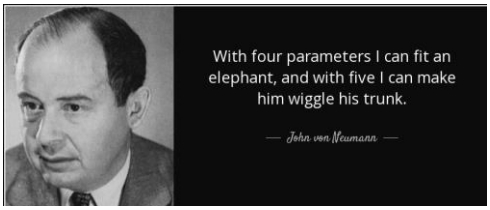
Convolution Neural Networks

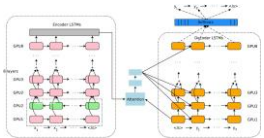
Now you see me



Ref: Mayer, J., Khairy, K., & Howard, J. (2010). Drawing an elephant with four complex parameters. *American Journal of Physics*, *78*(6), 648-649.

Recent Deep Learning Successes and Research Areas

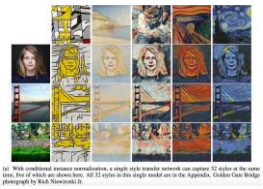
Machine Translation



- Your Google Translate usage will now be powered by an 8 layer Long Short Term Memory Network with residual connections and attention

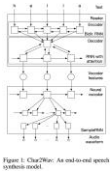
Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation; Wu *et al.*

Artistic Style



A Learned Representation for Artistic Style; Dumoulin, Shlens, Kudlur; ICLR 2017

Speech Synthesis



Char2Wav: End-to-End Speech Synthesis; Sotelo *et al.*, ICLR 2017; http://josesotelo.com/speechsynthesis/

Game Playing

Mastering the game of Go with deep neural networks and tree search; Silver *et al.*, Nature; 2016

Even Star Power! :)

Reasons for the inception of CNNs

1. Imitate the brain's process of visual cognition

2. To overcome some of the problems of basic multi layer perceptron (MLP).

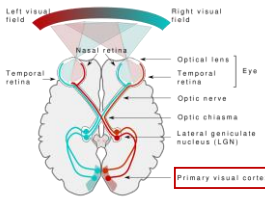
1. Imitating the Brain

Source: https://www.youtube.com/watch?v=40riCqvRoMs

How our brain understand any image?

Source: https://www.youtube.com/watch?v=40riCqvRoMs

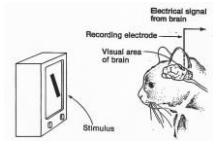
How our brain understand any image?



This is where the magic happens. Let's debunk the magic!

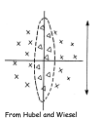
The Visual Pathway. — Source: https://commons.wikimedia.org/wiki/File:Human_visual_pathway.svg

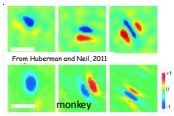
Hubel and Wiesel 1959



- Light of different wavelengths incident on the retina through fully open (slitted) Iris
  - Defines *immediate* (20ms) response of these cells
- Beamed light of different patterns into the eyes and measured neural responses in striate cortex

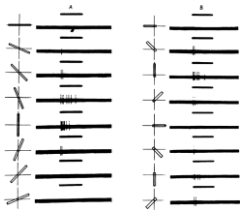
Hubel and Wiesel 1959


From Hubel and Wiesel



- Restricted retinal areas which on illumination influenced the firing of single cortical units were called ***receptive fields***.
  - These fields were usually subdivided into excitatory and inhibitory regions.
- Findings:
  - A light stimulus covering the whole receptive field, or diffuse illumination of the whole retina, was ineffective in driving most units, as excitatory regions cancelled inhibitory regions
    - Light must fall on excitatory regions and NOT fall on inhibitory regions, resulting in clear patterns
  - Receptive fields could be oriented in a vertical, horizontal or oblique manner.
    - Based on the arrangement of excitatory and inhibitory regions within receptive fields.
  - A spot of light gave greater response for some directions of movement than others.

