WELCOME TO THE

UCL x DeepMind lecture series

In this lecture series, leading research scientists from leading AI research lab, DeepMind, will give 12 lectures on an exciting selection of topics in Deep Learning, ranging from the fundamentals of training neural networks via advanced ideas around memory, attention, and generative modelling to the important topic of responsible innovation.

Please join us for a deep dive lecture series into Deep Learning!

#UCLxDeepMind

General information

Exits:

At the back, the way you came in

Wifi:

UCL guest

TODAY’S SPEAKER

Sander Dieleman

Sander Dieleman is a Research Scientist at DeepMind in London, UK, where he he has worked on the development of AlphaGo and WaveNet. He was previously a PhD student at Ghent University, where he conducted research on feature learning and deep learning techniques for learning hierarchical representations of musical audio signals. During his PhD he also developed the deep learning library Lasagne and won solo and team gold medals respectively in Kaggle's "Galaxy Zoo" competition and the first National Data Science Bowl. In the summer of 2014, he interned at Spotify in New York, where he worked on implementing audio-based music recommendation using deep learning on an industrial scale.

TODAY'S LECTURE

Convolutional Neural Networks for Image Recognition

In the past decade, convolutional neural networks have revolutionised computer vision. In this lecture, we will take a closer look at convolutional network architectures through several case studies, ranging from the early 90's to the current state of the art. We will review some of the building blocks that are in common use today, discuss the challenges of training deep models, and strategies for finding effective architectures, with a focus on image recognition.

DeepMind

Convolutional Neural Networks for Image Recognition

Sander Dieleman

UCL x DeepMind Lectures

Plan for this lecture

01
Background

02
Building blocks

03
Convolutional neural networks

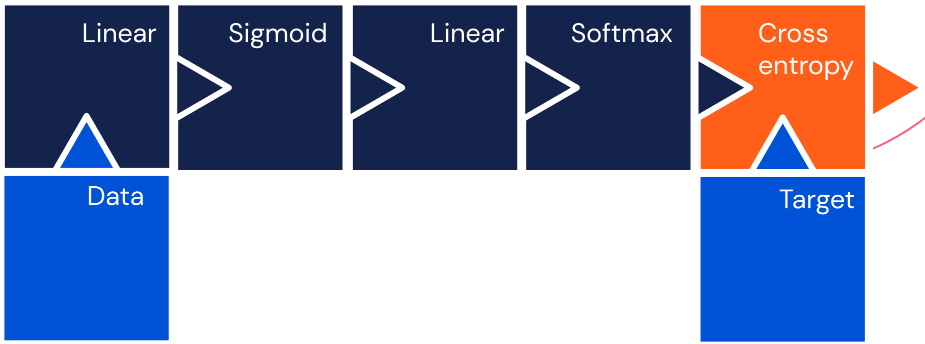
04
Going deeper: case studies

05
Advanced topics

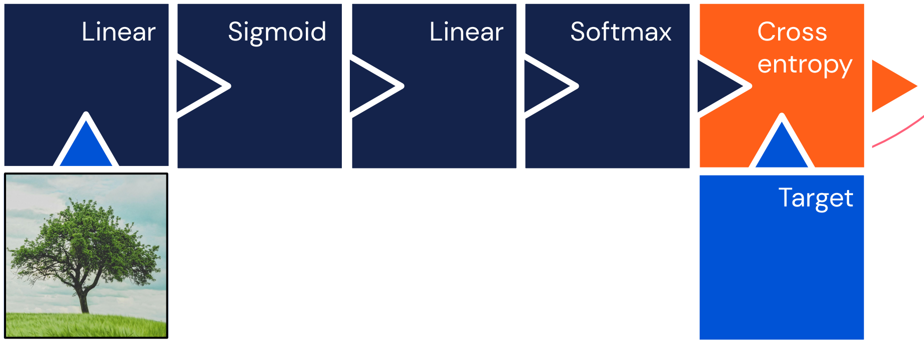
06
Beyond image recognition

1 Background

Last week: neural networks


Linear
Sigmoid
Linear
Softmax
Cross entropy
Data
Target

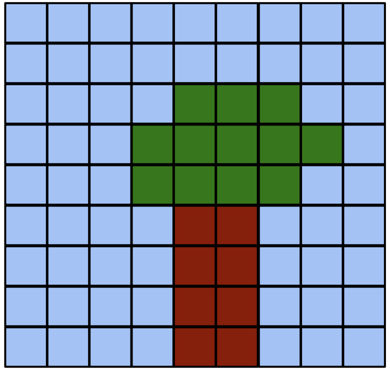
How can we feed images to a neural network?


Linear
Sigmoid
Linear
Softmax
Cross
entropy
Target

Neural networks for images

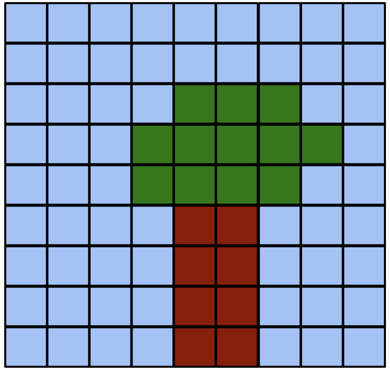
A digital image is a 2D **grid of pixels**.

Neural networks for images



A digital image is a 2D **grid of pixels**.

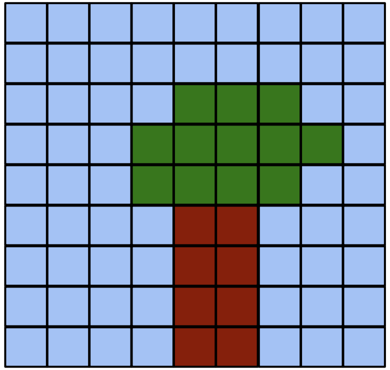
Neural networks for images



A digital image is a 2D **grid of pixels**.

