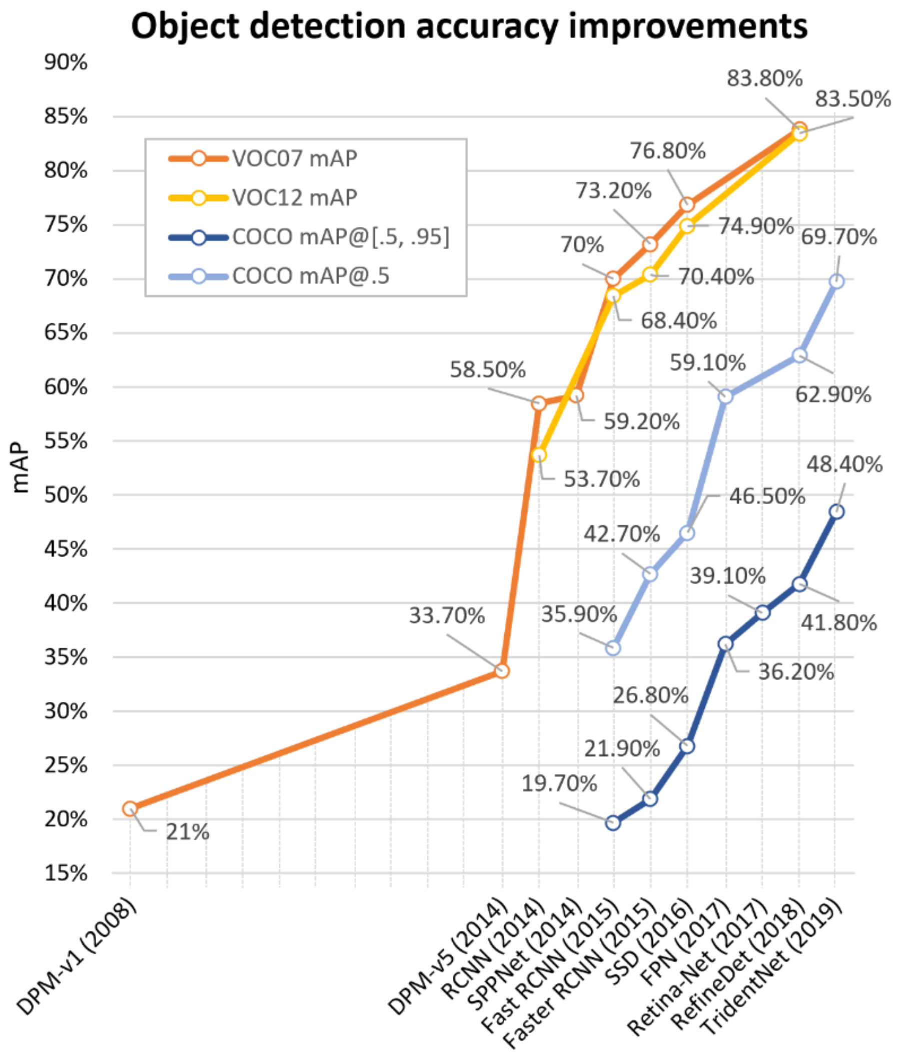
OBJECT DETECTION OVERVIEW

- Fundamental computer vision problem
- Categorize not just the whole image but delineate (with bounding boxes) where various objects are located (object localization)
  - Localization is viewed as a bounding box regression task
- Provides a semantic understanding of images (video)
- Related tasks: image classification, human behavior analysis, face recognition, autonomous driving