Hubel and Wiesel 59

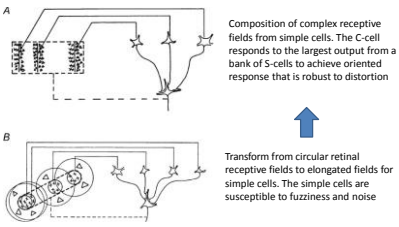


- Response as orientation of input light rotates
  - Note spikes – this neuron is sensitive to vertical bands

Hubel and Wiesel

- Oriented slits of light were the most effective stimuli for activating striate cortex neurons
- The orientation selectivity resulted from *the previous level of input* because lower level neurons responding to a slit also responded to patterns of spots if they were aligned with the same orientation as the slit.
- In a later paper (Hubel & Wiesel, 1962), they showed that within the striate cortex, two levels of processing could be identified
  - Between neurons referred to as *simple* S-cells and *complex* C-cells.
  - Both types responded to oriented slits of light, but complex cells were not "confused" by spots of light while simple cells could be confused

Hubel and Wiesel model



Composition of complex receptive fields from simple cells. The C-cell responds to the largest output from a bank of S-cells to achieve oriented response that is robust to distortion

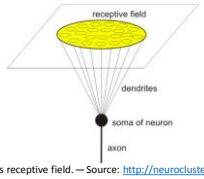
Transform from circular retinal receptive fields to elongated fields for simple cells. The simple cells are susceptible to fuzziness and noise

Hubel and Wiesel

- Complex C-cells build from similarly oriented simple cells
  - They "finetune" the response of the simple cell
- Show complex buildup – building *more complex patterns* by composing early neural responses
  - Successive transformation through Simple-Complex combination layers
- Demonstrated more and more complex responses in later papers
  - Later experiments were on waking macaque monkeys
    - Too horrible to recall

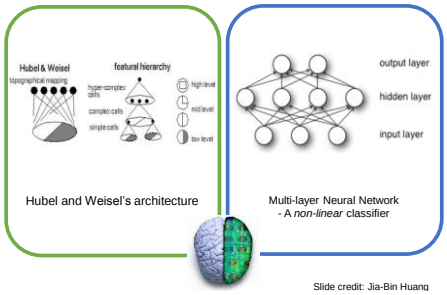
Moral of the Story

- Complex/Abstract cognition is built gradually from simple cells.
- We can fine-tune the output of the simple cells to change the complex cells activation.
- There is something called as a **receptive field** of a single sensory neuron is the specific region of the retina in which something will affect the firing of that neuron (that is, will active the neuron). Every sensory neuron cell has similar receptive fields, and their fields are overlying.



A neuron's receptive field. — Source: http://neuroclusterbrain.com/neuron_model.html

Hubel/Wiesel Architecture and Multi-layer Neural Network



Hubel and Weisel's architecture

Multi-layer Neural Network
- A *non-linear* classifier

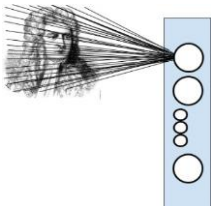
Slide credit: Jia-Bin Huang

Reasons for the inception of CNNs

1. Imitate the brain's process of visual cognition
2. **To overcome some of the problems of basic multi layer perceptron (MLP).**

2. Problems with MLPs

200 X 200 grayscale image, 40k hidden units **around 1.6 Billion parameters!**



2. Problems with MLPs

Now consider the task of Colored Image Recognition

Input Image Size: 200 X 200 X 3 (RGB)

Multi-Layer Perceptron (MLP): Hidden Layer with 40k neurons results in _____parameters.

Convolutional NN: The Savior

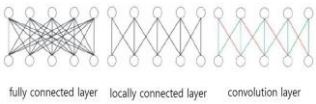
Convolutional Neural Networks is extension of traditional Multi-layer Perceptron, based on 3 ideas:

1. Local receptive fields
2. Shared weights
3. Spatial / temporal sub-sampling

See LeCun paper (1998) on text recognition:
http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

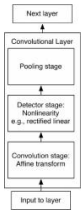
Convolutional NN: The Savior

Convolution Neural Network has smaller number of effective parameters due to **local connections, weight sharing and equivariant representations.**



Convolutional Networks

Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.



