Image histogram

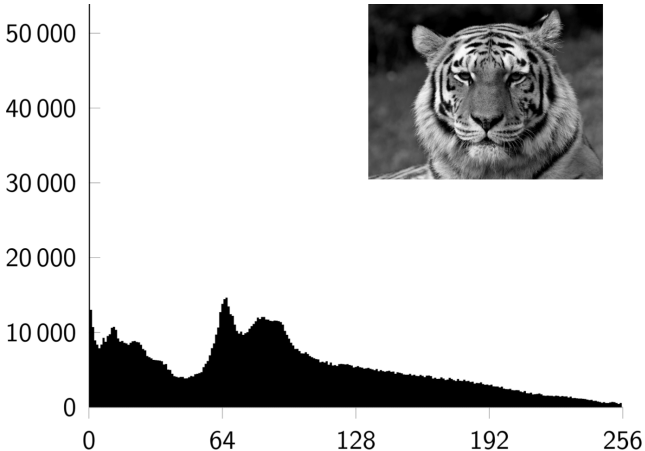


Image histogram

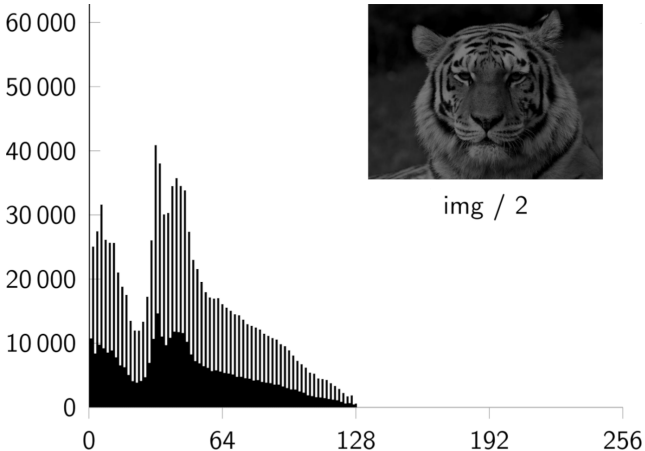


Image values don't fully use available luminance range or concentrate around certain values

Kernel examples

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} =$$

Kernel examples

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} =$$

Kernel examples

$$* \quad \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} =$$

Sharpening

We know how to blur image. How can we sharpen image?

Sharpening

We know how to blur image. How can we sharpen image?

Sharpening

$$* \quad \frac{1}{10} \quad \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline -2 & 22 & -2 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

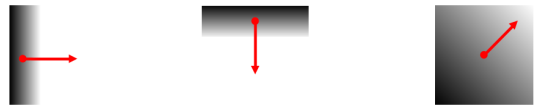
What is depicted on the image?

Image gradient

Gradient vector:
$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

Gradient direction:
$\theta = \text{arctg}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

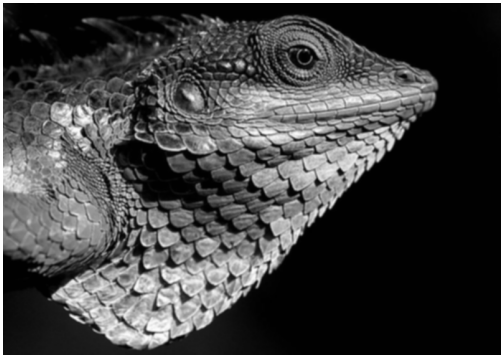
Gradient magnitude:
$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial y}\right)^2 + \left(\frac{\partial f}{\partial x}\right)^2}$

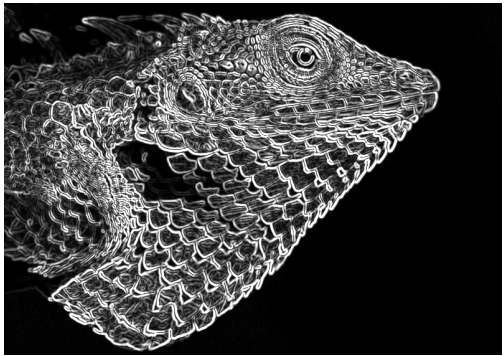


We will use Sobel 3×3 filters for computing gradient:

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

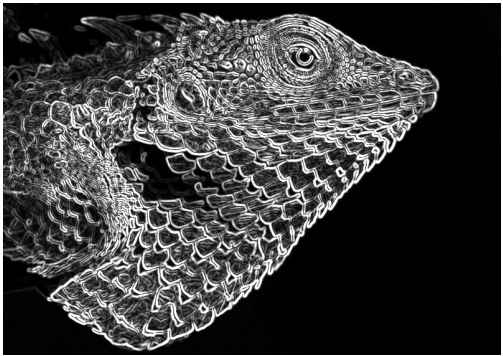
Image gradient

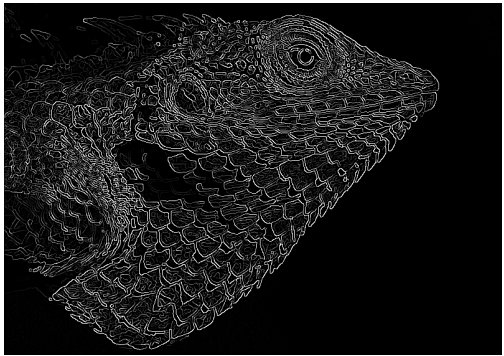


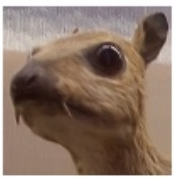
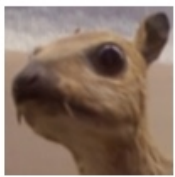
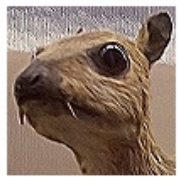


Non-maximum suppression

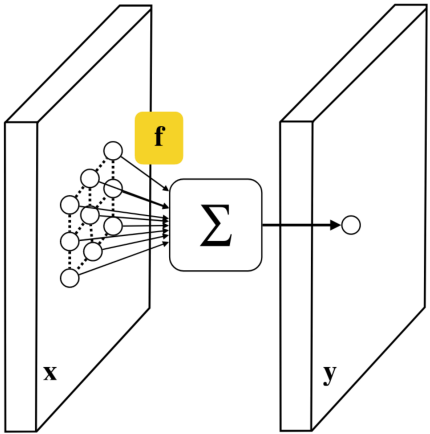
Use either 3×3 max filter or look at the gradient and antigradient neighbour pixels





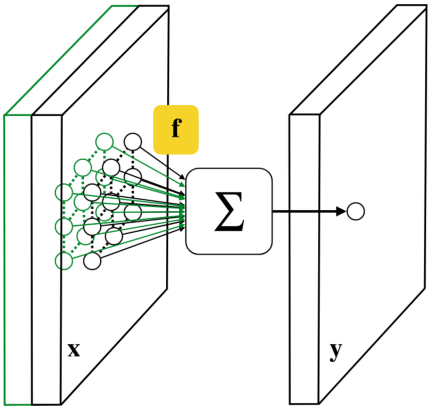
Original	*Gaussian Blur*	*Sharpen*	*Edge Detection*
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
			

Convolutional layer



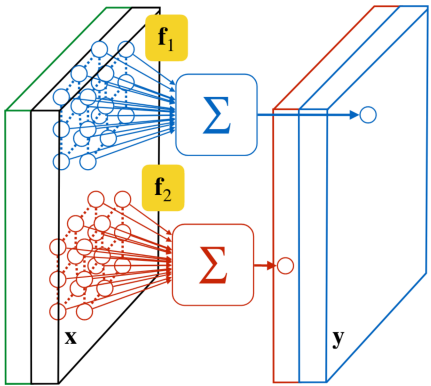
Convolution (linear filtering) for whole image may be modelled using a layer of neurons with shared weights.