A neural network expects a **vector of numbers** as input.

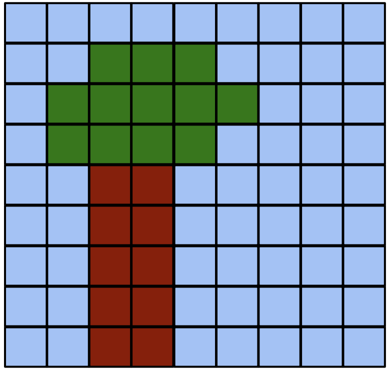
Neural networks for images



A digital image is a 2D **grid of pixels**.

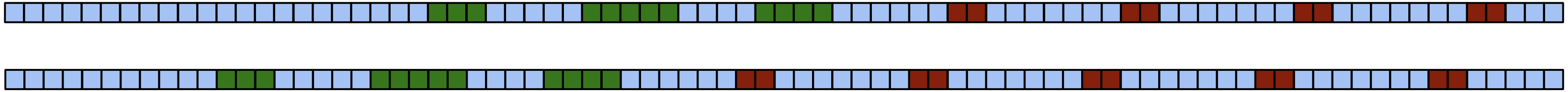
A neural network expects a **vector of numbers** as input.

Neural networks for images



A digital image is a 2D **grid of pixels**.

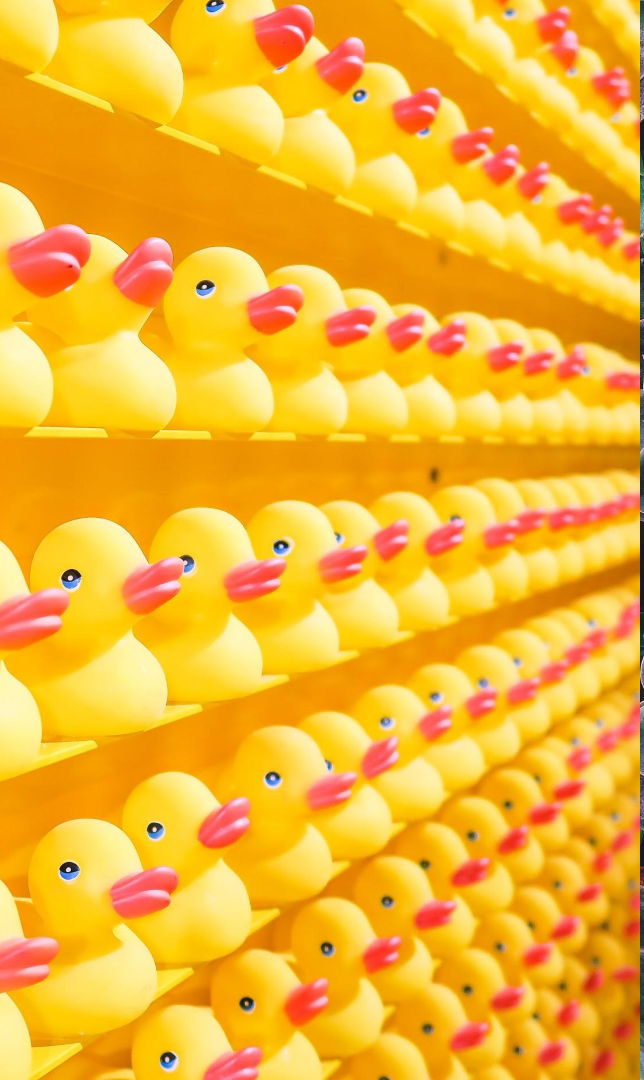
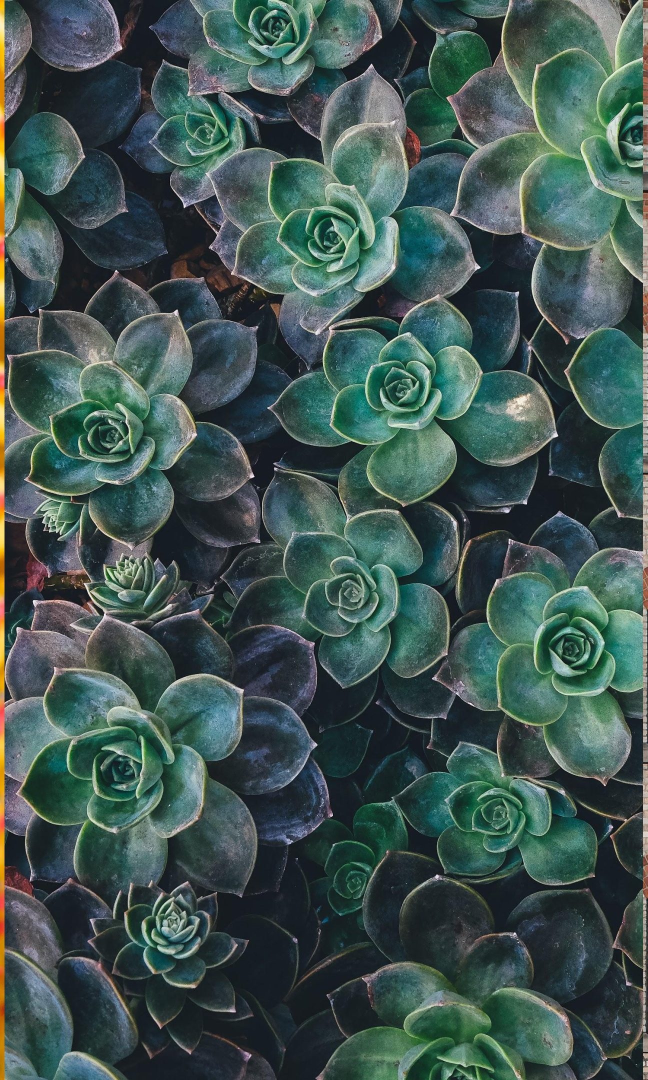
A neural network expects a **vector of numbers** as input.