Convolution

This operation is called convolution.

$$s(t) = \int x(a)w(t-a)da$$

The convolution operation is typically denoted with an asterisk:

$$s(t) = (x * w)(t)$$

Discrete convolution

If we now assume that **x** and **w** are defined only on integer t, we can define the discrete convolution:

$$s[t] = (x * w)(t) = \sum_{a=-\infty}^{\infty} x[a]w[t-a]$$

In practice

we often use convolutions over more than one axis at a time.

$$s[i,j] = (I * K)[i,j] = \sum_m \sum_n I[m,n]K[i-m, j-n]$$

- The input is usually a multidimensional array of data.
- Discrete convolution can be viewed as multiplication by a matrix
- The kernel is usually a multidimensional array of parameters that should be learned.

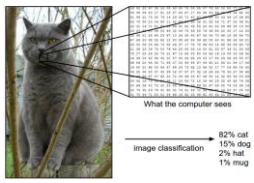
Convolution and Cross-correlation

Many machine learning libraries implement cross-correlation but call it convolution.

Source: https://www.youtube.com/watch?v=Ma0YONjMZLI

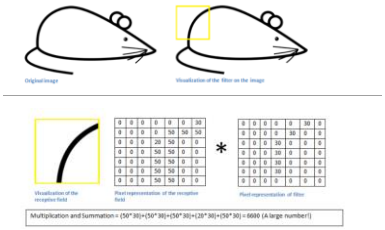
Convolution

- **This convolution operation is actually a similarity measure between the kernel/filter and the original signal/image.**
- **Let's see how:**

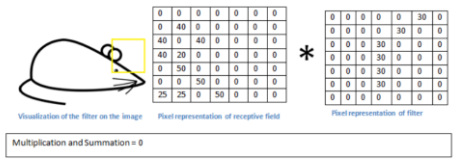


Source: http://cs231n.github.io/classification/

Convolution



Convolution



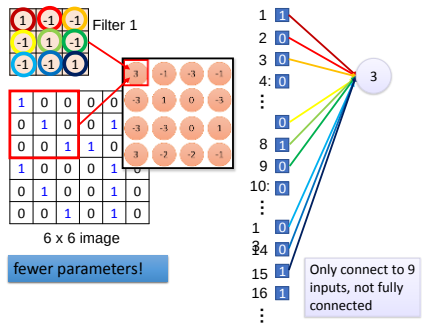
Sparse interactions

- In feed forward neural network every output unit interacts with every input unit.
- Convolutional networks, typically have sparse connectivity (sparse weights)
- This is accomplished by making the kernel smaller than the input.

Sparse interactions

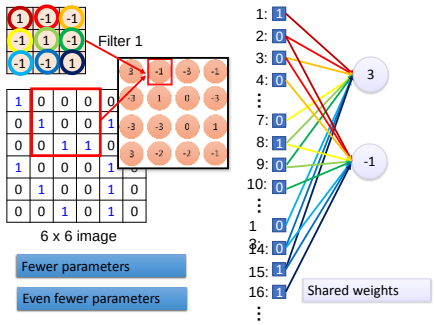
• When we have m inputs and n outputs, then matrix multiplication requires m × n parameters. and the algorithms used in practice have O(m × n) runtime (per example).

Limit the number of connections each output to k, then requires only k × n parameters and O(k × n) runtime.



Parameter sharing

- In a traditional neural net, each element of the weight matrix is multiplied by one element of the input. i.e. It is used once when computing the output of a layer.
- In CNNs each member of the kernel is used at every position of the input
- Instead of learning a separate set of parameters for every location, we learn only one set.



Equivariance

A function $f(x)$ is equivariant to a function g if $f(g(x)) = g(f(x))$.

