

Deep Learning - COSC2779

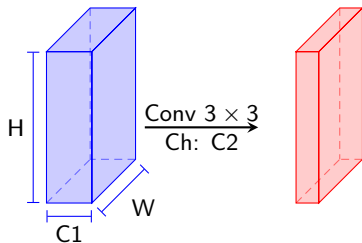
Vision Application & CNN Architectures

Dr. Ruwan Tennakoon



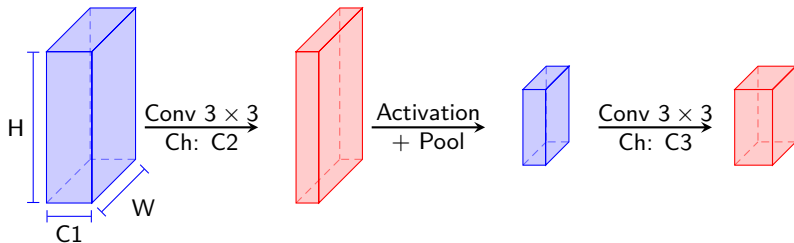
Aug 16, 2021

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation



Convolutions can be combined with pooling to construct a chain of layers.

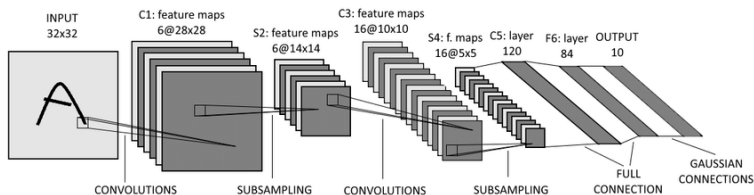
- Feature extraction usually happens locally - **sparse connectivity**.
- In feature extraction the same operation is applied at different locations - **parameter sharing**.
- Pooling help reduce redundant information and provide some level of *invariance to translations*.



Convolutions can be combined with pooling to construct a chain of layers.

- Feature extraction usually happens locally - **sparse connectivity**.
- In feature extraction the same operation is applied at different locations - **parameter sharing**.
- Pooling help reduce redundant information and provide some level of *invariance to translations*.

"LeNet is a classic example of convolutional neural network to successfully predict handwritten digits." [LeNet]



There are so many hyper parameter to choose in CNN:

- Number of convolutional layers, filter size, number of filters, stride, initialization ...
- Pooling size, Number of pooling layers ...
- Number of FC layers, units, ...
- optimization type, learning rate, ...
- ...

There are so many hyper parameter to choose in CNN:

- Number of convolutional layers, filter size, number of filters, stride, initialization ...
- Pooling size, Number of pooling layers ...
- Number of FC layers, units, ...
- optimization type, learning rate, ...
- ...

Use classic networks like LeNet, AlexNet, VGG-16, VGG-19, ResNet etc. as inspiration (follow the trend used in those architectures).

- Identify how deep networks are developed via case study: Image classification (IMAGENET).
- Understand the main trends in classic architectures and why they work.
- Identify the classic network architectures used for common computer vision problems:
 - Image Classification
 - Object Detection
 - Image Segmentation

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is an annual competition held between 2010 and 2017.

The datasets comprised approximately 1 million images and 1,000 object classes.

The annual challenge focuses on multiple tasks for image classification.

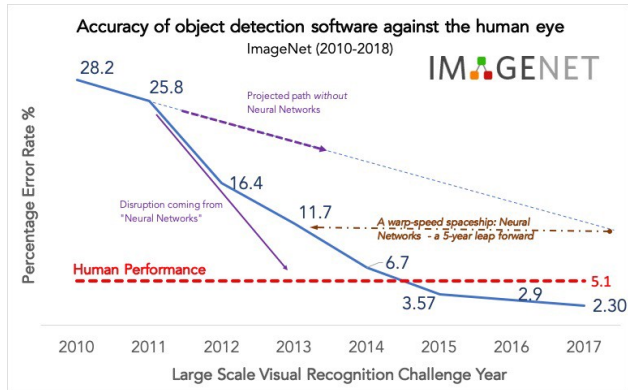
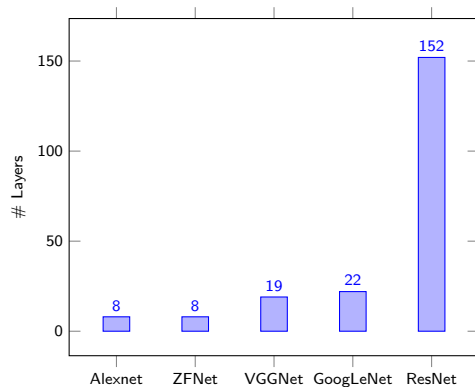
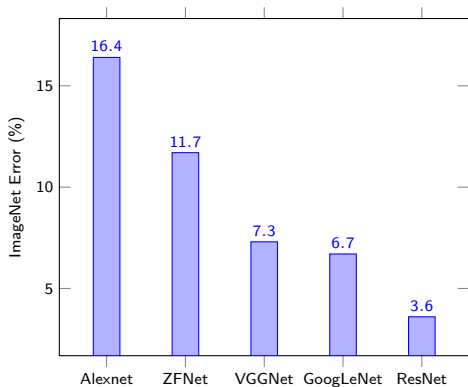


Image source: ImageNet

Alex Krizhevsky, et al. "ImageNet Classification with Deep Convolutional Neural Networks" developed a convolutional neural network that achieved top results on the ILSVRC-2010 and ILSVRC-2012 image classification tasks.

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation



Supervised learning based image classification.