DEEP CNN DOMINANCE IN DETECTION





Zou et al., "Object Detection in 20 Years: A survey, 2019

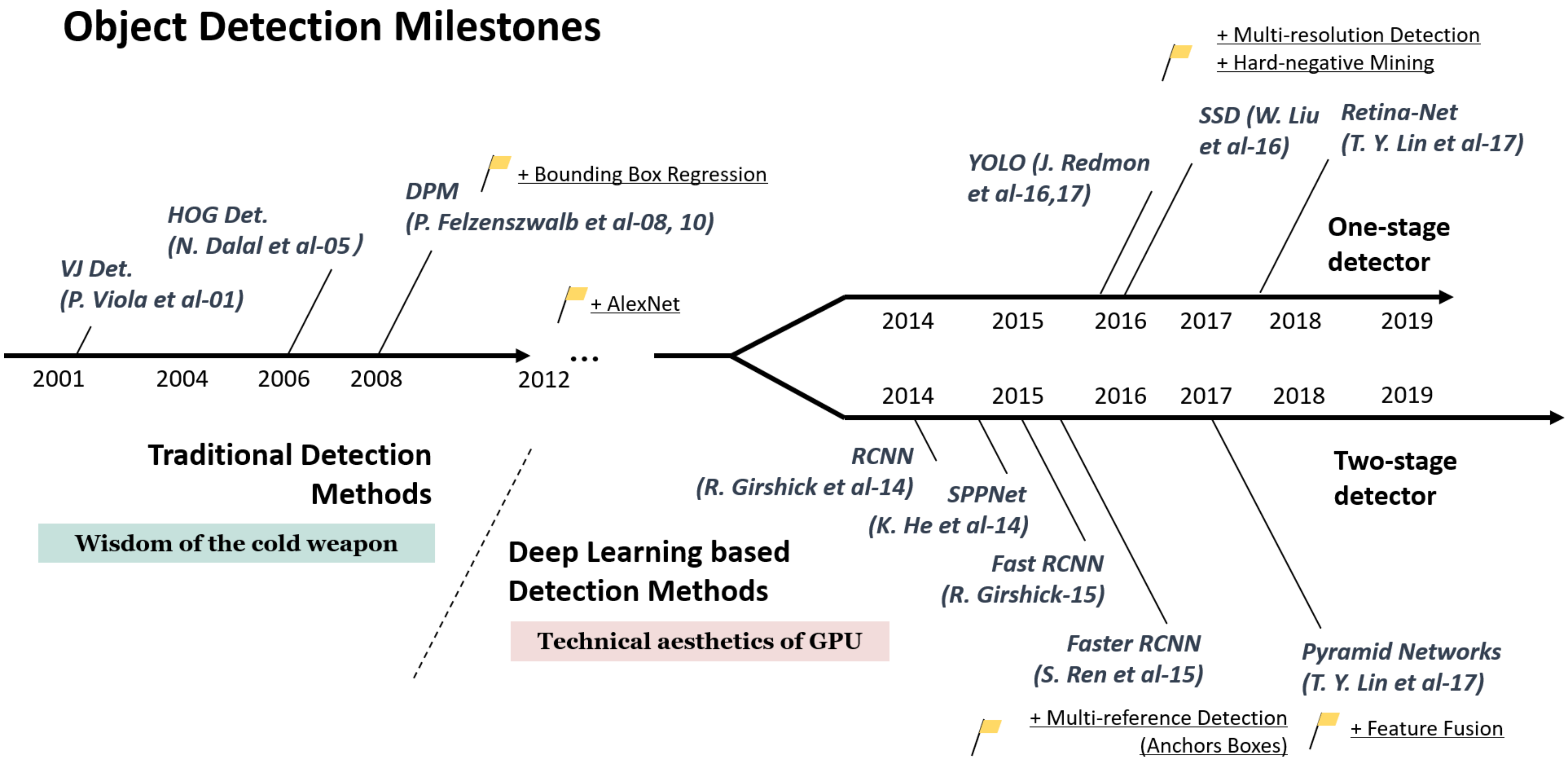
DEEP LEARNING AND CNNS

- Deep learning dominance:
  - Large scale annotated training datasets
  - Fast development of high performance parallel computing (GPUs)
  - Advances in network structures
    - Initialization: pre-training
    - Overfitting: Dropout and data augmentation
    - Efficiency: batch normalization
    - Architectures: AlexNet, Inception, ResNet
- CNN advantages:
  - Hierarchical feature representation
  - Deeper architecture for increased expressive capability
  - Can jointly optimize several related tasks (multi-task learning)
  - Classical CV can be recast as high-D data transform problems

GENERIC OBJECT DETECTION

- Locate and classify all objects (of interest) in an image
  - Label each object with a rectangular bounding box
  - Have a measure of confidence in detection
- Two major approaches:
  - Two-stage: i) generate region proposals and ii) classify each proposal into different object categories
  - One-stage: detection as a regression or classification to get both categories and locations directly at once

OBJECT DETECTION MILESTONES



Zou et al., "Object Detection in 20 Years: A survey, 2019

TRADITIONAL DETECTOR REVIEW

- Viola Jones cascade detector
  - Viola and Jones, 1999
- Histogram of Oriented Gradients (HOG) detector
  - Dalal and Triggs, 2005
- Deformable Part-based Model (DPM)
  - Felzenszwalb, 2008

VIOLA JONES

- Real-time face detection with sliding window for position and scale
- Integral image: speeded up Haar-like feature computation (speeded up filtering)
- Feature selection: Adaboost to automatically select a small but useful set of features (application driven filters)
- Detection cascades: multi-stage detector to avoid heavy computation on background windows but on faces

HOG

- Designed for pedestrian detection
- Improvement over SIFT and shape contexts
  - Balances feature invariance (translation, scale, illumination) and nonlinearity (different object categories)
- Descriptor computed on dense grid of uniformly spaced cells
- Used overlapping local contrast normalization over blocks
- Resizes input image while keeping detection window fixed for scale

DPM

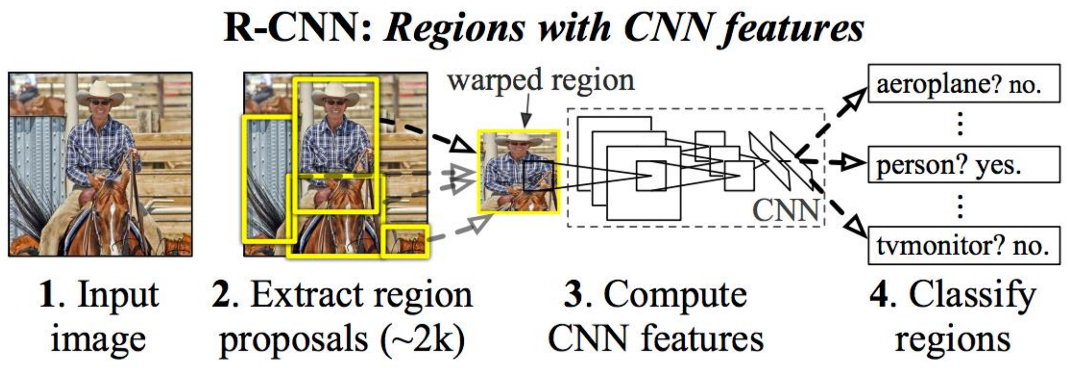
- Extension of HOG and was winner of VOC 07-09
- Divide and conquer detection – object built from smaller parts to detect (bike has wheels, body, etc.)
  - Use of a star-model for connections – a root filter and part-filters
- Important contributions:
  - Extended with mixture models for more real-world variation (e.g. bike from front or side)
  - Hard negative mining – create negative examples on the margin
  - Bounding box regression

TWO-STAGE DETECTOR MILESTONES

- Region proposal based frameworks
  - "Coarse-to-fine" process somehow similar to human brain – scan full scene and then focus on region of interest
- Approaches
  - Overfeat – sliding window
  - Region CNN (R-CNN)
  - Spatial Pyramid Pooling Networks (SPPNet)
  - Fast R-CNN
  - Faster R-CNN
  - Feature pyramid network (FPN)

R-CNN (GIRSHICK 2013)

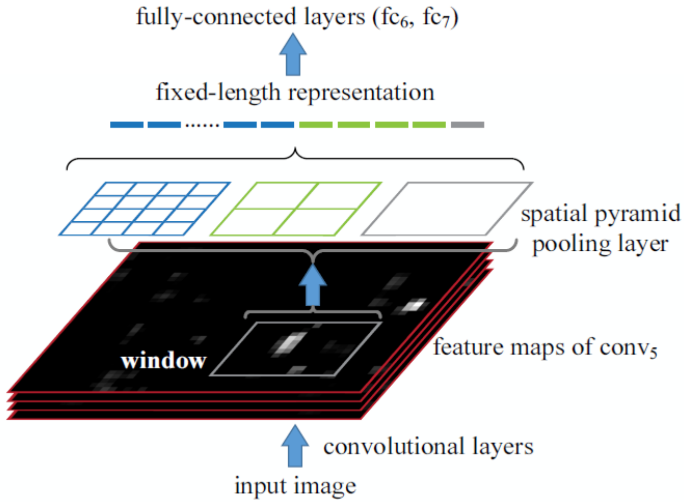
- Use selective search (Uijlings 2011) to generate a small set of potential object regions
  - Bottom-up grouping and saliency for proposals of various size
- Rescale proposals to fixed size and evaluate ImageNet pretrained CNN for feature extraction
- Multi-class linear SVM for classification