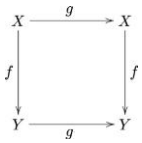


Figure from Wikipedia

Equivariance

A convolutional layer have equivariance to translation.
For example

$$g(x)[i] = x[i-1]$$

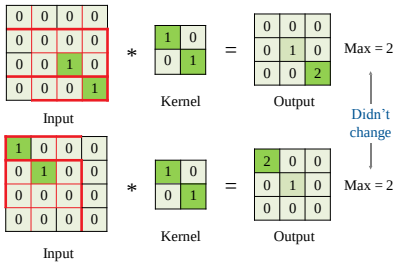
If we apply this transformation to x, then apply convolution, the result will be the same as if we applied convolution to x, then applied the transformation to the output.

Equivariance

For images, convolution creates a 2-D map of where certain features appear in the input.

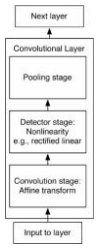
Note that convolution is not equivariant to some other transformations, such as changes in the scale or rotation of an image.

Convolution is translation equivariant

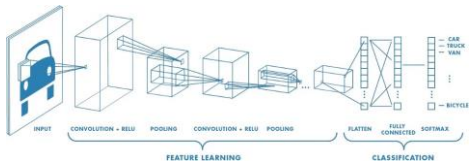


Convolutional Networks

- The first stage (Convolution): the layer performs several convolutions in parallel to produce a set of preactivations.
- The second stage (Detector): each preactivation is run through a nonlinear activation function (e.g. rectified linear).
- The third stage (Pooling)



Typical Architecture of CNNs



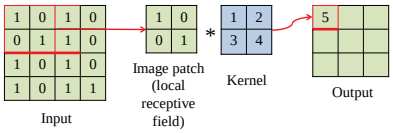
Feature Learning: Part-1 The Convolution layer

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

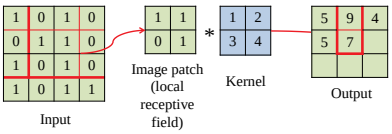
Feature Learning: Part-1 The Convolution layer

Convolution is a dot product of a **kernel** (or filter)
and a patch of an image (**local receptive field**) of the same size

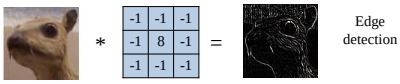


Feature Learning: Part-1 The Convolution layer

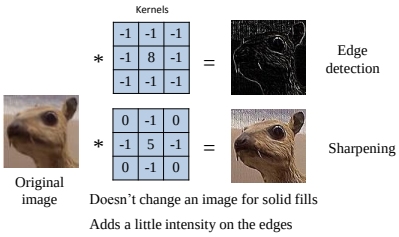
Convolution is a dot product of a **kernel** (or filter)
and a patch of an image (**local receptive field**) of the same size

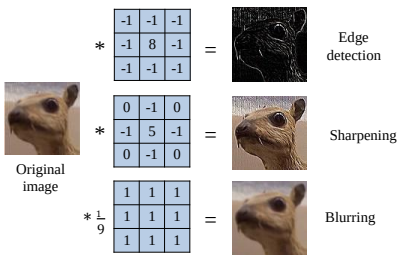


Convolutions have been used for a while



Sums up to 0 (black color) when the patch is a solid fill



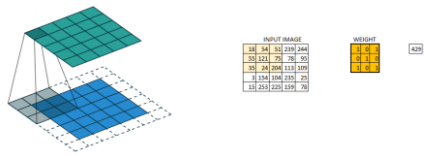


Convolutional Layer: Different Parts

- **Filter**, **Kernel**, or **Feature Detector** is a small matrix used for features detection. A typical filter on the first layer of a ConvNet might have a size [5x5x3].
- **Convolved Feature**, **Activation Map** or **Feature Map** is the output volume formed by sliding the filter over the image and computing the dot product.
- **Receptive field** is a local region of the input volume that has the same size as the filter.
- **Depth** is the number of filters.
- **Depth column** (or **fibre**) is the set of neurons that are all pointing to the same receptive field.
- **Stride** has the objective of producing smaller output volumes spatially. For example, if a stride=2, the filter will shift by the amount of 2 pixels as it convolves around the input volume. Normally, we set the stride in a way that the output volume is an integer and not a fraction. Common stride: 1 or 2 (Smaller strides work better in practice), uncommon stride: 3 or more.
- **Zero-padding** adds zeros around the outside of the input volume so that the convolutions end up with the same number of outputs as inputs. If we don't use padding the information at the borders will be lost after each Conv layer, which will reduce the size of the volumes as well as the performance.