Convolutional layer



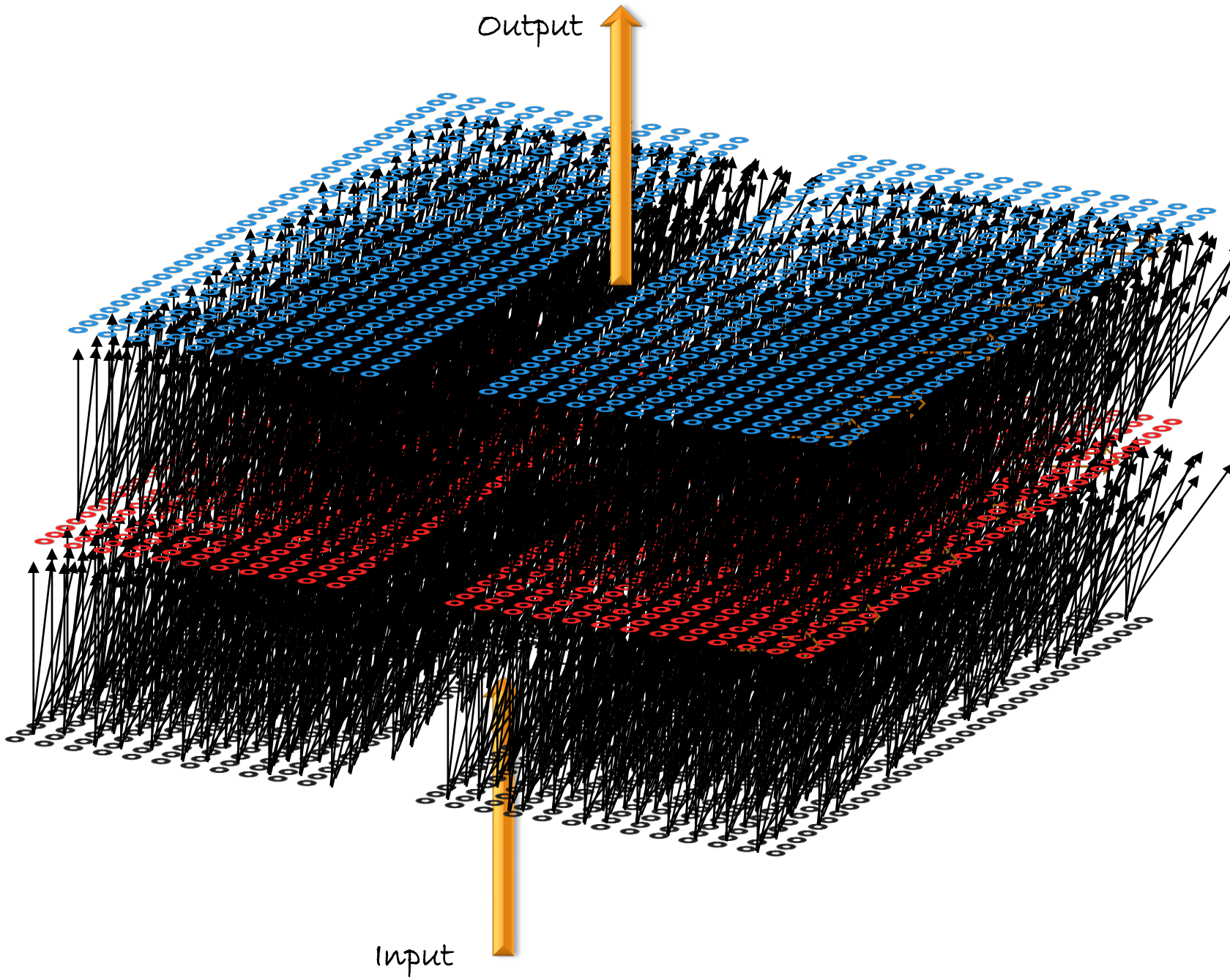
Convolution (linear filtering) for whole image may be modelled using a layer of neurons with shared weights.

Convolutional layer

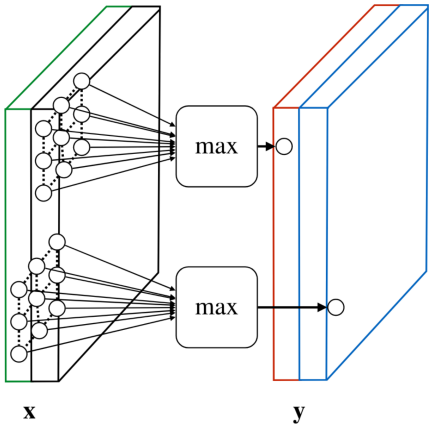


Convolution (linear filtering) for whole image may be modelled using a layer of neurons with shared weights. Convolutional layer is a set of convolutions over the same input