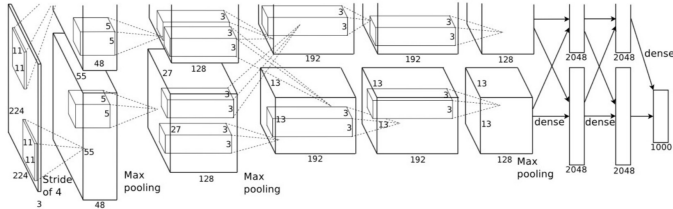


Image: ImageNet Classification with Deep Convolutional Neural Networks

Input: $227 \times 227 \times 3$

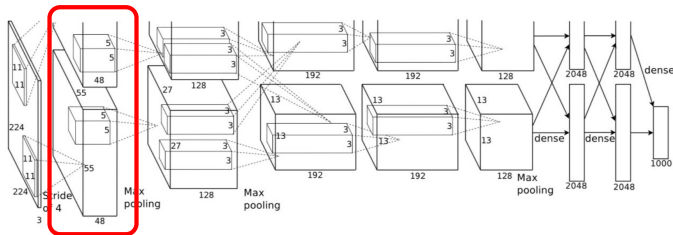


Image: ImageNet Classification with Deep Convolutional Neural Networks

Input: $227 \times 227 \times 3$

Layer 1: 2D Convolution with 96, $[11 \times 11]$ filters, with stride of 4. 'ReLU' activation.
Output Shape ?

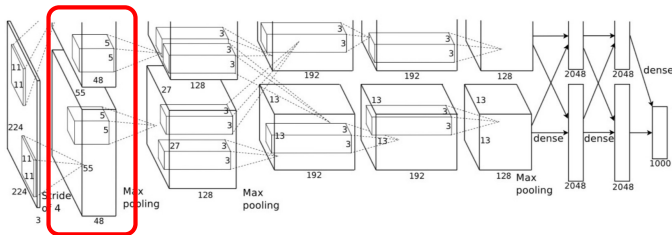


Image: ImageNet Classification with Deep Convolutional Neural Networks

Input: $227 \times 227 \times 3$

Layer 1: 2D Convolution with 96, $[11 \times 11]$ filters, with stride of 4. 'ReLU' activation.
Output Shape $(W - F + 2P)/S + 1 = (227 - 11 + 2 * 0)/4 + 1 = 55 \rightarrow [?, 55, 55, 96]$
Parameters ?

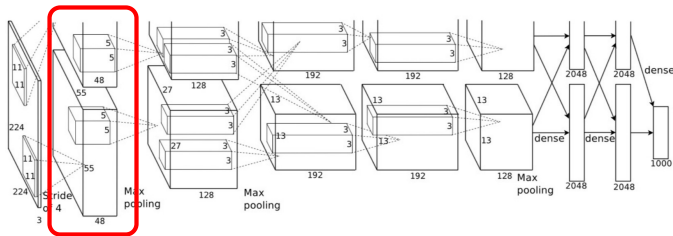


Image: ImageNet Classification with Deep Convolutional Neural Networks

Input: $227 \times 227 \times 3$

Layer 1: 2D Convolution with 96, $[11 \times 11]$ filters, with stride of 4. 'ReLU' activation.
Output Shape $(W - F + 2P)/S + 1 = (227 - 11 + 2 * 0)/4 + 1 = 55 \rightarrow [?, 55, 55, 96]$
Parameters $11 \times 11 \times 3 \times 96 + 96$

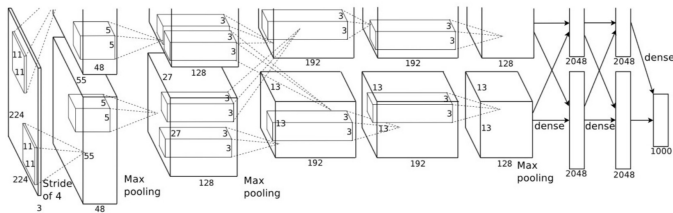


Image: [ImageNet Classification with Deep Convolutional Neural Networks](#)

Input: $227 \times 227 \times 3 \rightarrow$ **After Conv1:** $55 \times 55 \times 96$

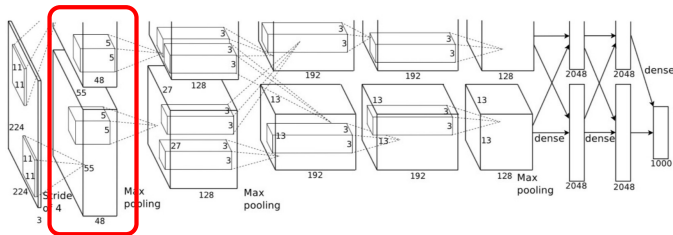


Image: ImageNet Classification with Deep Convolutional Neural Networks

Input: $227 \times 227 \times 3 \rightarrow$ **After Conv1:** $55 \times 55 \times 96$

Layer 2: Max Pooling with, $[3 \times 3]$, with stride of 2.
Output Shape ?

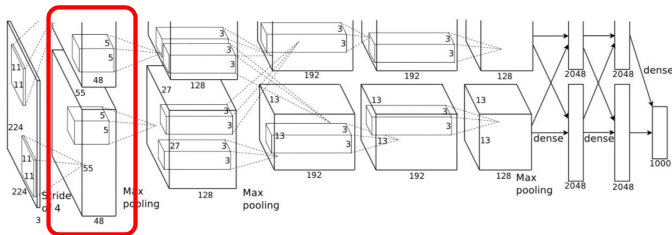


Image: ImageNet Classification with Deep Convolutional Neural Networks

Input: $227 \times 227 \times 3 \rightarrow$ **After Conv1:** $55 \times 55 \times 96$

Layer 2: Max Pooling with, $[3 \times 3]$, with stride of 2.