Stride

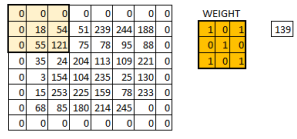
- **Stride** is the size of the step the convolution filter moves each time. A stride size is usually 1, meaning the filter slides pixel by pixel. By increasing the stride size, your filter is sliding over the input with a larger interval and thus has less overlap between the cells.
- The animation below shows stride size 1 and 2 in action.



Padding

- As you can see the size of image keeps on reducing as we increase the stride value. Padding the input image with zeros across it solves this problem for us. We can also add more than one layer of zeros around the image in case of higher stride values.
- We can see how the initial shape of the image is retained after we padded the image with a zero. This is known as **same padding** since the output image has the same size as the input.

Padding



How to compute the output volume[W2xH2xD2]

- W2=(W1−F+2P)/S+1
- H2=(H1−F+2P)/S+1
- D2=K

where:

[W1xH1xD1] : input volume size
F: receptive field size (Same as the kernel size)
S: stride
P: amount of zero padding used on the border.
K: depth

Example

- What is the output volume of the first Convolutional Layer of Krizhevsky et al. architecture that won the ImageNet challenge in 2012?

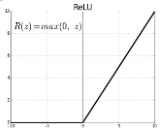
  Input size: [227x227x3], W=227, F=11, S=4, P=0, and K=96.
- Answer:

  (227 - 11) / 4 + 1 = 55

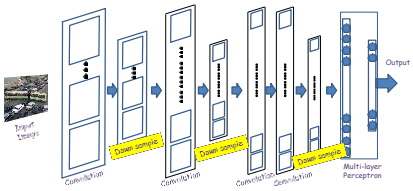
  The size of the Conv layer output volume is [55x55x96].

RELU - The Favourite Non-Linearity

- ReLU stands for Rectified Linear Unit for a non-linear operation. The output is ***f(x) = max(0,x).***
- Why ReLU is important : ReLU's purpose is to introduce non-linearity in our ConvNet. Since, the real world data would want our ConvNet to learn would be non-negative linear values.

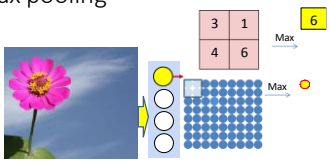


Downsampling/Pooling



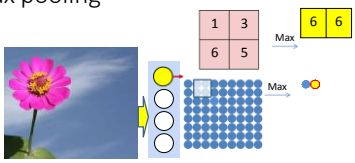
- Convolution (and activation) layers are followed intermittently by "downsampling" (or "pooling") layers
  - Often, they alternate with convolution, though this is not necessary

Max pooling



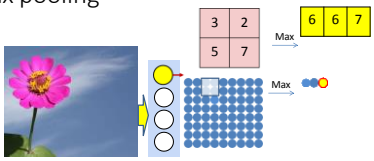
- Max pooling selects the largest from a pool of elements
- Pooling is performed by "scanning" the input

Max pooling



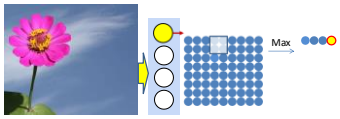
- Max pooling selects the largest from a pool of elements
- Pooling is performed by "scanning" the input

Max pooling



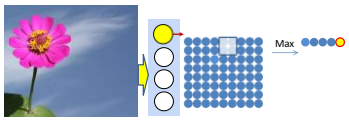
- Max pooling selects the largest from a pool of elements
- Pooling is performed by "scanning" the input