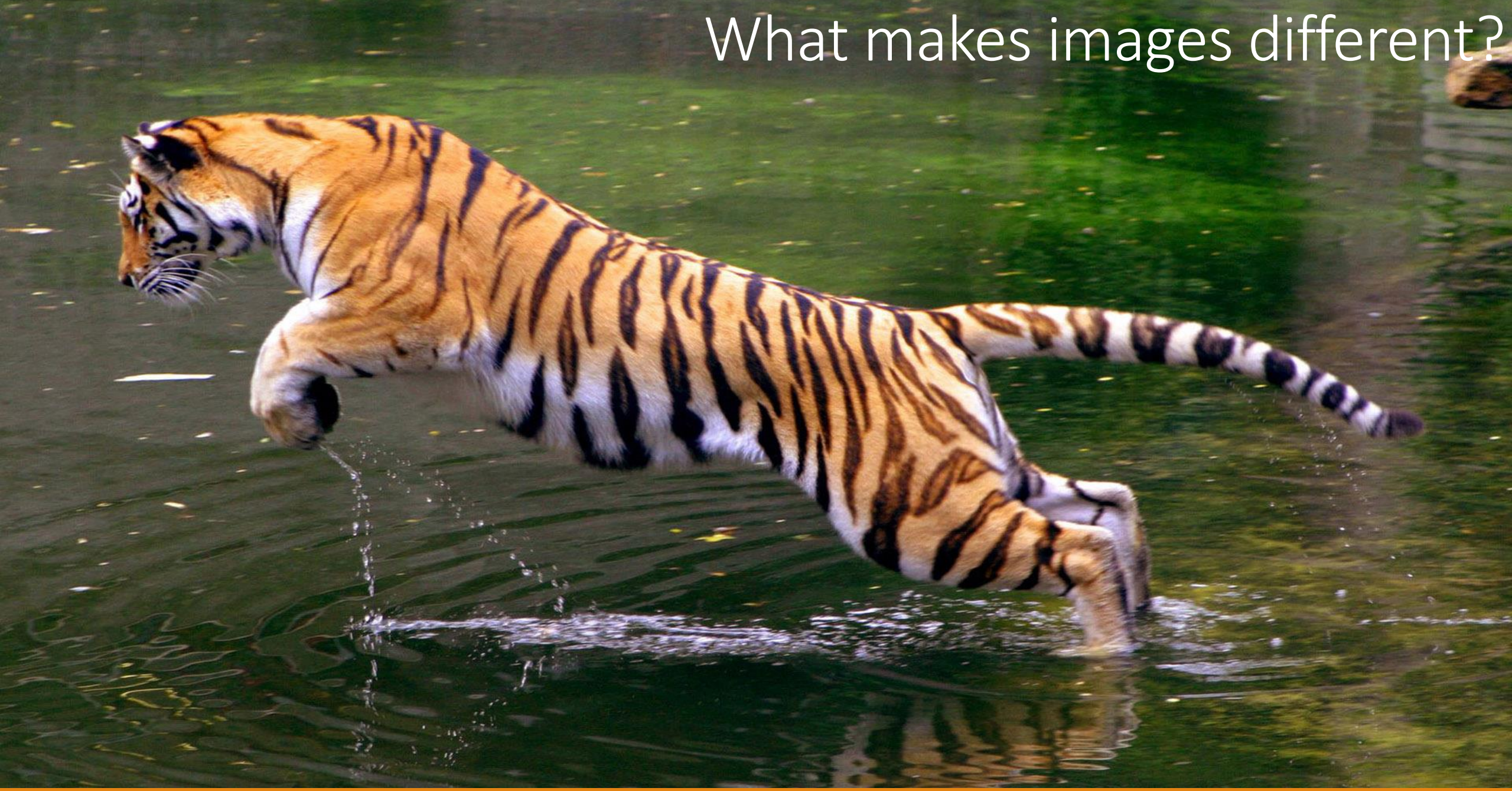
Filters *vs* Convolutional k-d filters



Max pooling layer



What makes images different?



What makes images different?

What makes images different?

- An image has spatial structure
- Huge dimensionality
  - A 256x256 RGB image amounts to ~200K input variables
  - 1-layered NN with 1,000 neurons → 200 million parameters
- Images are stationary signals → they share features
  - After variances images are still meaningful
  - Small visual changes (often invisible to naked eye) → big changes to input vector
  - Still, semantics remain
  - Basic natural image statistics are the same

Input dimensions are correlated

Traditional task: Predict my salary!

Shift 1 dimension

Level of education	Age	Years of experience	Previous job	Nationality
"Higher"	28	6	Researcher	Spain

Level of education	Age	Years of experience	Previous job	Nationality
Spain	"Higher"	28	6	Researcher

Vision task: Predict the picture!