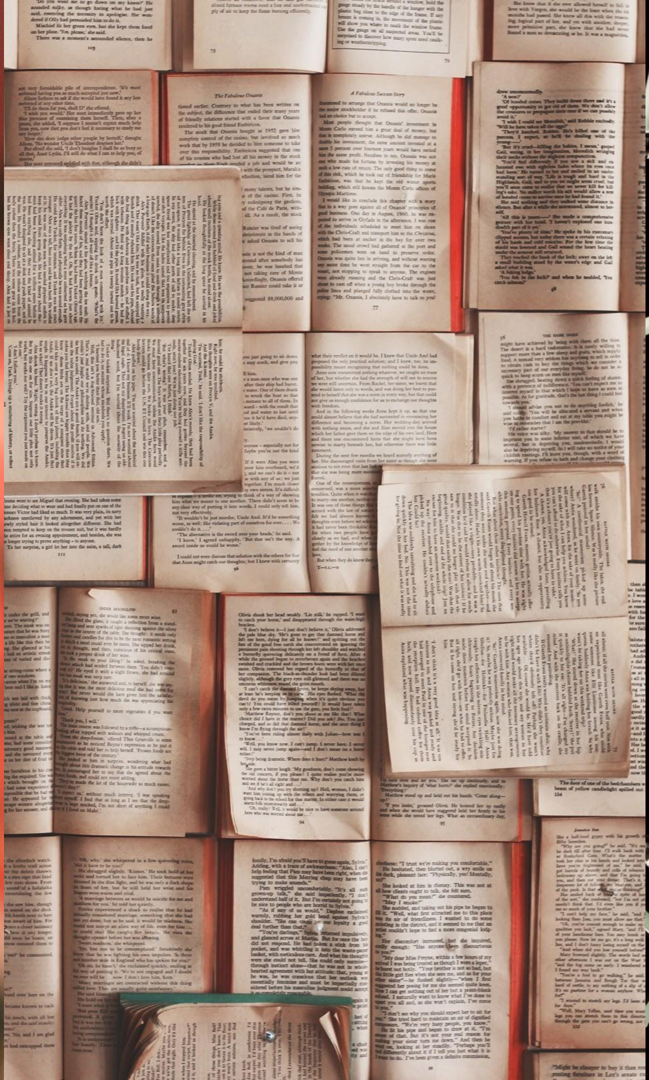


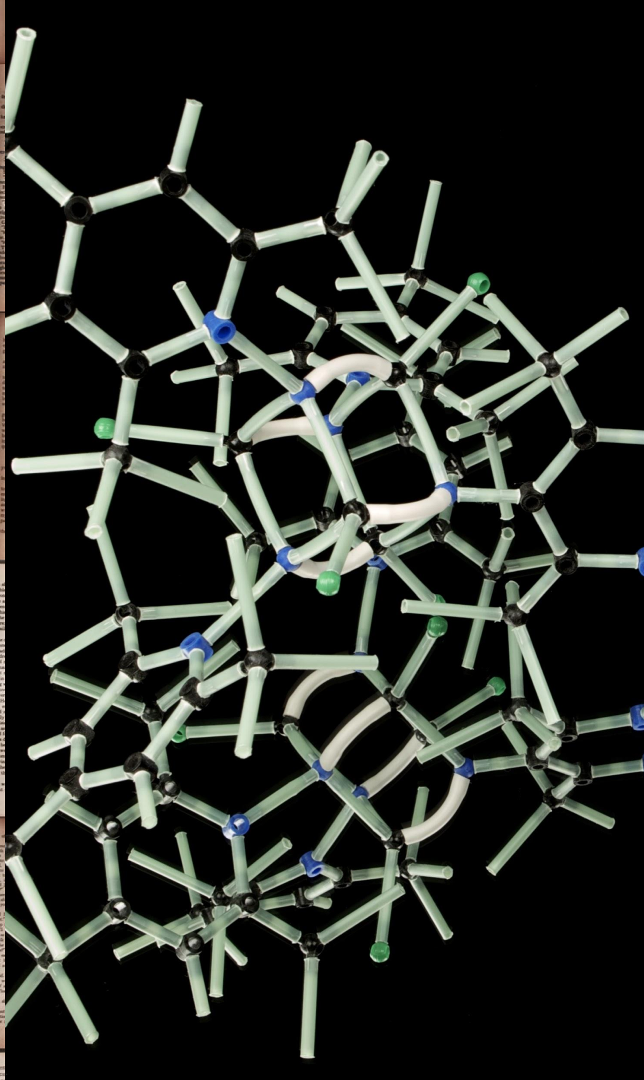
Locality and translation invariance

Locality: nearby pixels are more strongly correlated

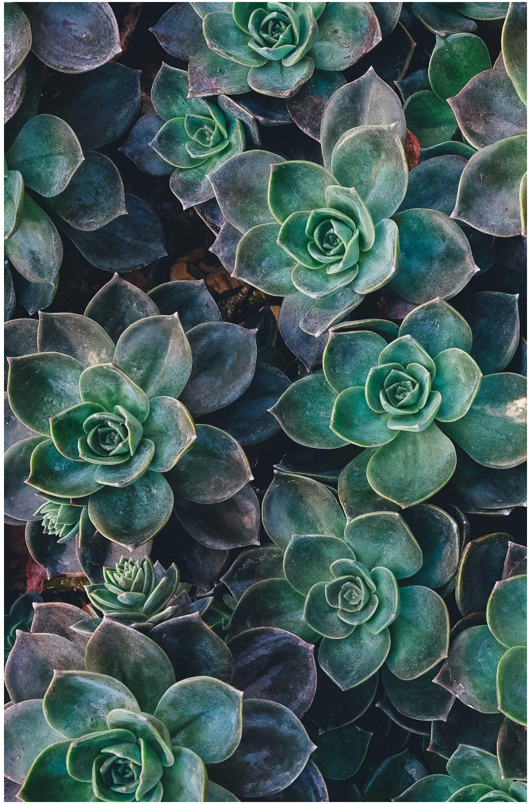
Translation invariance: meaningful patterns can occur anywhere in the image