Output Shape $(W - F + 2P)/S + 1 = (55 - 3 + 2 * 0)/2 + 1 = 27 \rightarrow [?, 27, 27, 96]$

Parameters ?

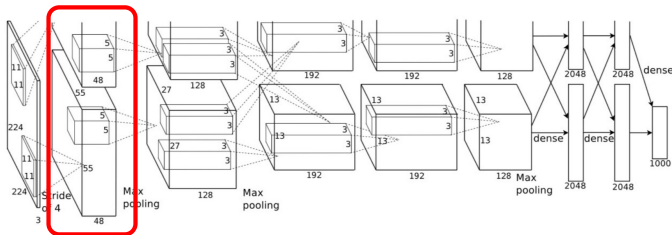


Image: [ImageNet Classification with Deep Convolutional Neural Networks](#)

Input: $227 \times 227 \times 3 \rightarrow$ **After Conv1:** $55 \times 55 \times 96$

Layer 2: Max Pooling with, $[3 \times 3]$, with stride of 2.

Output Shape $(W - F + 2P)/S + 1 = (55 - 3 + 2 * 0)/2 + 1 = 27 \rightarrow [?, 27, 27, 96]$

Parameters 0

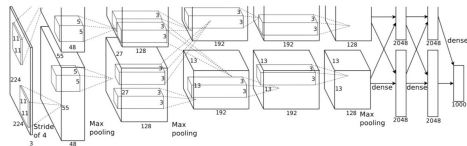


Image: [Krizhevsky et al. 2012](#)

227x227x3 Input

55x55x96 Conv1: 96 11x11 filters at stride 4, pad 0

27x27x96 Max-pool1: 3x3 filters at stride 2

27x27x96 Norm1: Normalization layer

27x27x256 Conv2: 256 5x5 filters at stride 1, pad 2

13x13x256 Max-pool2: 3x3 filters at stride 2

13x13x256 Norm2: Normalization layer

13x13x384 Conv3: 384 3x3 filters at stride 1, pad 1

13x13x384 Conv4: 384 3x3 filters at stride 1, pad 1

13x13x256 Conv5: 256 3x3 filters at stride 1, pad 1

6x6x256 Max-pool3: 3x3 filters at stride 2

4096 FC6: 4096 neurons

4096 FC7: 4096 neurons

1000 FC8: 1000 neurons (class scores)

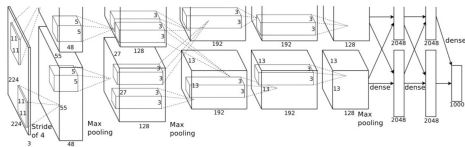
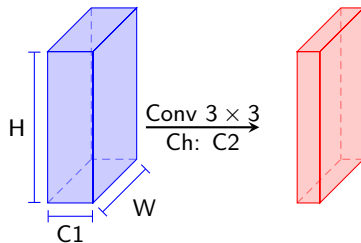


Image: [Krizhevsky et al. 2012](#)

What happens if we change the input tensor height & width to conv layer?



Can we change the input to AlexNet?
No, This will change the dimensions of the input tensor to FC layers.

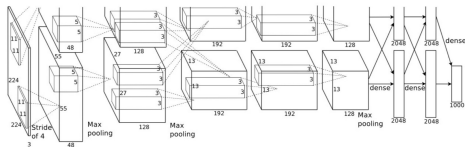


Image: [Krizhevsky et al. 2012](#)

- **Number of Parameters:** Overall, AlexNet has about 61M parameters.
- Use of ReLU
- Norm layers - not common anymore
- Data augmentation
- Dropout 0.5 in FC layers.
- Batch size 128
- SGD Momentum 0.9
- Initial learning rate $1e-2$, reduced by 10x manually when val accuracy plateaus.
- Regularization L2 weight decay $5e-4$
- 7 CNN ensemble: 18.2% to 15.4%

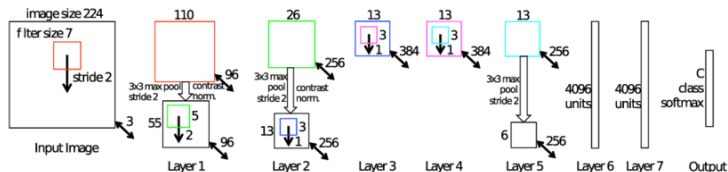
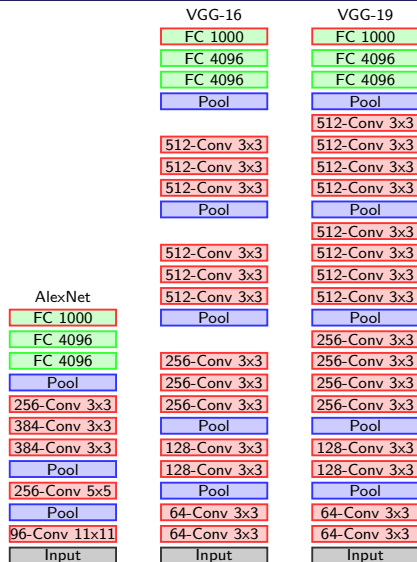


Image: [Visualizing and Understanding Convolutional Networks](#)

Similar to AlexNet:

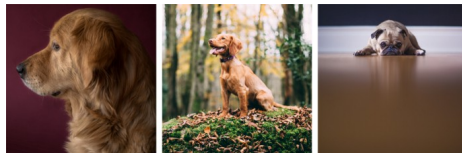
- Conv1: Change from 11x11 stride 4 to 7x7 stride 2.
- Conv3,4,5: Change from 384, 384, 256 filters to 512, 1024, 512.
- ImageNet top 5 error improved from 16.4% to 11.7%

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation



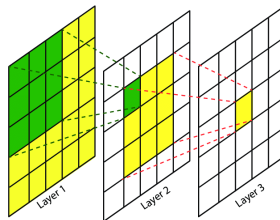
Convolution layers: 3x3, stride 1, pad 1.
Pooling layers: 2x2 max-pool stride 2.