- Advantages: significant performance boost on VOC07
- Shortcomings: Redundant feature computations on overlapping regions make this slow

SPPNET (HE 2014)

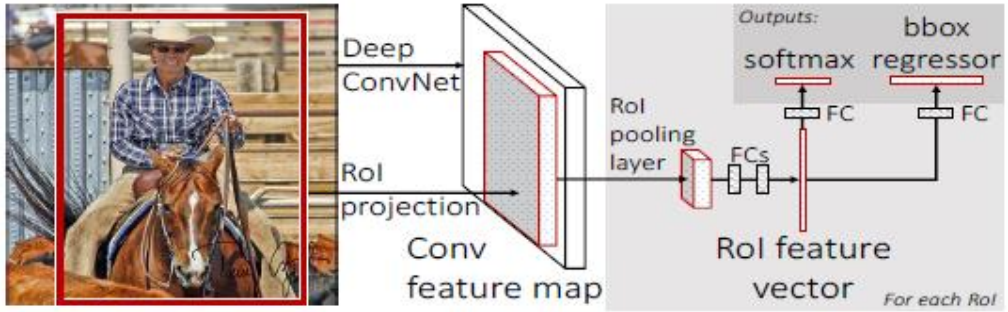
- Spatial pyramid pooling (SPP) layer enables a CNN to generate a fixed-length representation regardless of image size/ROI without rescaling
- Feature maps computed once for entire image and fixed-length representation can be made of arbitrary region
  - Use conv5 layer for SPP layer



- Advantage: 20x faster than R-CNN without accuracy loss
- Shortcomings: Training is still multi-stage and only FC layers are trained

FAST R-CNN (GIRSHICK 2015)

- Simultaneously train detector and bounding box regressor
  - No need for linear SVM layers
- Like SPPNet, image is only processed with convolutions once
  - RoI pooling layer to generate fixed-length feature vector
- FC layers branch to outputs:
  - Softmax class probabilities
  - Refined bounding box positions
- Optimized jointly with multitask loss (classification + localization)



- Advantages: Increased VOC mAP from by 11.5% from R-CNN
- Shortcomings: speed still limited by region proposals

FASTER R-CNN (REN 2015)

- Generate object proposals with a CNN model
  - First end-to-end and near real-time deep learning detector
- Introduced region proposal network (RPN)
  - Nearly cost-free region proposals as opposed to selective search
  - Produces object boundaries and scores for all positions simultaneously
  - Sliding window across conv layer
- Use of reference boxes (anchors) that match popular object dimensions
  - Later regressed for final bbox

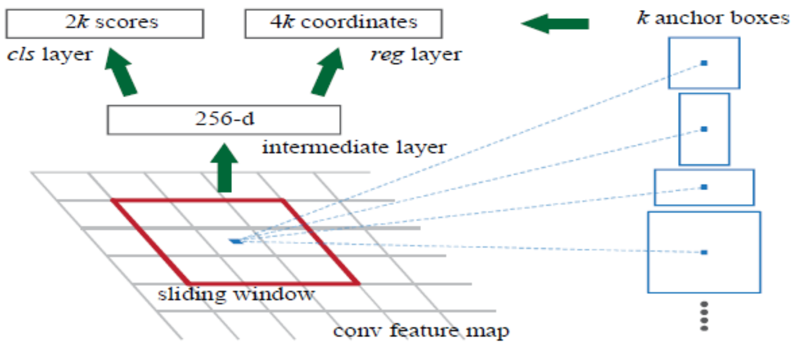


Fig. 6. The RPN in Faster R-CNN [18]. K predefined anchor boxes are convoluted with each sliding window to produce fixed-length vectors which are taken by cls and reg layer to obtain corresponding outputs.

- Advantages: trained end-to-end (all layers) and high 5 fps on GPU with SOTA VOC results
- Shortcomings: long training time, poor performance on extreme scales/shapes, object regions rather than instances

FPN (LIN 2017)

- Handle wide scale variation through use of image pyramid
  - Deeper CNN layers useful for category recognition but poor for localization
- Top-down architecture with lateral connections to share high level features with higher resolution of lower layers
  - Avoid expensive explicit image pyramid computation
- General approach for efficient multi-scale representation
  - Extensively used in semantic segmentation