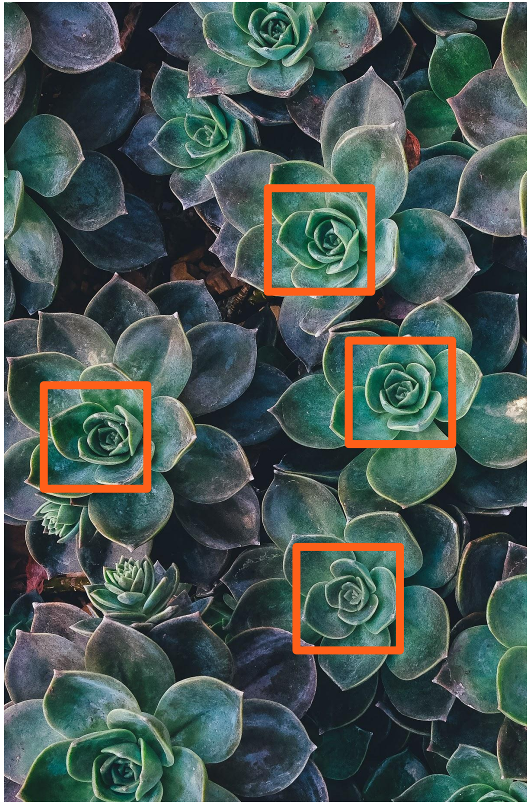





Taking advantage of topological structure

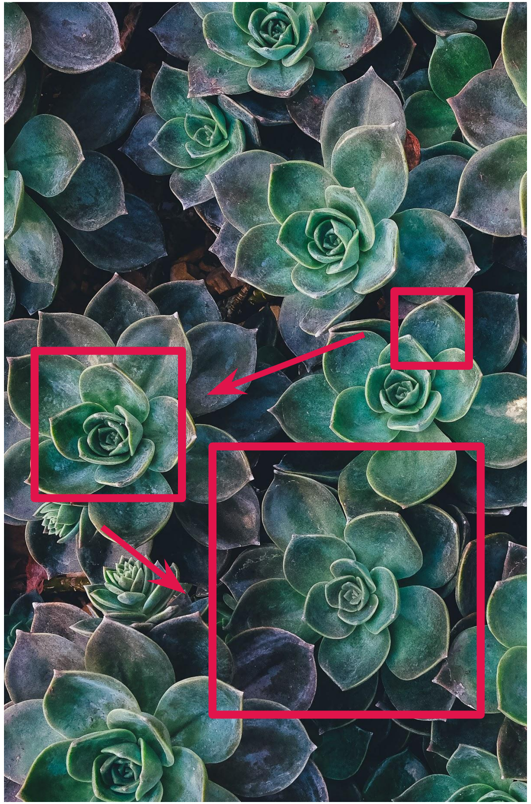


Taking advantage of topological structure



Weight sharing: use the same network parameters to detect local patterns at many locations in the image

Taking advantage of topological structure



Weight sharing: use the same network parameters to detect local patterns at many locations in the image

Hierarchy: local low-level features are composed into larger, more abstract features