First 5x5 values

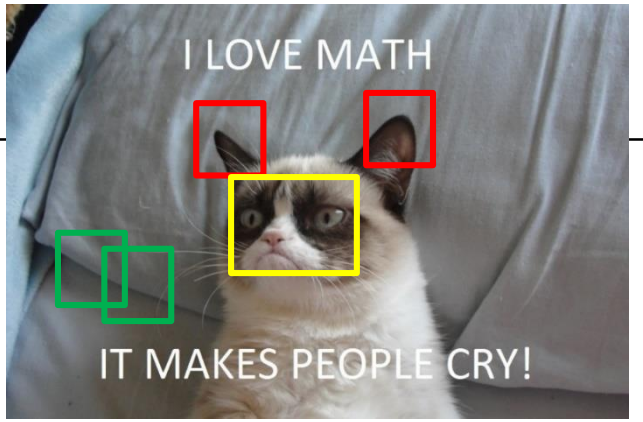
```
array([[51, 49, 51, 56, 55],
       [53, 53, 57, 61, 62],
       [67, 68, 71, 74, 75],
       [76, 77, 79, 82, 80],
       [71, 73, 76, 75, 75]], dtype=uint8)
```

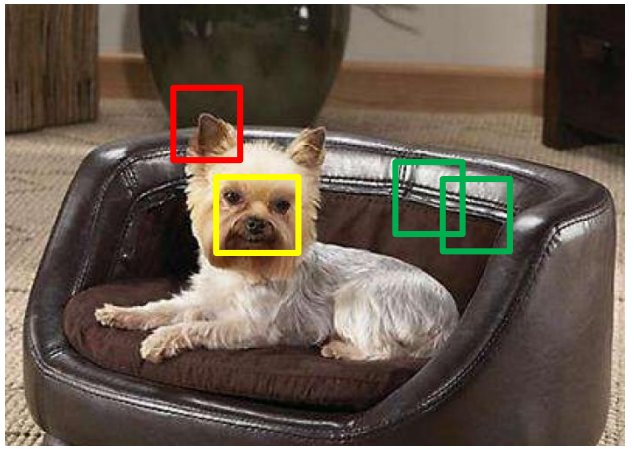
First 5x5 values

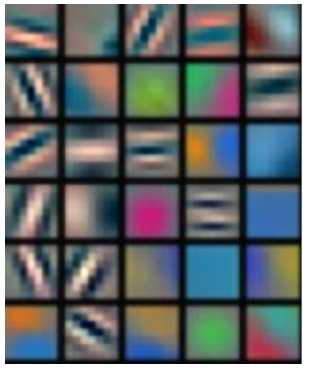
```
array([[58, 57, 57, 59, 59],
       [58, 57, 57, 58, 59],
       [59, 58, 58, 58, 58],
       [61, 61, 60, 60, 59],
       [64, 63, 62, 61, 60]], dtype=uint8)
```

Hypothesis

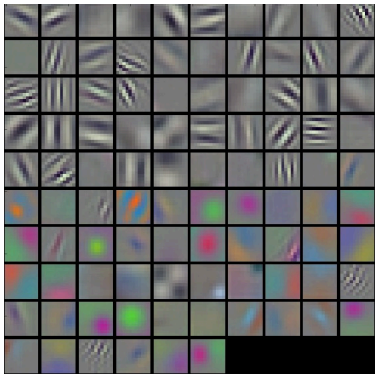
- Imagine
  - With the right amount of data …
  - … and if we connect all inputs of layer l with all outputs of layer $l+1$, …
  - … and if we would visualize the (2d) filters (local connectiveity → 2d) …
  - … we would see very similar filters no matter their location
- Why?
  - Natural images are stationary
  - Visual features are common for different parts of one or multiple image