Max pooling



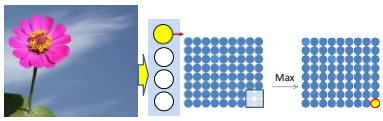
- Max pooling selects the largest from a pool of elements
- Pooling is performed by "scanning" the input

Max pooling



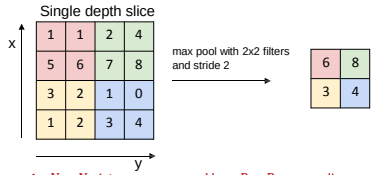
- Max pooling selects the largest from a pool of elements
- Pooling is performed by "scanning" the input

Max pooling



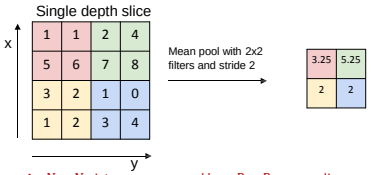
- Max pooling selects the largest from a pool of elements
- Pooling is performed by "scanning" the input

Max Pooling



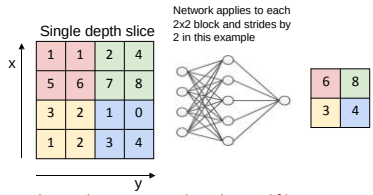
- An $N \times N$ picture compressed by a $P \times P$ maxpooling filter with stride D results in an output map of side $\frac{(N -$

Alternative to Max pooling: Mean Pooling



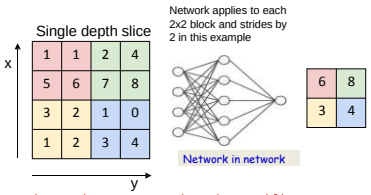
- An $N \times N$ picture compressed by a $P \times P$ maxpooling filter with stride D results in an output map of side $\frac{(N -$

Other options



- The pooling may even be a *learned* filter
  - The *same* network is applied on each block
    - (Again, a shared parameter network)

Other options



- The pooling may even be a *learned* filter
  - The *same* network is applied on each block
    - (Again, a shared parameter network)

Convolution and Pooling as an Infinitely Strong Prior

A weak prior is a prior distribution with high entropy, such a Gaussian distribution with high variance

A strong prior has very low entropy, such as a Gaussian distribution with low variance.

An infinitely strong prior places zero probability on some parameters and says

a convolutional net is similar to a fully connected net with an infinitely strong prior over its weights.

Convolution and Pooling as an Infinitely Strong Prior

The weights for one hidden unit must be identical to the weights of its neighbor, but shifted in space. The weights must be zero, except for in the small, spatially contiguous receptive field assigned to that hidden unit.

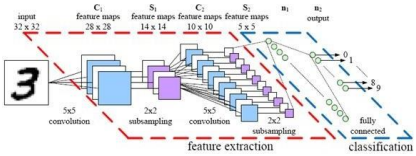
use of convolution as infinitely strong prior probability distribution over the parameters of a layer. This prior says that the function the layer should learn contains only local interactions and is equivariant to translation.

Convolution and Pooling as an Infinitely Strong Prior

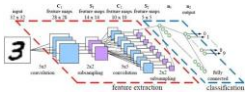
the use of pooling is in infinitely strong prior that each unit should be invariant to small translations.

convolution and pooling can cause underfitting

Putting Everything Together: Case Studies



Le-net 5



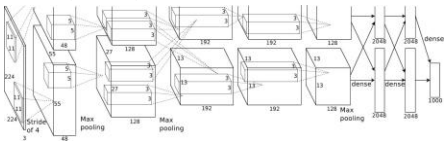
- Digit recognition on MNIST (32x32 images)
  - **Conv1:** 6 5x5 filters in first conv layer (no zero pad), stride 1
    - Result: 6 28x28 maps
  - **Pool1:** 2x2 max pooling, stride 2
    - Result: 6 14x14 maps
  - **Conv2:** 16 5x5 filters in second conv layer, stride 1, no zero pad
    - Result: 16 10x10 maps
  - **Pool2:** 2x2 max pooling with stride 2 for second conv layer
    - Result 16 5x5 maps (400 values in all)
  - **FC:** Final MLP: 3 layers
    - 120 neurons, 84 neurons, and finally 10 output neurons