edges and textures | object parts | objects

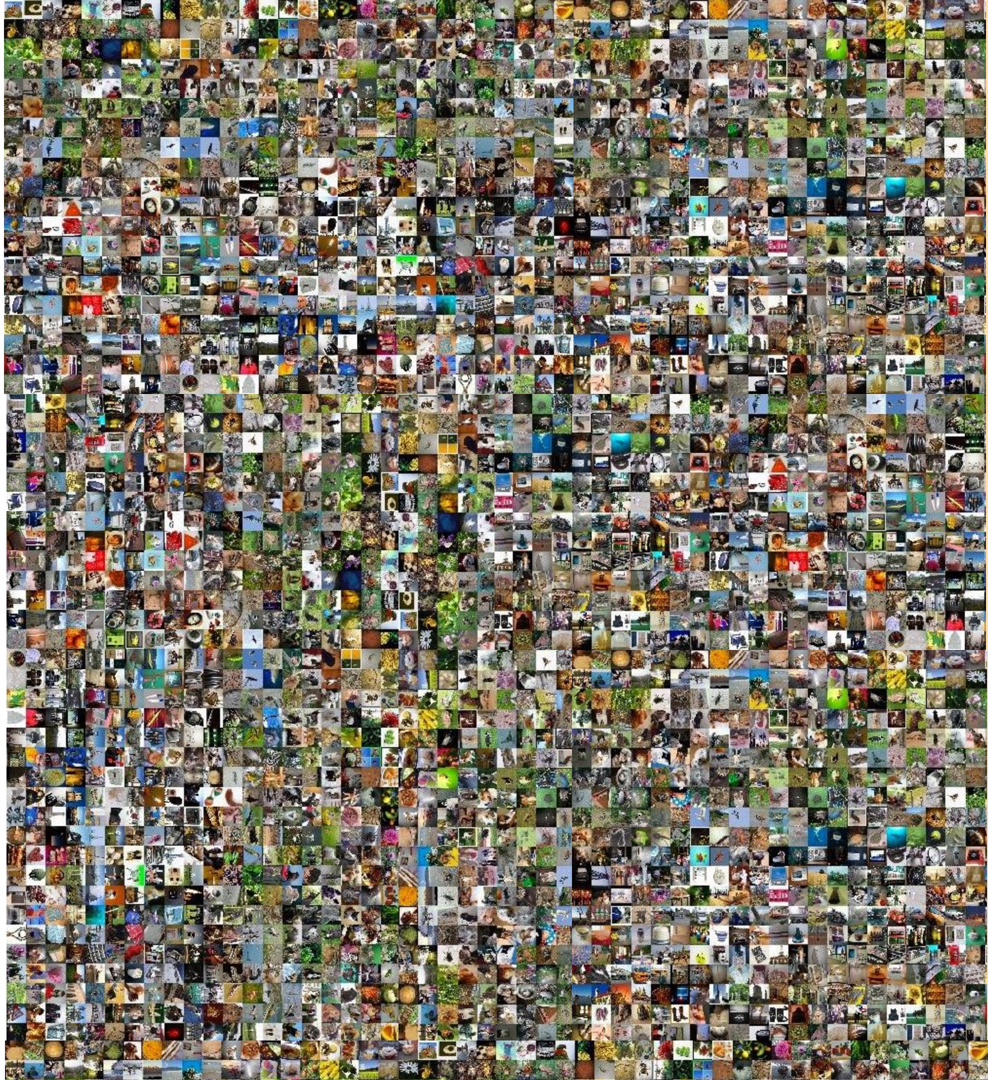
Data drives research

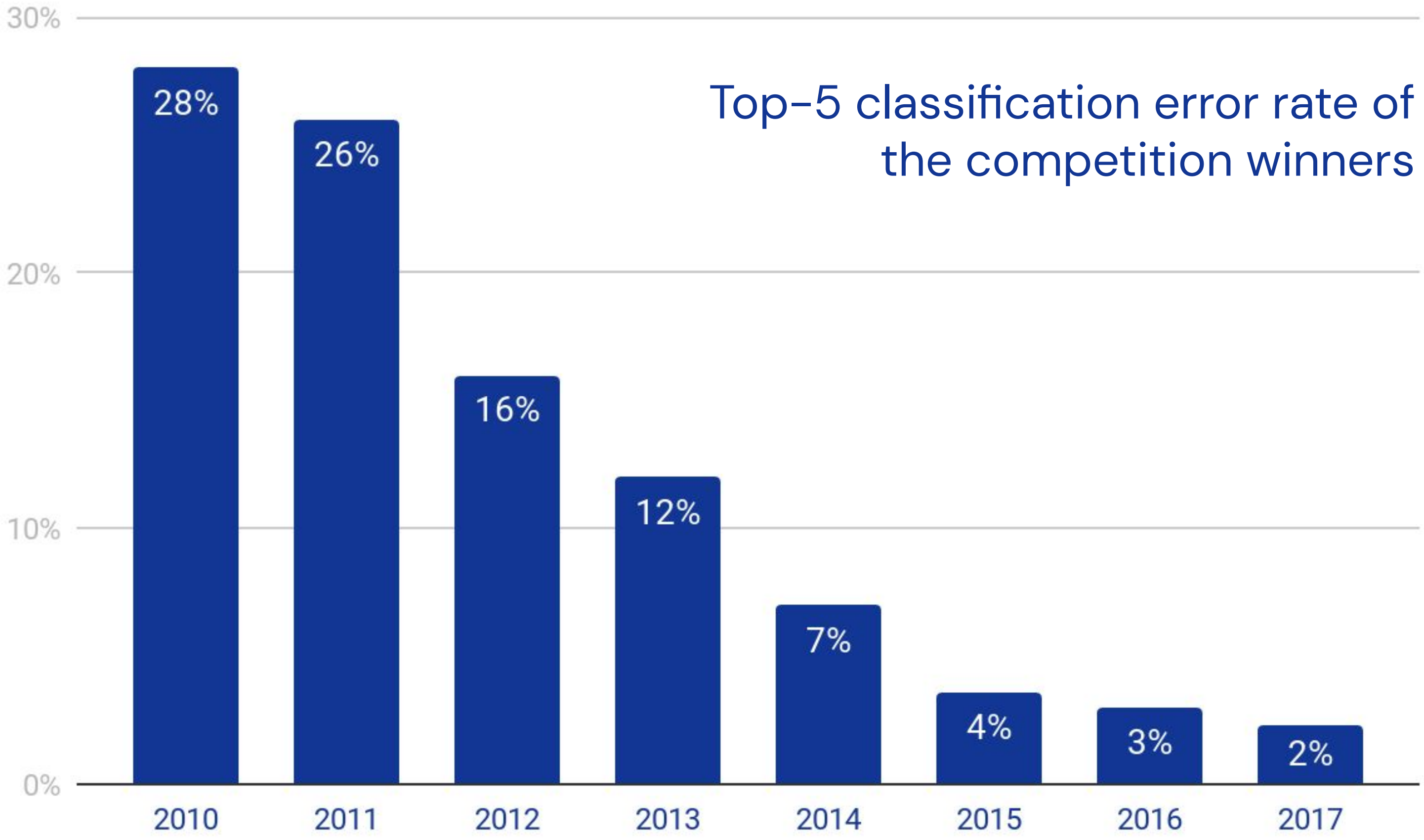
The ImageNet challenge

- Major computer vision benchmark
- Ran from 2010 to 2017
- 1.4M images, 1000 classes
- Image classification