	VGG-16	VGG-19
	FC 1000	FC 1000
	FC 4096	FC 4096
	FC 4096	FC 4096
	Pool	Pool
	512-Conv 3x3	512-Conv 3x3
	512-Conv 3x3	512-Conv 3x3
	512-Conv 3x3	512-Conv 3x3
	Pool	Pool
	512-Conv 3x3	512-Conv 3x3
	512-Conv 3x3	512-Conv 3x3
	512-Conv 3x3	512-Conv 3x3
	Pool	Pool
	512-Conv 3x3	512-Conv 3x3
	512-Conv 3x3	512-Conv 3x3
	512-Conv 3x3	512-Conv 3x3
	Pool	Pool
AlexNet		
FC 1000		
FC 4096		
FC 4096		
Pool		
256-Conv 3x3	256-Conv 3x3	256-Conv 3x3
384-Conv 3x3	256-Conv 3x3	256-Conv 3x3
384-Conv 3x3	256-Conv 3x3	256-Conv 3x3
Pool	Pool	Pool
256-Conv 5x5	128-Conv 3x3	128-Conv 3x3
Pool	128-Conv 3x3	128-Conv 3x3
96-Conv 11x11	Pool	Pool
Input	64-Conv 3x3	64-Conv 3x3
	64-Conv 3x3	64-Conv 3x3
	Input	Input



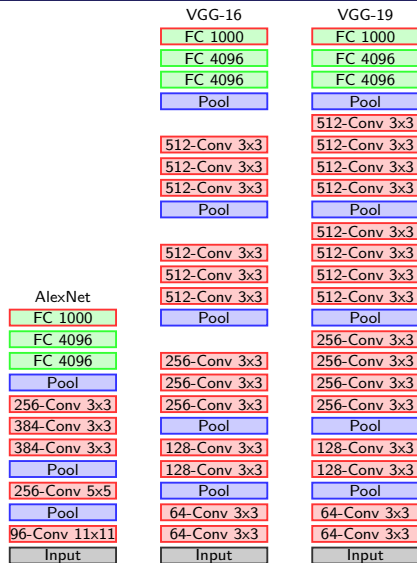
Images: from Unsplash

What will happen when objects are at different scales?



Stack of three 3x3 convolution (stride 1) layers has same effective receptive field

as one 7x7 convolution layer.



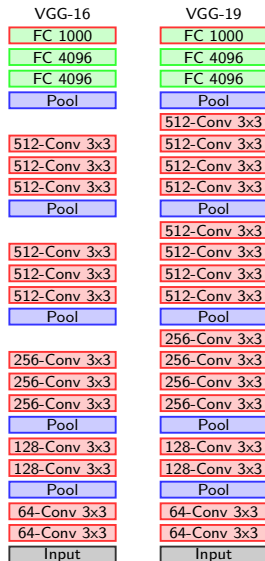
Convolution layers: 3x3, stride 1, pad 1.

Pooling layers: 2x2 max-pool stride 2.

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

Deeper structure allows more non-linearities.

Still have fewer parameters: $3 \times (3 \times 3 \times C_i \times C_o)$ Vs. $(7 \times 7 \times C_i \times C_o)$



Memory: 96MB per image (forward pass)

Number of Parameters: 138M parameters

- Similar training procedure as AlexNet.
- Use ensembles for best results.
- FC7 features generalize well to other tasks.
- Large amount of parameters at the last FC layers (80%).

Main idea: Smaller filters and deeper networks.

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation

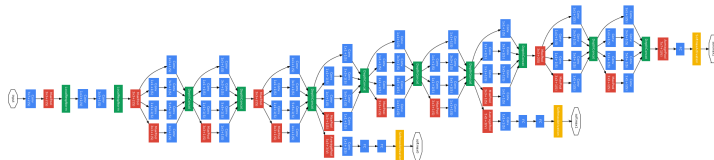


Image: [Going deeper with convolutions](#)

Computationally efficient deeper network:

- **Number of parameters:** 5M (12x less than AlexNet, 27x less than VGG-16)
- No fully connected layers at the end. Average pooling across channels.
-

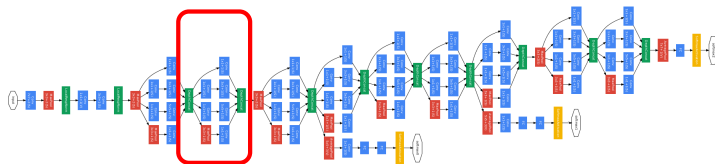
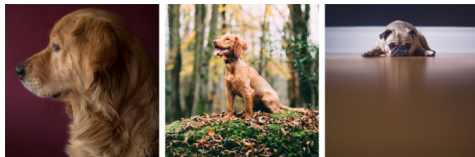


Image: [Going deeper with convolutions](#)

Computationally efficient deeper network:

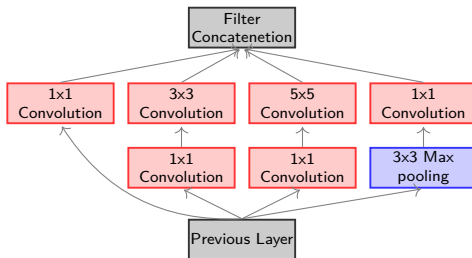
- **Number of parameters:** 5M (12x less than AlexNet, 27x less than VGG-16)
- No fully connected layers at the end. Average pooling across channels.
- 22 layers (with efficient “Inception module”)



Images: from Unsplash

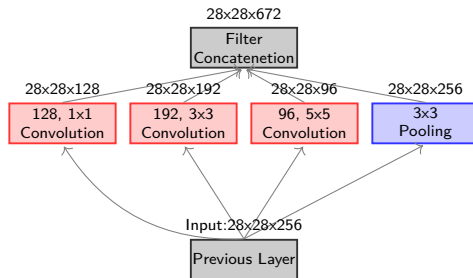
Inception module:

- Design a good local network topology
- Apply filters with different size receptive fields to the input from previous layer (1x1, 3x3, 5x5, 3x3 pooling). Then Concatenate all filter outputs together in channel diminution.
- 'ReLU' activation or convolution modules.
- Why use 1x1 convolutions?