The imagenet task

- **Imagenet Large Scale Visual Recognition Challenge (ILSVRC)**
- http://www.image-net.org/challenges/LSVRC/
- Actual dataset: Many million images, thousands of categories
- For the evaluations that follow:
  - 1.2 million pictures
  - 1000 categories

AlexNet

- 1.2 million high-resolution images from ImageNet LSVRC-2010 contest
- 1000 different classes (softmax layer)
- NN configuration
  - NN contains 60 million parameters and 650,000 neurons,
  - 5 convolutional layers, some of which are followed by max-pooling layers
  - 3 fully-connected layers



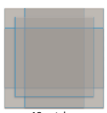
Krizhevsky, A., Sutskever, I. and Hinton, G. E. "ImageNet Classification with Deep Convolutional Neural Networks" NIPS 2012: Neural Information Processing Systems, Lake Tahoe, Nevada

Krizhevsky et. al.

- Input: 227x227x3 images
- Conv1: 96 11x11 filters, stride 4, no zeropad
- Pool1: 3x3 filters, stride 2
- "Normalization" layer [Unnecessary]
- Conv2: 256 5x5 filters, stride 2, zero pad
- Pool2: 3x3, stride 2
- Normalization layer [Unnecessary]
- Conv3: 384 3x3, stride 1, zeropad
- Conv4: 384 3x3, stride 1, zeropad
- Conv5: 256 3x3, stride 1, zeropad
- Pool3: 3x3, stride 2
- FC: 3 layers,
  - 4096 neurons, 4096 neurons, 1000 output neurons

Alexnet: Total parameters

- 650K neurons
- 60M parameters
- 630M connections



10 patches

- Testing: Multi-crop
  - Classify different shifts of the image and vote over the lot!

Learning magic in Alexnet

- **Activations were RELU**
  - Made a large difference in convergence
- "Dropout" – 0.5 (in FC layers only)
- *Large amount of data augmentation*
- SGD with mini batch size 128
- Momentum, with momentum factor 0.9
- L2 weight decay 5e-4
- Learning rate: 0.01, decreased by 10 every time validation accuracy plateaus
- Evaluated using: Validation accuracy

- **Final top-5 error: 18.2% with a single net, 15.4% using an ensemble of 7 networks**
  - **Lowest prior error using conventional classifiers: > 25%**

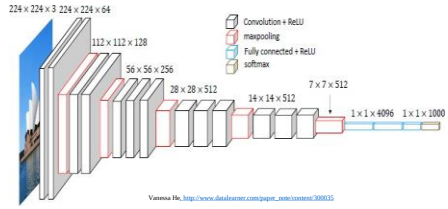
ImageNet

Figure 3: 96 convolutional kernels of size 11×11×3 learned by the first convolutional layer on the 224×224×3 input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

Krizhevsky, A., Sutskever, I. and Hinton, G. E. "ImageNet Classification with Deep Convolutional Neural Networks" NIPS 2012: Neural Information Processing Systems, Lake Tahoe, Nevada

VGG (2015)

- ImageNet top 5 error: 8.0% (single model)



Vanessa He, http://www.datalearner.com/paper_note/content/300035

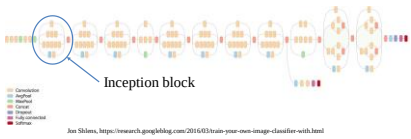
VGG (2015)

- Training similar to AlexNet with additional multi-scale cropping.
- 138 million parameters
- Trains on 4 GPUs for 2-3 weeks
- Strength of VGG as compared to AlexNet was its simplicity
  - 3 x 3 convolutions, same padding, s=1, 2x2 maxpool with s=2