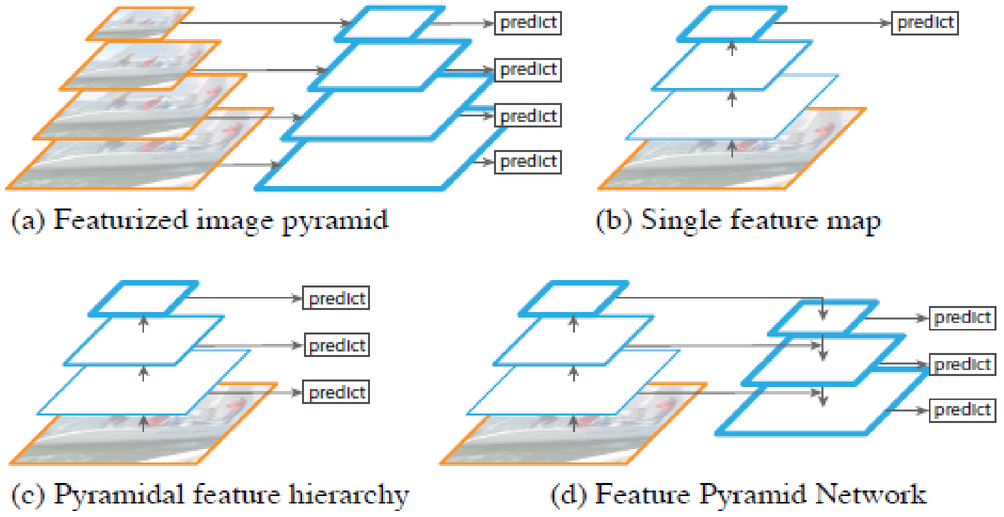


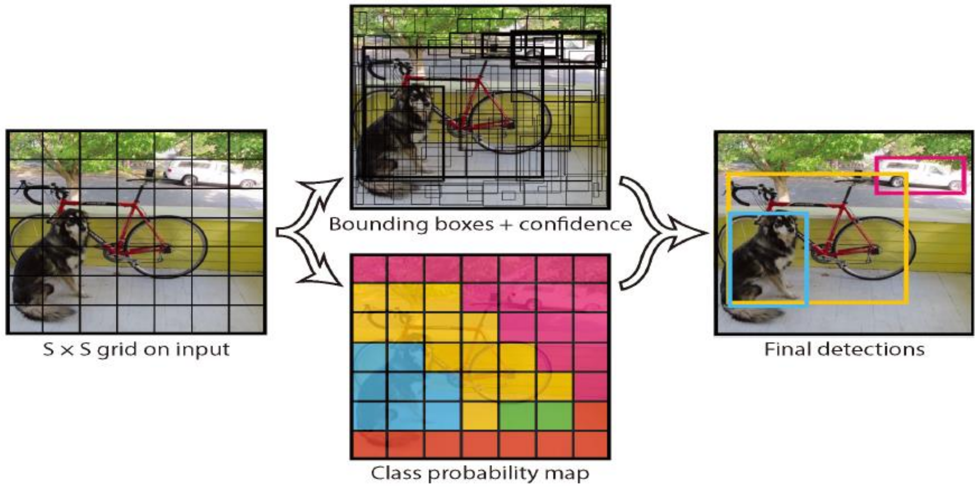
Fig. 7. The main concern of FPN [66]. (a) It is slow to use an image pyramid to build a feature pyramid. (b) Only single scale features is adopted for faster detection. (c) An alternative to the featurized image pyramid is to reuse the pyramidal feature hierarchy computed by a ConvNet. (d) FPN integrates both (b) and (c). Blue outlines indicate feature maps and thicker outlines denote semantically stronger features.

ONE-STAGE DETECTOR MILESTONES

- End-to-end regression/classification methods
  - Single step to produce detections
- Approaches
  - MultiBox
  - AttentionNet
  - Grid-based object detector (G-CNN)
  - You Only Look Once (YOLO)
  - Single Shot Multi-box Detector (SSD)

YOLO (REDMOND 2015)

- First one-stage detector
  - Extremely fast by abandoning proposal detection + verification approach
- Divides an image into regions and predicts bounding boxes and probabilities for all regions simultaneously
  - Each grid region predicts objects centered within that grid cell
  - B bounding boxes are predicted with associated confidence score



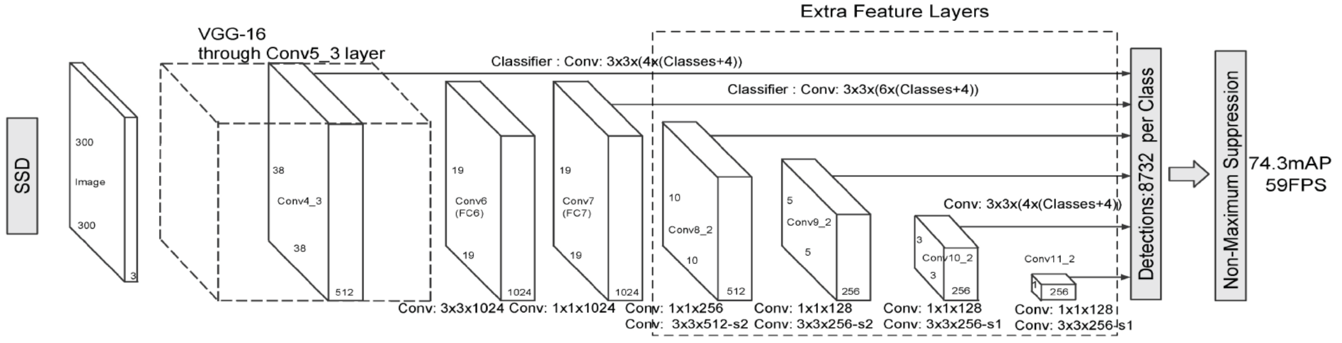
- Advantages:
  - Extremely fast (45-155 fps VOC)
- Shortcomings:
  - Poorer localization than two-stage detectors
  - Difficulty with small scale objects

YOLO II

- Customized CNN architecture from scratch
  - Inception-like modules
- Divide image into $S \times S$ grid
- Each grid cell predicts an object centered with the cell
  - Local search with relative coordinates (scale for image size)
  - B bounding boxes predicted for each cell with confidence
  - Conditional class probabilities predicted for each of the C
- Training loss
  - Bounding box localization
    - Box center relative to grid
    - Normalized height/width relative to image size
  - Confidence score
  - Classification error
    - Only when object is in cell
- Upgrades (v2, v3, etc.)
  - Batch normalization
  - Anchor boxes
  - Dimension cluster
  - Multi-scale training

SSD (LIU 2015)

- Multi-reference and multi-resolution detection technique
  - Detects at different scales at different layers of network
  - Better handles small objects
- Inspired by anchors of MultiBox, RPN, and multi-scale representation
- Add feature layers at the end of standard backbone (VGG16)
  - Predict offsets to default bounding boxes of different scales and aspect ratios and confidences
  - Final detection after NMS on multi-scale refined boxes



- Advantages:
  - Fast (59 fps) while more accurate than YOLO
- Shortcomings:
  - Still issues with small objects (better backbone e.g. ResNet101)

SSD II

- MultiBox (Szegedy 2014)
  - Inception-like structure to reduce dimensionality but not spatial resolution (height x width)
  - Confidence loss to measure objectiveness of bounding box (categorical cross-entry)
  - Location loss to measure how far a predicted bounding box (L2 but SSD uses smooth L1)
- Used anchors to get good prediction starting point for regression
  - 11 priors/feature map = 1420 anchors/image for images at multiple scales and sizes
  - SSD extended idea to each cell in feature map to avoid explicit anchor pre-train (6/cell)
- Hard negative mining - 3:1 ratio of neg:pos train examples
  - Need to keep low IoU predictions
- Data augmentation – random flipping and patches of original image at different IoU ratios
- Non-maximum suppression (NMS) – discard low confidence and IoU
- 80% of time is spent on base VGG16
  - Can improve speed/performance with better backbone