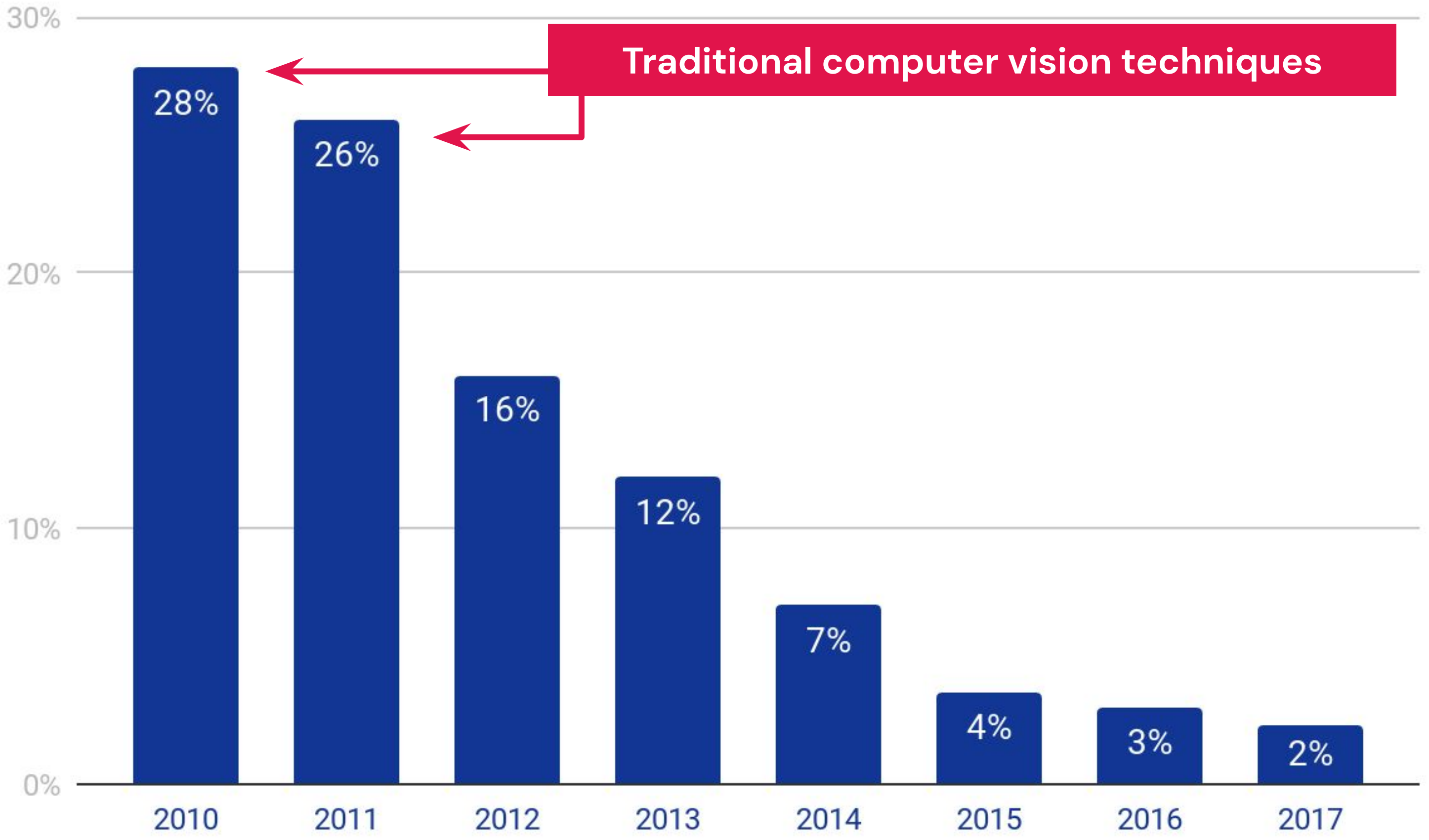
Want to learn more?

Russakovsky, Olga et al. ***ImageNet Large Scale Visual Recognition Challenge*** International Journal of Computer Vision 115.3 (**2015**)