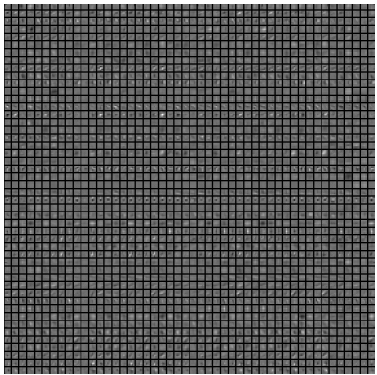




Visualizing filters

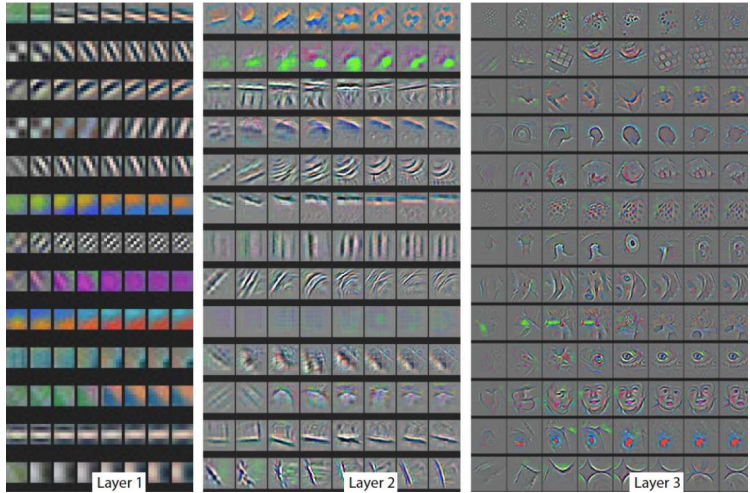


conv1



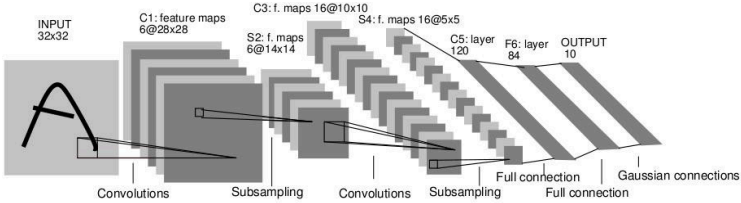
conv2

Visualizing filters with deconvnet during training



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

LeNet

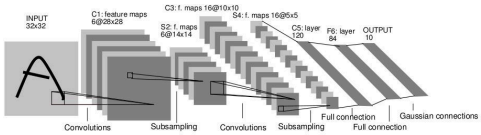


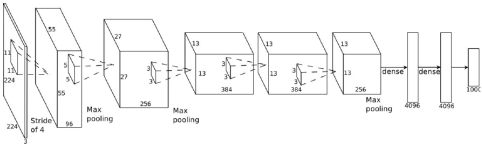
Neocognitron idea + error backpropagation method
→ Convolutional Neural Network (CNN)

Since convolutional neurons share parameters and look at a small neighbourhood, convolutional networks are very effective

LeCun et al. Gradient-based learning applied to document recognition. 1998

LeNet and AlexNet comparison





1998:

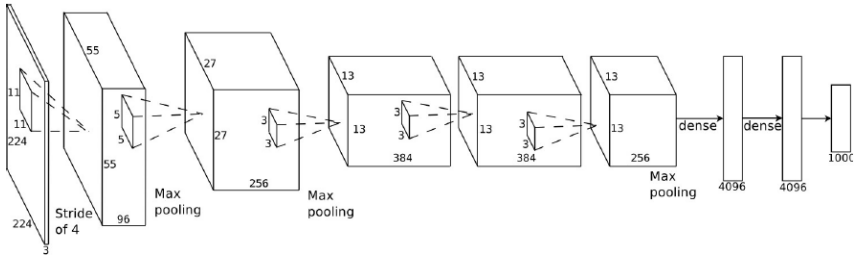
- 2 conv layers (6, 16 filters)
- 2 fully connected layers (120, 84 neurons)

2012:

- 5 conv layers (96, 256, 384, 384, 256 filters)
- 2 fully connected layers (4096, 4096 neurons)

Krizhevsky A., Sutskever I., Hinton G. E. Imagenet classification with deep convolutional neural networks. NIPS 2012

AlexNet

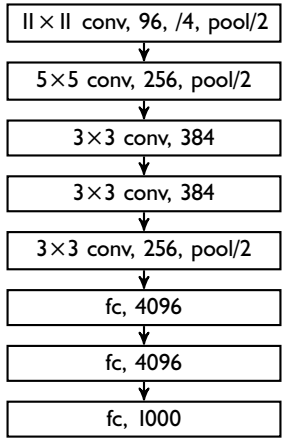


- 60M parameters
- 2GPU × 3GB, 5GB RAM, 27GB HDD
- 1 week to train

Key ideas:

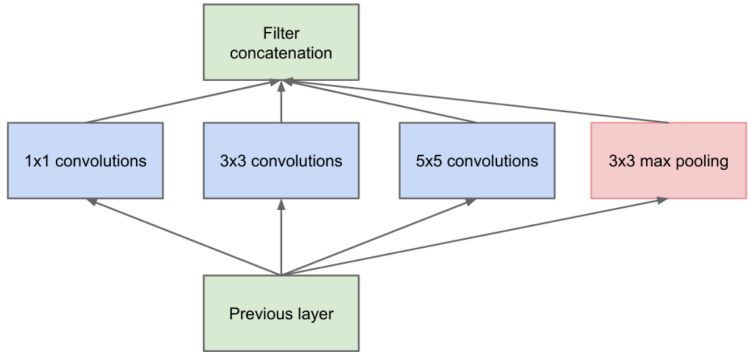
- ReLU activation
- image augmentations
- dropout

AlexNet



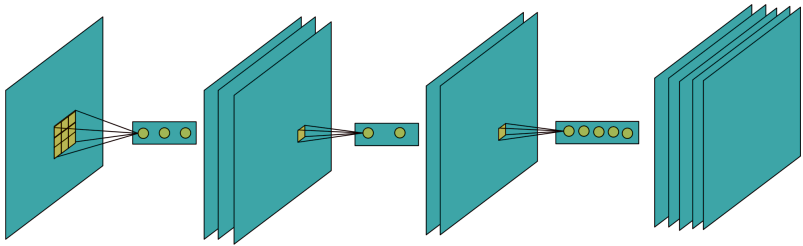
Krizhevsky et al. Imagenet classification with deep convolutional neural networks. NIPS 2012

Inception block



Szegedy et al. Going deeper with convolutions. CVPR 2015

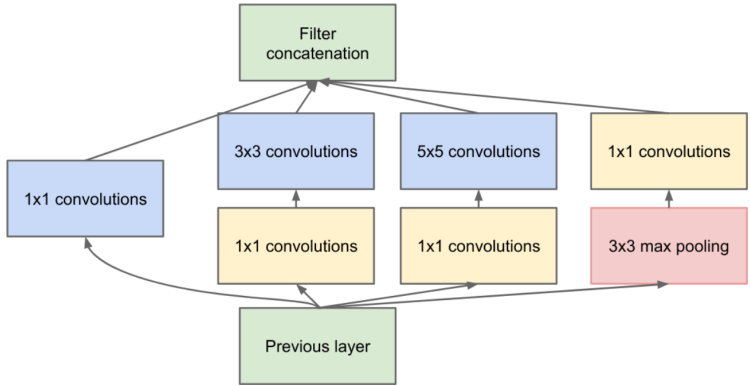
1×1 convolutions



1×1 convolution maps N_{in} channels to N_{out} channels.
May be used as:

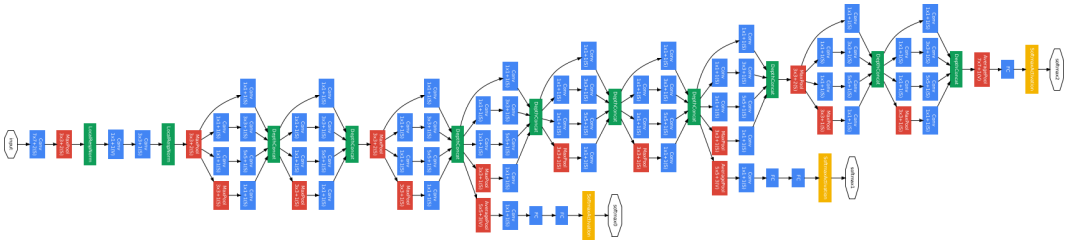
- a set of local classifiers
- a method for expanding ($N_{\text{in}} < N_{\text{out}}$) or reducing ($N_{\text{in}} > N_{\text{out}}$) tensor depth