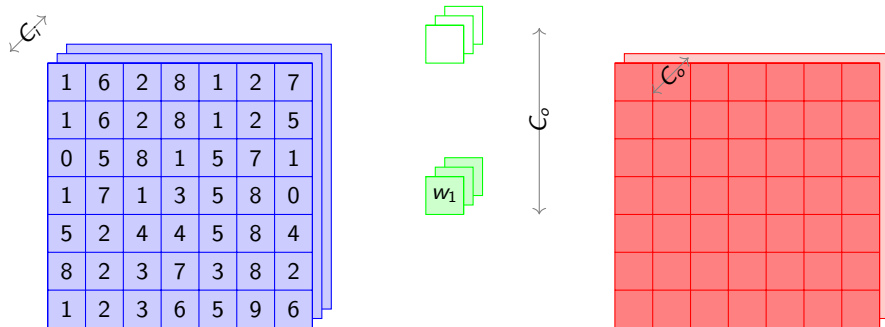
Inception module:

- Design a good local network topology
- Apply filters with different size receptive fields to the input from previous layer (1x1, 3x3, 5x5, 3x3 pooling). Then Concatenate all filter outputs together in channel diminution.
- 'ReLU' activation or convolution modules.
- Why use 1x1 convolutions?

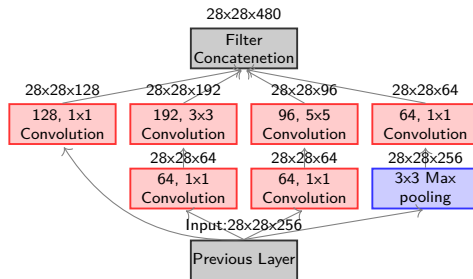


Inception module:

- The channel dimension grows with depth of the network.
- Very expensive to compute.
- Path [3x3] has $3 \times 3 \times 192 \times 256 = 442368$ parameters.



We can do channel dimensionality reduction (increase) with 1x1 convolutions.



- “Bottleneck” modules [1x1] reduce computational cost and output shape.
- Path [3x3] originally had $3 \times 3 \times 192 \times 256 = 442,368$ parameters.
- Path [3x3] now has $1 \times 1 \times 64 \times 256 + 3 \times 3 \times 192 \times 64 = 126,976$ parameters.

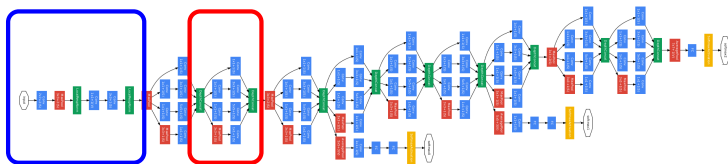


Image: [Going deeper with convolutions](#)

- Some convolution + max pooling at start.
- Stack “Inception module” on top of each other.
- Some intermediate stacking will have max-pooling.

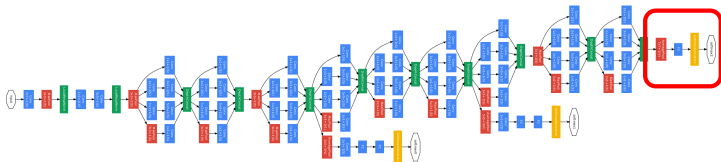


Image: [Going deeper with convolutions](#)

- No FC layers at the end. The output of last inception module is subjected to **global average pooling** and then the final “softmax” layer with 1000 classes.

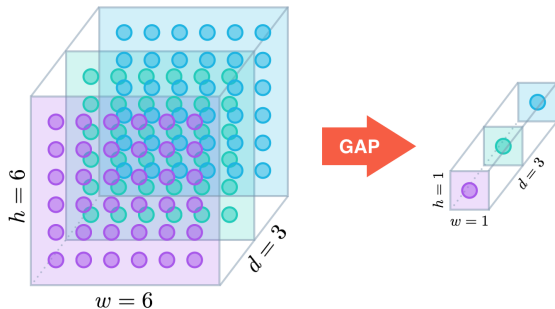


Image: Global Average Pooling Layers for Object Localization

- Normal pooling does each channel independently with 2D masks.
- In GAP, all the pixels in each channel is averaged (or max in global max-pooling) independently to produce a vector.
- If the input to GAP is $[B, H, W, C]$ the output will be $[B, 1, 1, C]$.
- Allows different input shapes at train and test times.
- **First paper to use GAP-Network In Network**

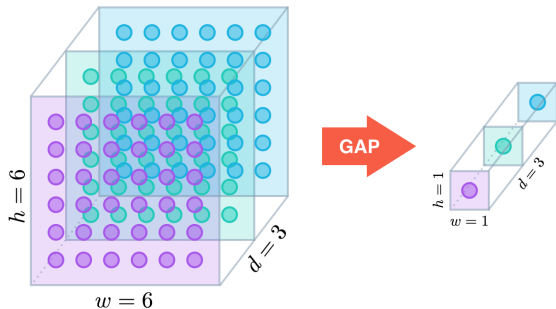
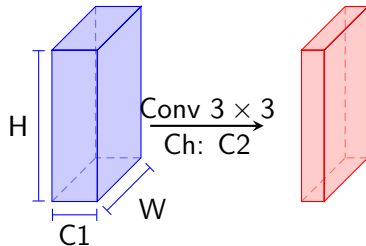


Image: Global Average Pooling Layers for Object Localization

What happens if we change the input tensor height & width to conv layer?