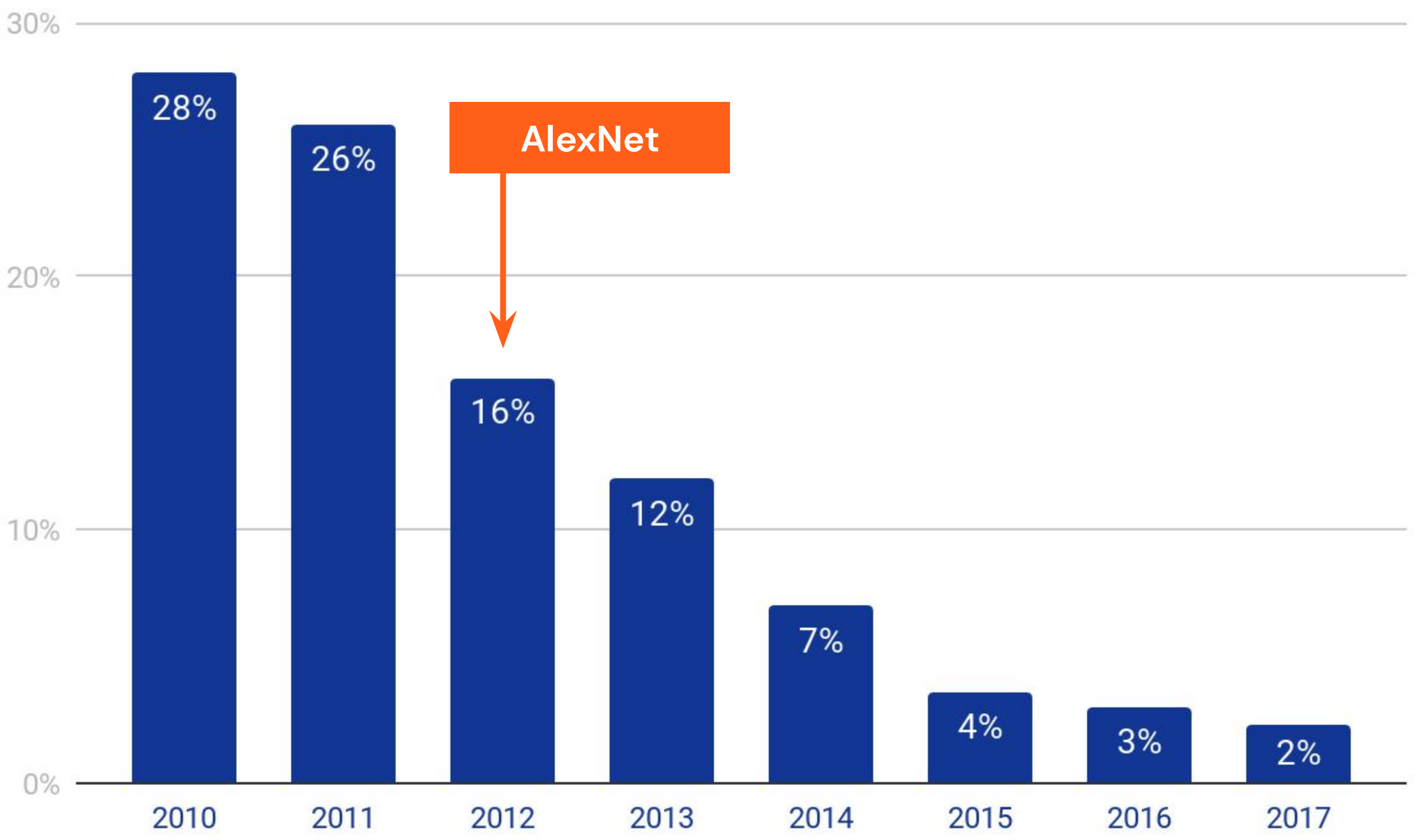


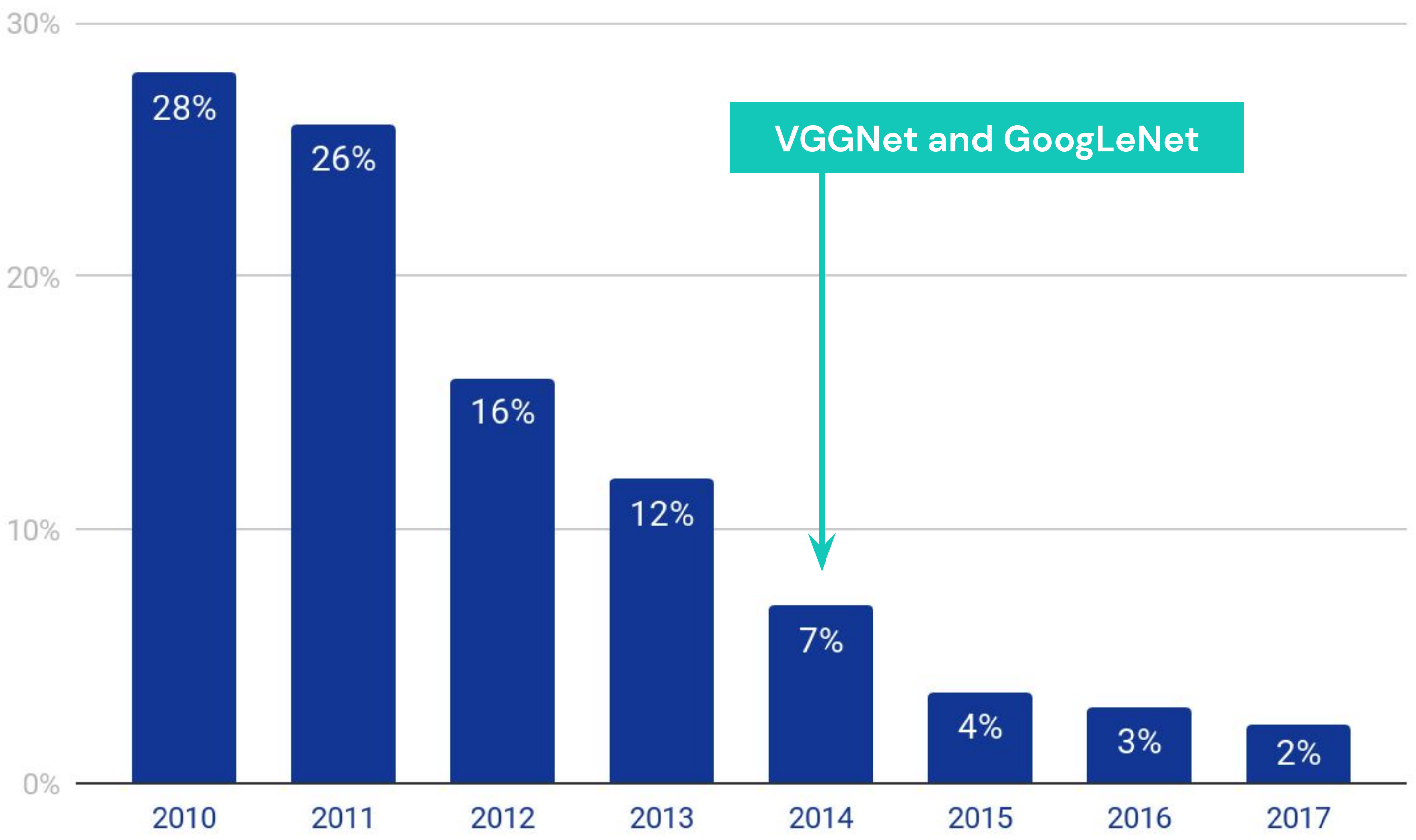


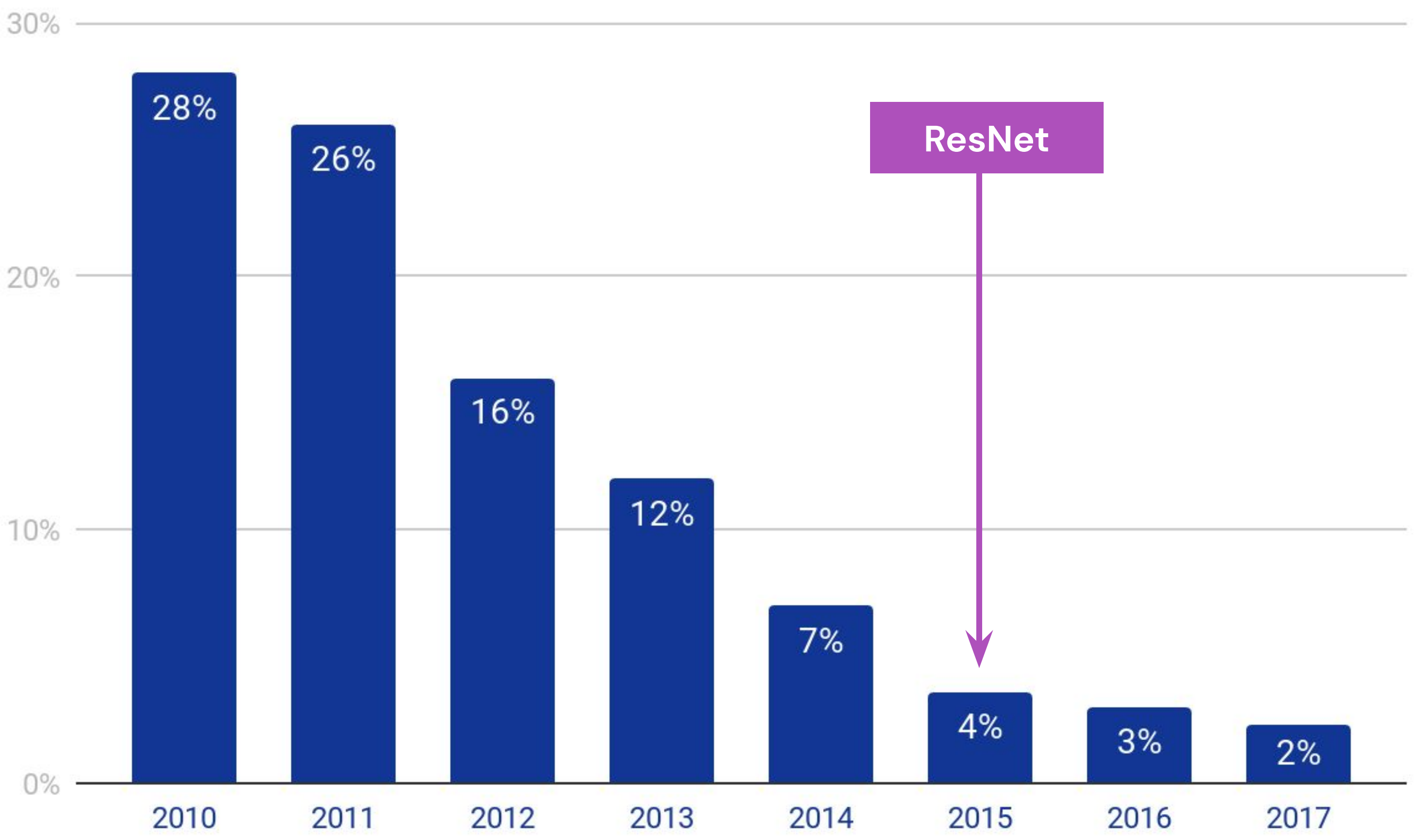
Top-5 classification error rate of
the competition winners
30%
20%
10%
0%
28%
26%
16%
12%
7%
4%
3%
2%
2010
2011
2012
2013
2014
2015
2016
2017



Traditional computer vision techniques
30%
20%
10%
0%
28%
26%
16%
12%
7%
4%
3%
2%
2010
2011