TECHNIQUES FOR BASE IMPROVEMENT

- Multi-task learning – learn better representation from multiple correlated tasks
  - Train conv layers for e.g. region proposal, classification, and segmentation
- Multi-scale representation – combine activations from multiple layers with skip-layer connections
  - Provide semantic information of different spatial resolutions
- Contextual modeling – exploit features from surround
  - Provide features from different support regions/resolutions which help with occlusion and local similarities (e.g. tennis ball versus lemon when a racket is nearby)

REFERENCES

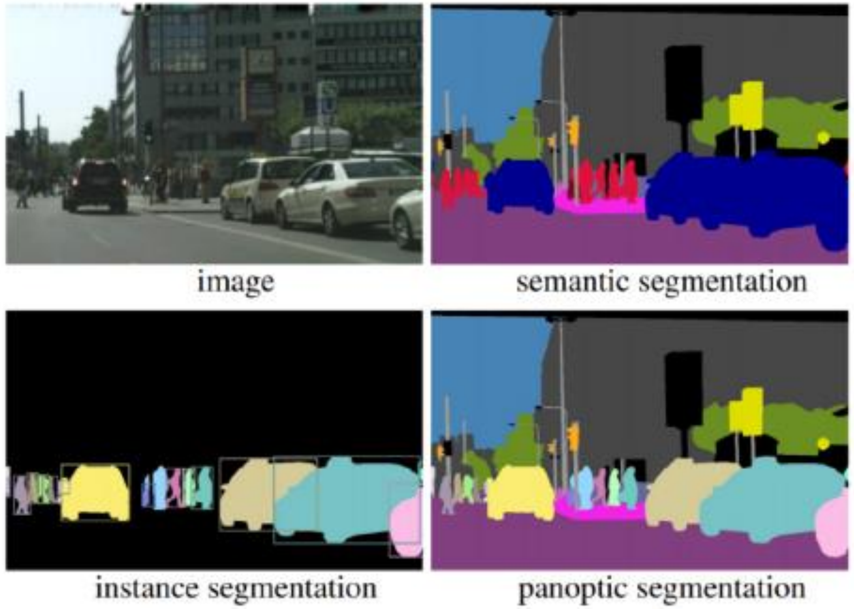
- For more complete overview, see recent surveys
- Object Detection with Deep Learning: A Review
- Object Detection in 20 Years: A Survey

IMAGE SEGMENTATION

EVOLUTION OF IMAGE SEGMENTATION USING DEEP CONVOLUTIONAL NEURAL NETWORKS: A SURVEY, SULTANA, SUFIAN, AND DUTTA, KBS 2020

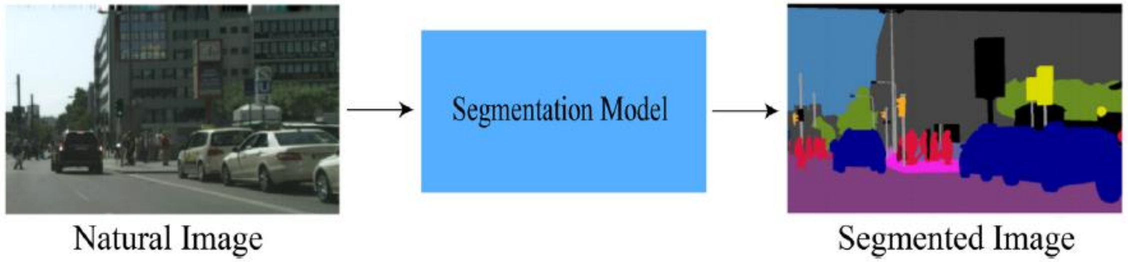
SEGMENTATION TASKS

- Segmentation – CV task of segregating an image into multiple regions according to different properties of pixels (e.g. color, intensity, texture)
  - Typically a low-level task that relies on spatial information (neighborhood)
- Semantic segmentation – associate a class label for every pixel in an image
- Instance segmentation – mask (segment) each instance of an object in an image independently
- Panoptic segmentation – combination of semantic segmentation and instance segmentation
  - Label both class and separate instances (detection)



SEMANTIC SEGMENTATION

- Pixel level class labels
- Have relied heavily on CNNs since 2012
- Popular approaches:
  - Fully convolutional network
  - Dilated/atrous convolution
  - Top-down/bottom-up approach
  - Global context
  - Receptive field enlargement and multi-scale context



FCN [LONG 2017]

- Fully convolutional network (FCN) was proposed for semantic segmentation
- Use standard CNN backbone but remove dense FC layers
  - Use of 1x1 convolution instead
  - Produces a class presence heatmap in low-resolution
- Bilinear interpolation used to upsample coarse output to pixel resolution
- Skip connections (deep jet) to combine final prediction layer with higher res/feature-rich lower layers