Can we change the input to AlexNet?
No, This will change the dimensions of the input tensor to FC layers.

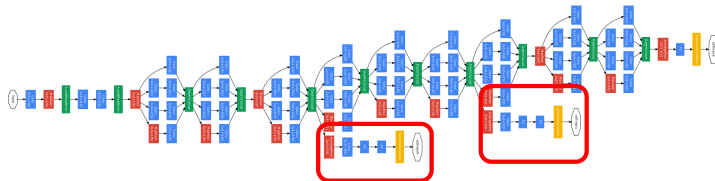


Image: [Going deeper with convolutions](#)

- Deep networks has the issue of vanishing gradients.
- Auxiliary classification outputs to inject additional gradient at lower layers.

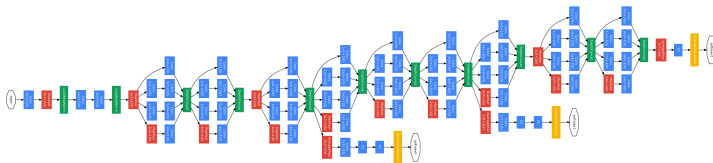


Image: [Going deeper with convolutions](#)

- Stochastic gradient descent with 0.9 momentum.
- Fixed learning rate schedule: decreasing the learning rate by 4% every 8 epochs.
- Data Augmentation and dropout (last layer) for preventing over-fitting.

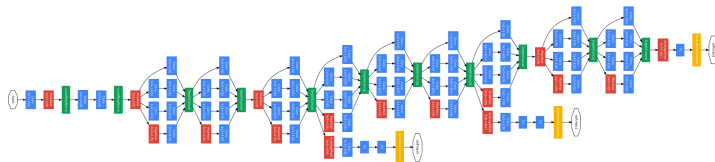
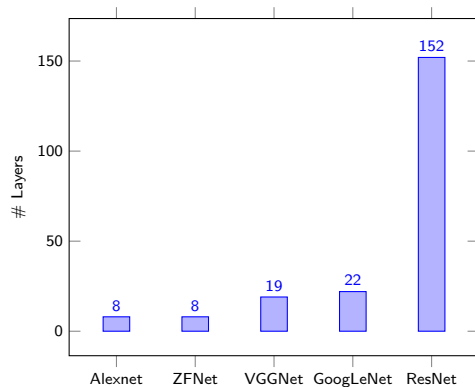
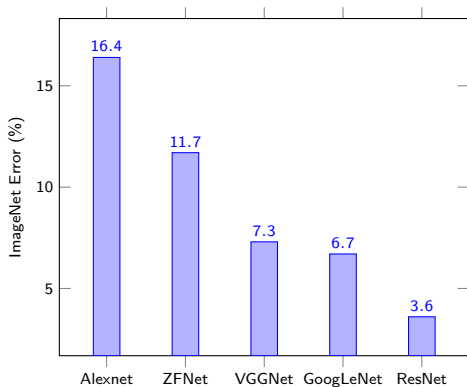


Image: [Going deeper with convolutions](#)

Main ideas:

- Efficient “Inception” module integrates information at multiple receptive fields.
- No FC at the end (use GAP instead) which will reduce the number of parameters significantly.
- Auxiliary classification outputs to inject additional gradient at lower layers.
- ILSVRC 2014 classification winner with 6.7% top 5 error.

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation



Make the network more and more deep to increase performance?

What happens if we increase the depth?

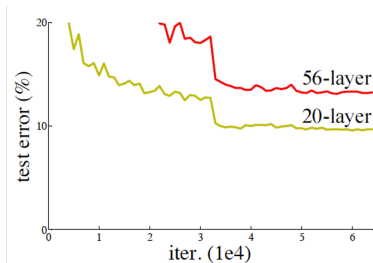


Image: [Deep Residual Learning for Image Recognition](#)

Deeper model performs worse than the shallow model. Maybe over-fitting?

What happens if we increase the depth?

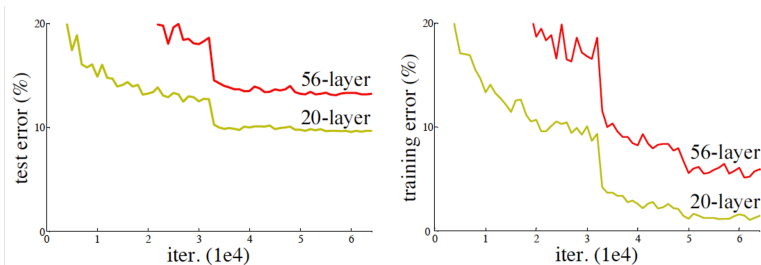


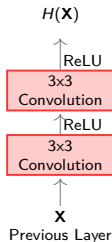
Image: [Deep Residual Learning for Image Recognition](#)

Deeper model performs worse than the shallow model. Maybe over-fitting?

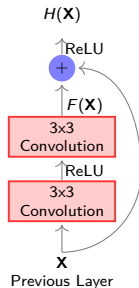
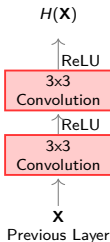
The training error of deeper model is also worse. Not over-fitting.