Inception V3 (2015)

Inception V3 (2015)

- Similar to AlexNet? Not quite, uses Inception block introduced in GoogLeNet (a.k.a. Inception V1)
- ImageNet top 5 error: 5.6% (single model), 3.6% (ensemble)

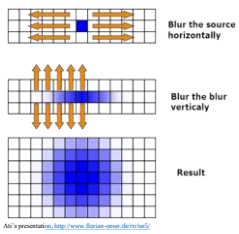


Jon Shlens, https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html

- Batch normalization, image distortions, RMSProp
- 25 million parameters!
- Trains on 8 GPUs for 2 weeks

Filter decomposition

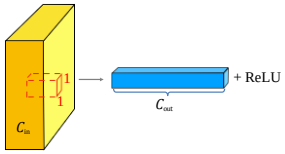
It's known that a Gaussian blur filter can be decomposed in two 1 dimensional filters:



Ati's presentation, http://www.florian-oeser.de/en/ue5/

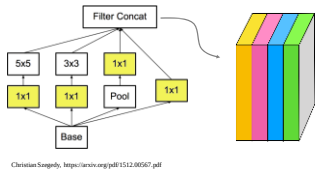
1x1 convolutions

- Such convolutions capture interactions of input channels in one "pixel" of feature map
- They can reduce the number of channels not hurting the quality of the model, because different channels can correlate
- Dimensionality reduction with added ReLU activation



Basic Inception block

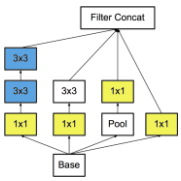
- All operations inside a block use stride 1 and enough padding to output the same spatial dimensions ($W \times H$) of feature map.
- 4 different feature maps are concatenated on depth at the end

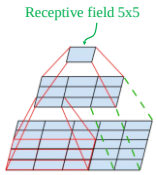


Christian Szegedy, https://arxiv.org/pdf/1512.00567.pdf

Replace 5x5 convolutions

5x5 convolutions are expensive! Let's replace them with two layers of 3x3 convolutions which have an effective receptive field of 5x5.

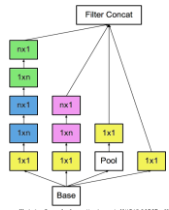




Christian Szegedy, https://arxiv.org/pdf/1512.00567.pdf

Filter decomposition in Inception block

- 3x3 convolutions are currently the most expensive parts!
- Let's replace each 3x3 layer with 1x3 layer followed by 3x1 layer.



Christian Szegedy, https://arxiv.org/pdf/1512.00567.pdf

ResNet (2015)

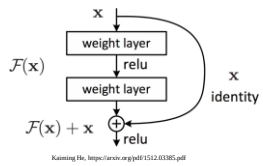
- Introduces residual connections
- ImageNet top 5 error: 4.5% (single model), 3.5% (ensemble)

Kaiming He, https://arxiv.org/pdf/1512.03385.pdf

- 152 layers, few 7x7 convolutional layers, the rest are 3x3, batch normalization, max and average pooling.
- 60 million parameters
- Trains on 8 GPUs for 2-3 weeks.

Residual connections

- We create output channels adding a small delta $F(x)$ to original input channels x:

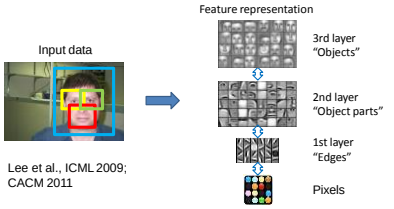


Kaiming He, https://arxiv.org/pdf/1512.03385.pdf

- This way we can stack thousands of layers and gradients do not vanish thanks to residual connections

Learning Feature Hierarchy

Goal: Learn **useful higher-level features** from images



Lee et al., ICML 2009; CACM 2011

Slide: Rob Fergus