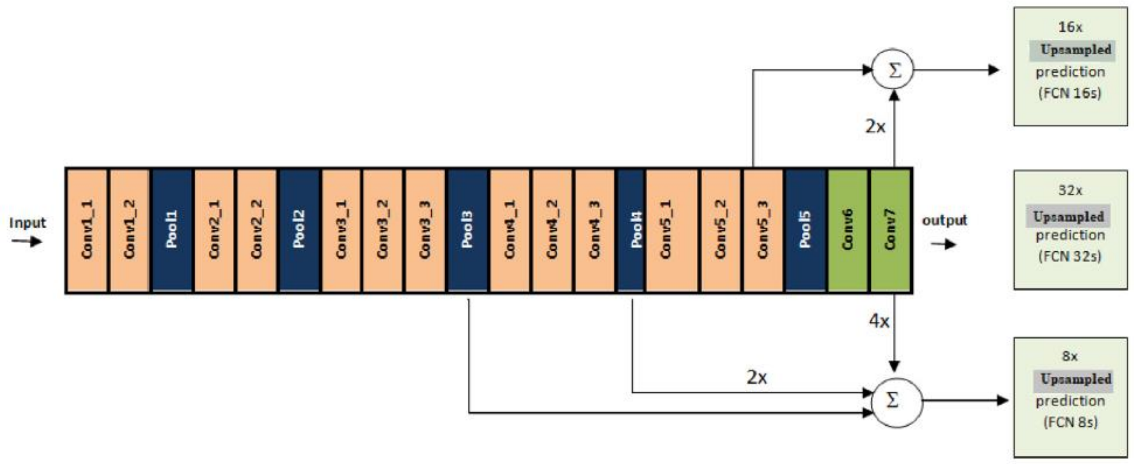
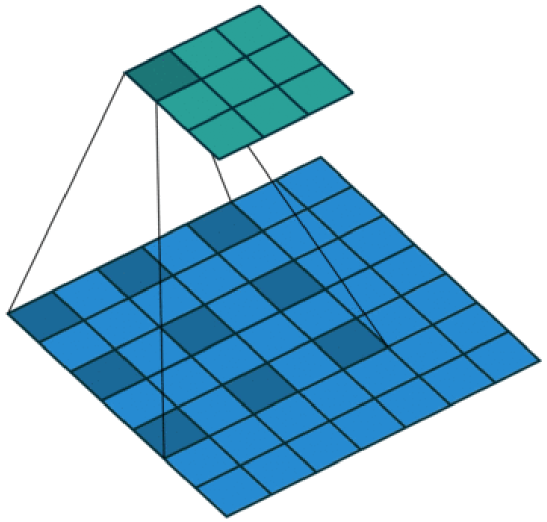


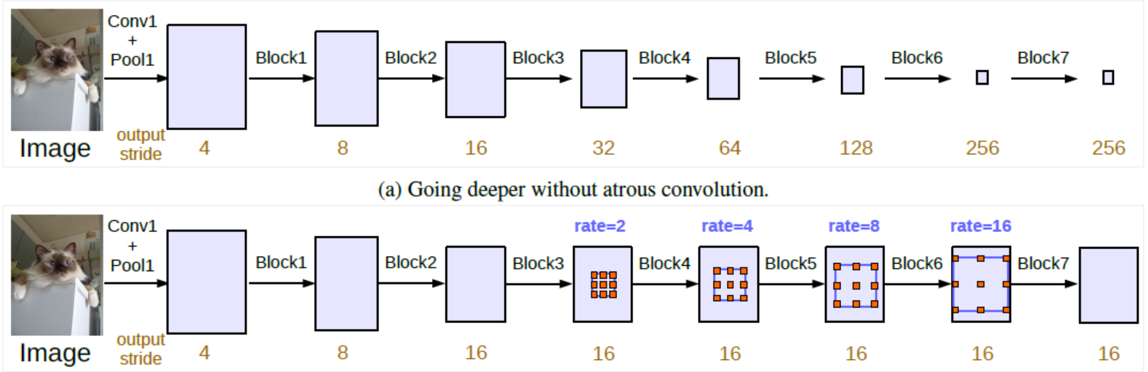
Fig. 4. Architecture of FCN32s, FCN16s, FCN8s.

DILATED/ATROUS CONVOLUTION

- Context is important for segmentation but Traditional convolution is expensive for larger field-of-view (kernel size)
- Atrous convolution introduces a dilation rate
  - Trade-off context vs localization
- Traditional CNN loses resolution while atrous can keep it
  - Larger feature map is better for segmentation (less interpolation)
  - However, isolates pixel from context
- Key architectures: DilatedNet and DeepLab (CRF for fine details)

source





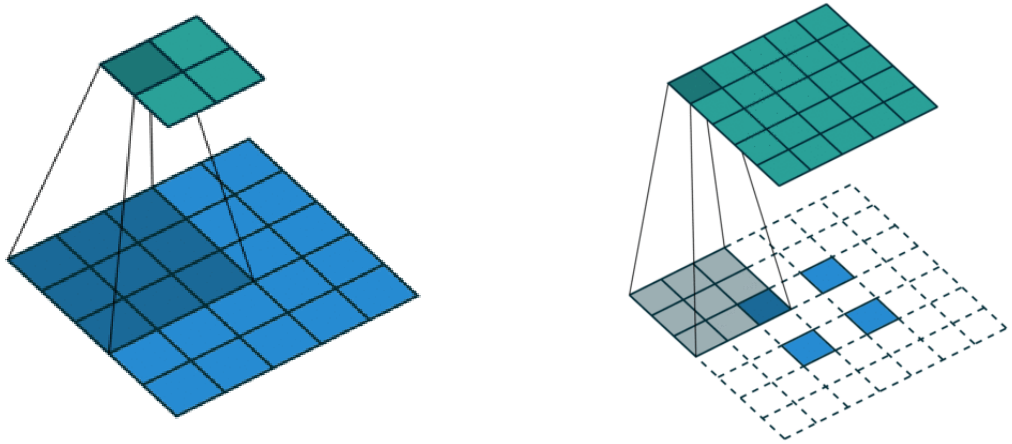
(a) Going deeper without atrous convolution.

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

TOP-DOWN/BOTTOM-UP APPROACH

- Encoder-decoder architecture
  - Convolution encodes image features
  - Deconvolutional network to decode features into pixels/labels
- Deconvolution (transposed convolution) reconstructs spatial resolution
  - Upscaling convolution operation
- Both encoder and decoder extract features
- Generally lose fine-grained information in encoding process
  - Skip connections utilized to pass higher-resolution features
- Key architectures: Deconvnet, U-Net, SegNet, FC-DenseNet, HRNet