Hypothesis - Training (optimizing) deeper models is harder.

Learn a residual mapping at each layer, instead of trying to learn the underlying mapping.

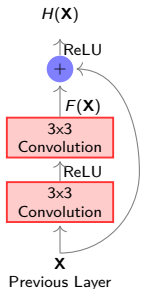


Learn a residual mapping at each layer, instead of trying to learn the underlying mapping.



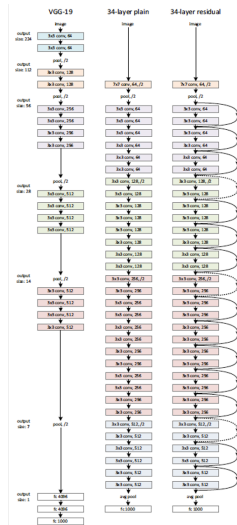
$$H(\mathbf{X}) = F(\mathbf{X}) + \mathbf{X}$$

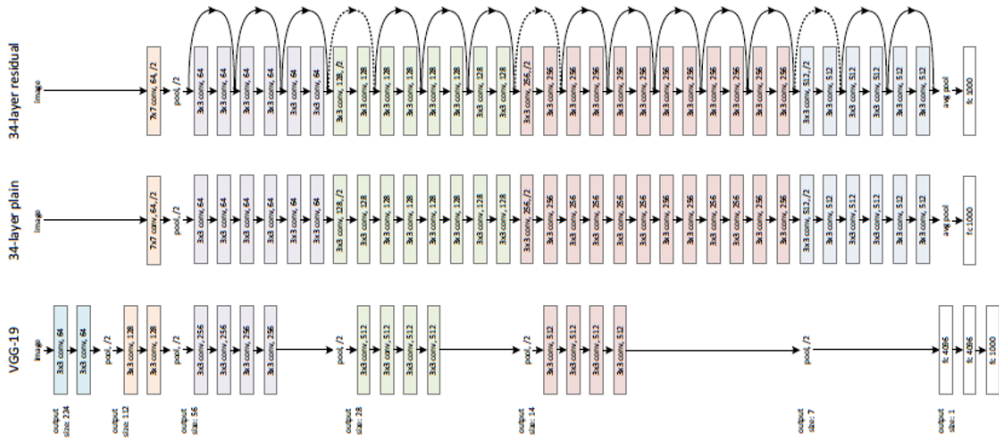
$$F(\mathbf{X}) = H(\mathbf{X}) - \mathbf{X} \quad \triangleright \text{Residual}$$



Full ResNet architecture:

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



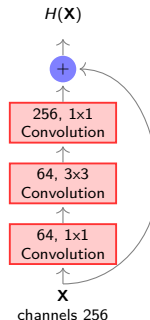


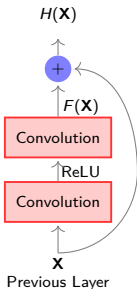
For very deep networks use “bottleneck” blocks to reduce computations.

If $F(\mathbf{X}) = 0$ then only identity: $H(\mathbf{X}) = \mathbf{X}$.

Combined with weight decay we can get some layers to be identity.

Skip connections also provide a direct path for gradients to the bottom layer (close to input).





ILSVRC 2015 classification winner (3.6% top 5 error)

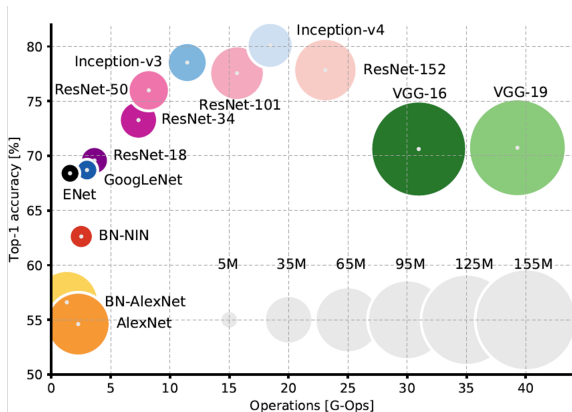
- Many layers: 152 layers on ImageNet, 1202 on Cifar.
- Additional conv layer at the beginning.
- No FC layers at the end (only FC 1000 to output classes)

Training ResNet in practice:

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

- Improve residual block: [Identity Mappings in Deep Residual Networks](#)
- Wide residual blocks not only deep: [Wide Residual Networks](#)
- Residual blocks that share multi scale convolutions from GoogleLeNet: [Aggregated Residual Transformations for Deep Neural Networks \(ResNeXt\)](#)
- Dropout layers in residual block: [Deep Networks with Stochastic Depth \(dropout layers\)](#)
- All convolutions in a residual block also gets skip connections: [Densely Connected Convolutional Networks.](#)

Comparing Different CNN Architectures for Image Classification



- AlexNet low accuracy, high computational cost.
- VGG Highest number of parameters.
- ResNet: Best accuracy, moderate complexity.

Image: [An Analysis of Deep Neural Network Models for Practical Applications.](#)

The size of the blobs is proportional to the number of network parameters

- Mobilenets: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.
- Squeeze Net: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size
- ShuffleNet: ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices

- 1 Image Classification
 - AlexNet
 - VGGNet
 - GoogLeNet
 - ResNet
- 2 Object Detection & Segmentation



Image: [SegNet: Road Scene Segmentation](#)

Predict a category label (class) for each pixel of the image.

Crop patch and do classification of the center pixel using classification CNN?



Image: SegNet: Road Scene Segmentation

Predict a category label (class) for each pixel of the image.