Inception block with dim reduction



Szegedy et al. Going deeper with convolutions. CVPR 2015

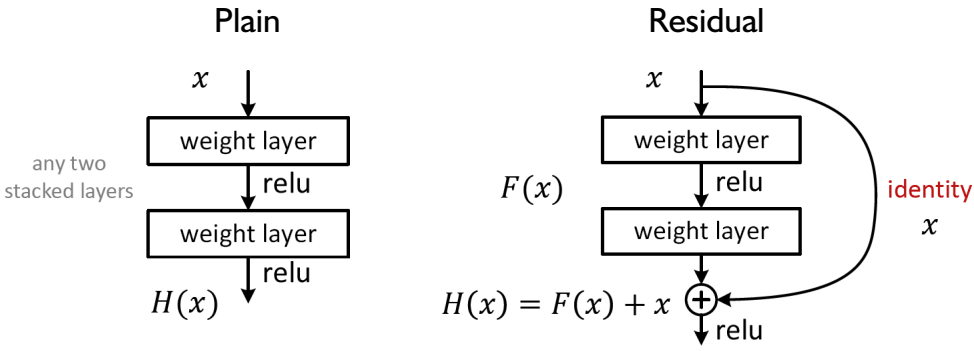
Inception architecture



Deep network made of inception blocks. To make training more stable, uses several heads for supervision

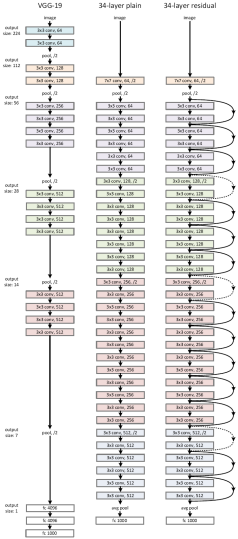
Szegedy et al. Going deeper with convolutions. CVPR 2015

Residual block



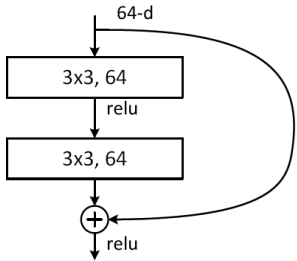
Skip connections will help network learn additive component to the identity function. Gradient are able now to flow through skip connections

ResNet



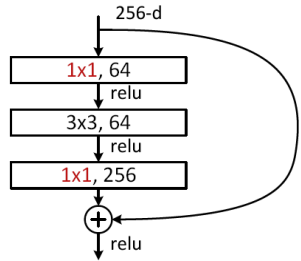
Only 3×3 convolutions, subsampling using stride 2

Repeating residual bottleneck blocks:



all-3x3

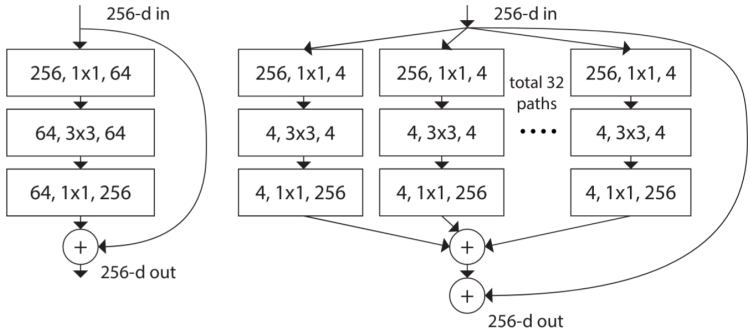
similar complexity



bottleneck

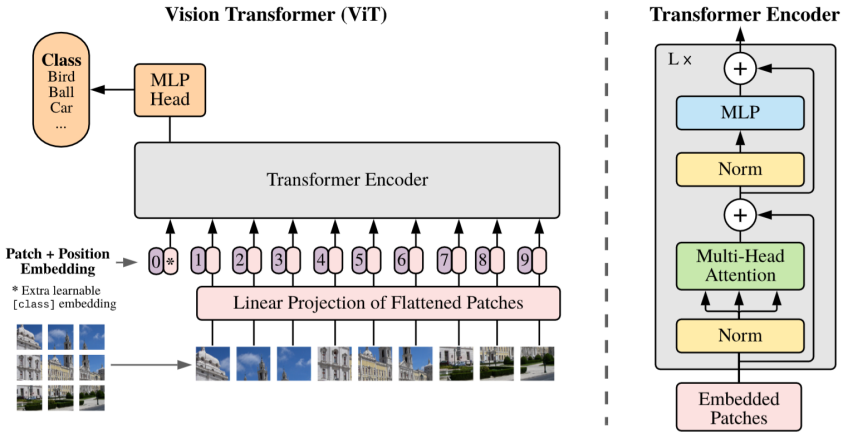
(for ResNet-50/101/152)

ResNeXt



Xie et al. Aggregated Residual Transformations for Deep Neural Networks. CVPR 2017

Vision Transformer



Dosovitskiy et al. An Image is Worth 16× 16 Words: Transformers for Image Recognition at Scale. ICLR 2021