- conv de-conv source



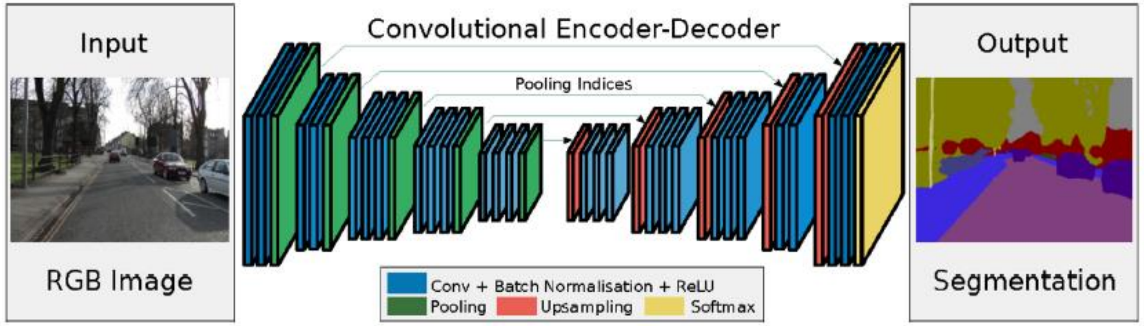
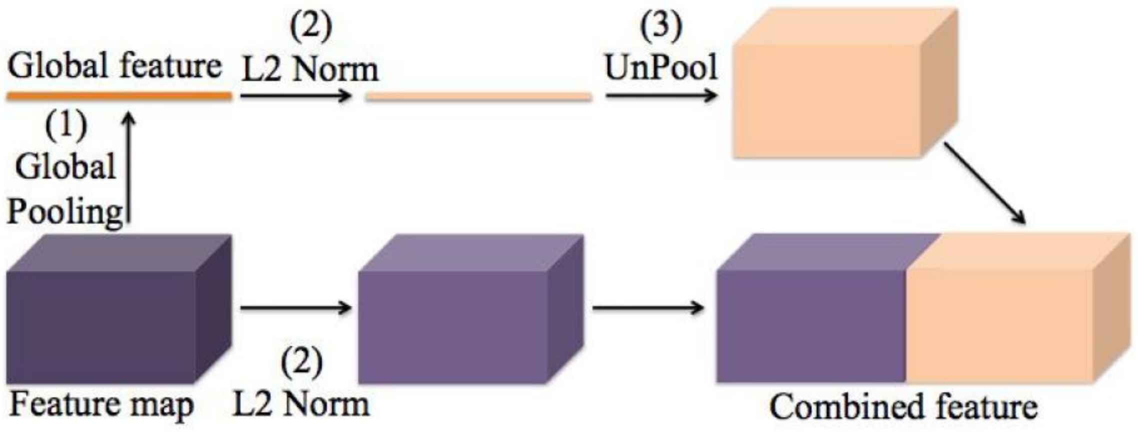


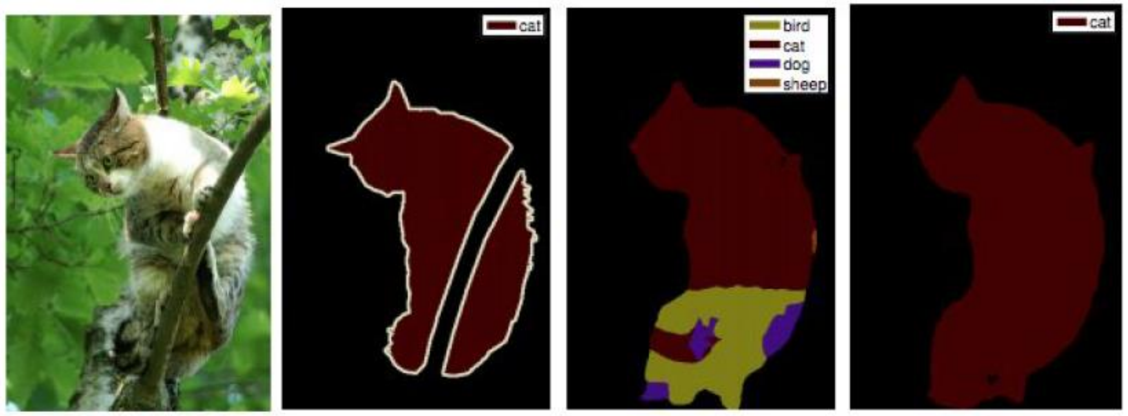
Fig. 9. Encoder–decoder architecture of SegNet.
Source: From [93].

GLOBAL CONTEXT

- Most segmentation relies on just local information but global context is important
  - Add global features or global context information
- Global features
  - Global average pool (final layers)
  - Large convolution kernels
- Context
  - Use of class mapping
- Helps resolve inaccuracies but lacks scaling information of multiscale objects
- Key architectures: ParseNet, GCN, EncNet



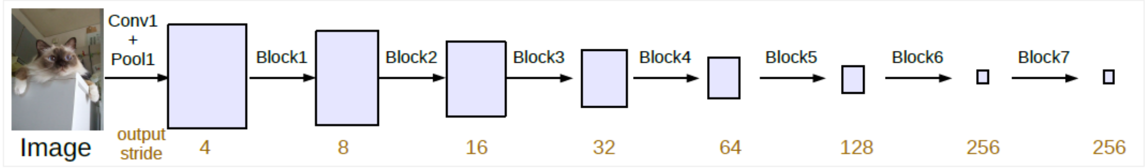
(e) ParseNet contexture module overview.



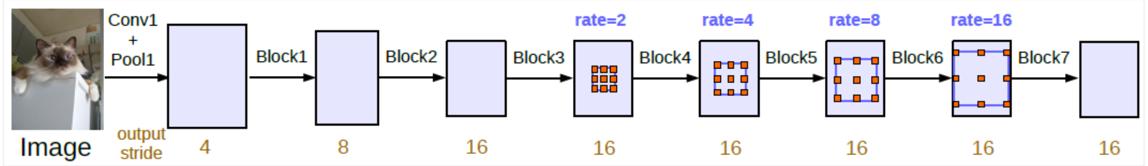
(a) Image (b) Truth (c) FCN (d) ParseNet

RECEPTIVE FIELD ENLARGEMENT AND MULTI-SCALE CONTEXT

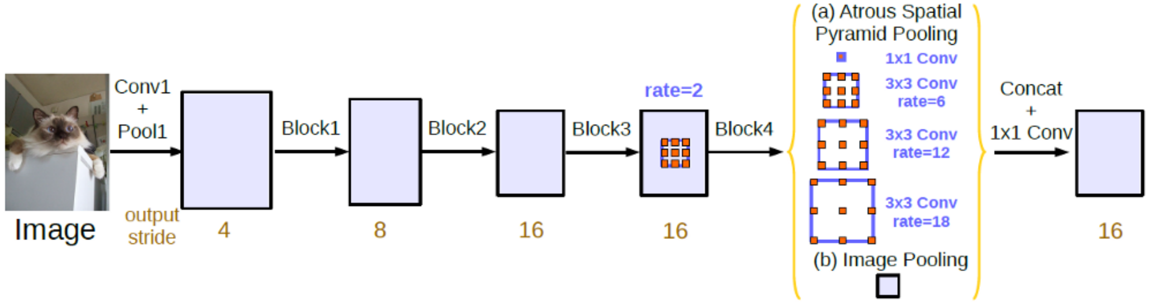
- Use of feature pyramid techniques for multi-resolution representation
  - Atrous Special Pooling Pyramid (ASPP)
  - Pyramid pooling module
- Provides better localization
- Helps incorporate scale information of objects for fine-grained segmentation
- Key architectures: DeepLabv2, DeepLabv3, PSPNet, Gated-SCNN



(a) Going deeper without atrous convolution.