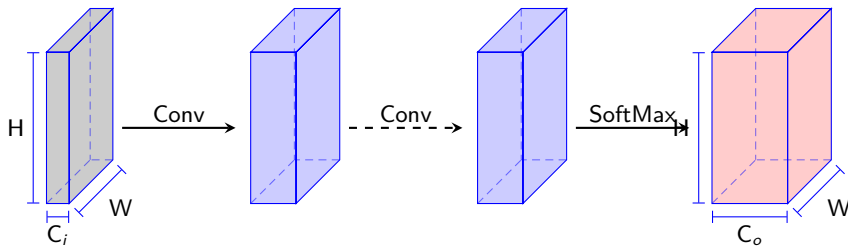
Crop patch and do classification of the center pixel using classification CNN?



Image: [SegNet: Road Scene Segmentation](#)

Predict a category label (class) for each pixel of the image.

Crop patch and do classification of the center pixel using classification CNN?
Very expensive to do inference. Not feasible.



Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once. C_o is the number of classes in the classification problem. Very expensive, large memory requirement.

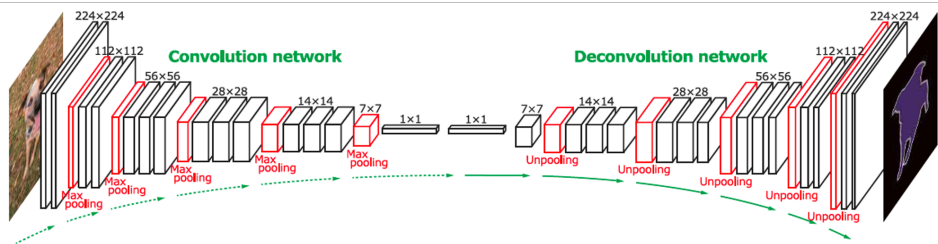


Image: [SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation](#)

Encoder decoder architecture for segmentation.

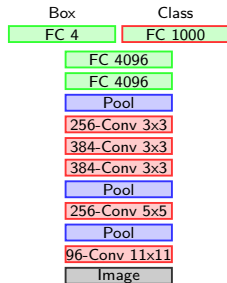
- Encoder First part of the network with convolution and pooling (strided convolutions).
- Decoder Second part with convolution and upsampling (transpose convolutions).

- Fully Convolutional Networks for Semantic Segmentation
- SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation
- Learning Deconvolution Network for Semantic Segmentation
- U-Net: Convolutional Networks for Biomedical Image Segmentation



Need to predict the location as well as the object class. The location is usually defined by a bounding box.

What if there is only one object?



Problem compose to two sub problems:

- Predict object class
- Object location (quantified by $[B_x, B_y, H, W]$)

Can use any CNN (discussed in classification section) and change the last layer to accommodate the two predictions.

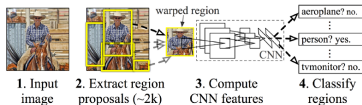
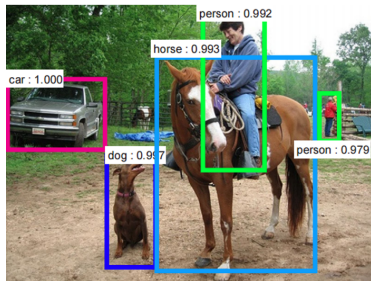
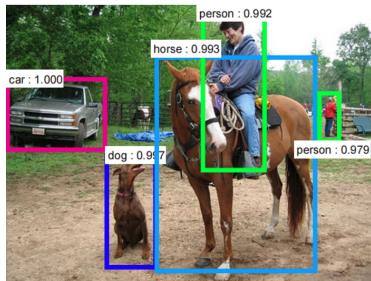


Image: RCNN

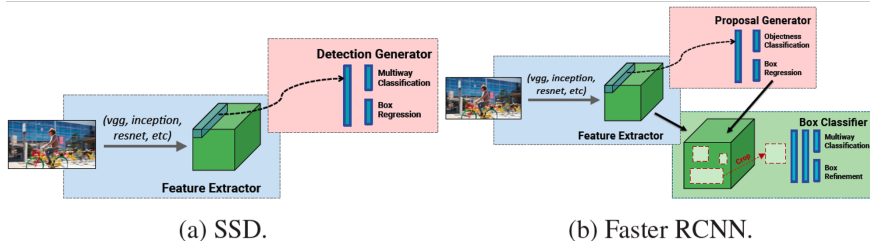
- Use a Region proposal algorithm (in traditional CV) to get initial bounding box proposals.
- Resize each proposed region to fixed size.
- Treat each proposal as a single object detection problem and follow the procedure in last slide.

Initial part subject to errors. Technical difficulties when objects overlap. Need non minimal suppression.



- Use a Region proposal algorithm (in traditional CV) to get initial bounding box proposals.
- Use a CNN based Region proposal algorithm.
- Treat each as a single object detection and follow the procedure in last slide.

End-to-End network for object detection.



- Use a back-borne network for CNN based region proposal algorithms.

- Faster RCNN
- SSD: Single Shot MultiBox Detector
- You Only Look Once: Unified, Real-Time Object Detection (YOLO)
- Speed/accuracy trade-offs for modern convolutional object detectors

Famous networks for image classification, segmentation and object detection.

- **AlexNet**: showed that you can use CNNs to train Computer Vision models.
- **VGG**: shows that bigger networks with smaller conv work better.
- **GoogLeNet**: Focus on efficiency using 1x1 bottleneck convolutions and global avg pool instead of FC layers.
- **ResNet**: showed us how to train extremely deep networks.
- **After ResNet**: CNNs were better than the human metric and focus shifted to Efficient networks:
- Lots of tiny networks aimed at **mobile devices**: MobileNet, ShuffleNet
- *ResNet is currently a good defaults to use.*