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



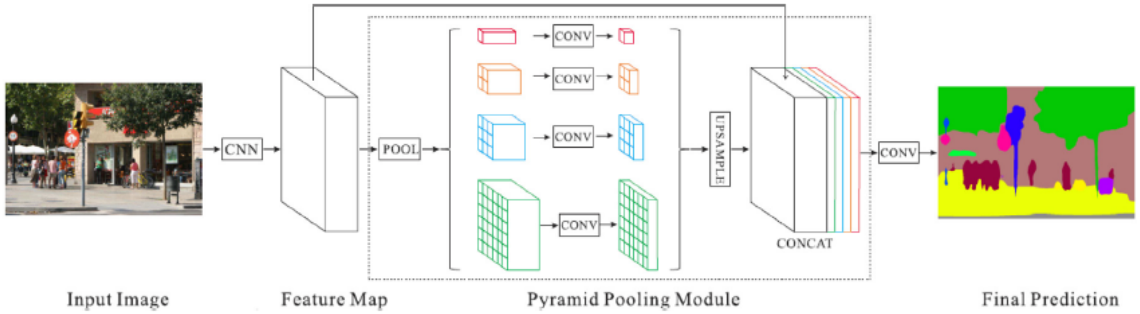


Fig. 15. PSPNet Model Design.

INSTANCE SEGMENTATION

- Each instance of a particular object is masked independently
- Task is intertwined with object detection
  - Detection gives bounding box while instance segmentation further refines with mask
- General approach is to give proposals of objects/masks and refine
- Mask R-CNN as example
  - Faster R-CNN extension
  - RPN for object proposals – classification and bounding box regression
  - Separate segmentation network for each ROI

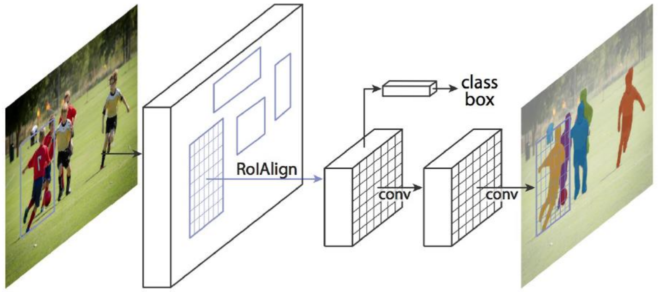


Fig. 17. Mask R-CNN architecture for instance segmentation. From [64].

Fig. 18. Mask R-CNN results on sample images from the COCO test set. From [64].

PANOPTIC SEGMENTATION

- Combination of instance segmentation and semantic segmentation
  - Newer segmentation task
- General approach:
  - Heads for semantic segmentation
  - Head for instance segmentation
  - Panoptic head to combine
- Key architectures: OANet, UPSNet, Multitask Network

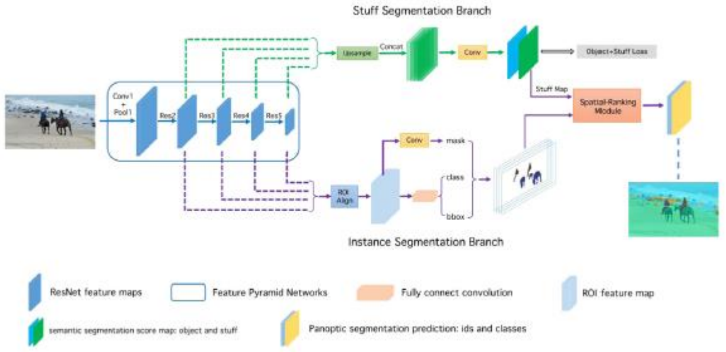


Fig. 27. Architecture of Occlusion Aware Network (OANet).
Source: From [183].

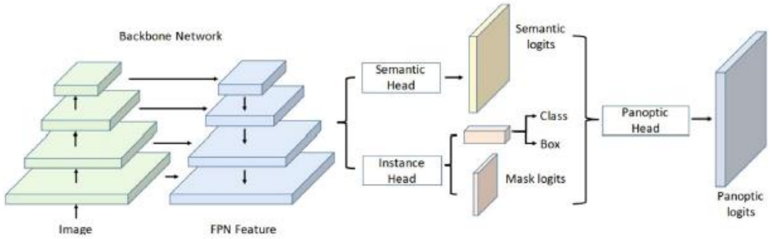


Fig. 28. Architecture of unified panoptic segmentation network (UPSNet).
Source: From [185].

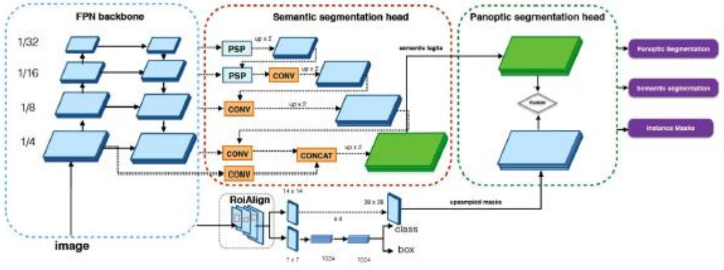


Fig. 29. Architecture of Multitask Network for Panoptic Segmentation.
Source: From [186].

REFERENCES

- For more complete overview, see recent surveys
- Evolution of Image Segmentation using Deep Convolutional Neural Network: A Survey
- Image Segmentation Using Deep Learning: A Survey