2012
2013
2014
2015
2016
2017


30%
20%
10%
0%
AlexNet
28%
26%
16%
12%
7%
4%
3%
2%
2010
2011
2012
2013
2014
2015
2016
2017

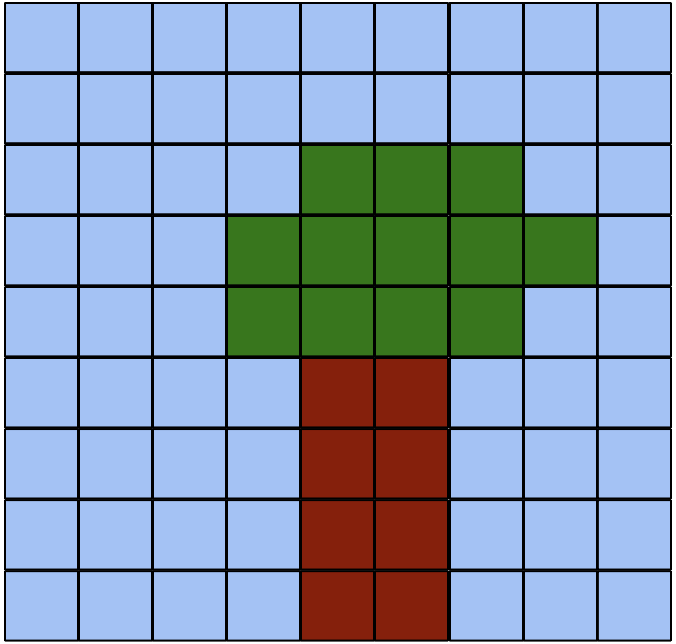

30%
20%
10%
0%
28%
26%
16%
12%
7%
4%
3%
2%
VGGNet and GoogLeNet
2010
2011
2012
2013
2014
2015
2016
2017


30%
20%
10%
0%
28%
26%
16%
12%
7%
4%
3%
2%
ResNet
2010
2011
2012
2013
2014
2015
2016
2017

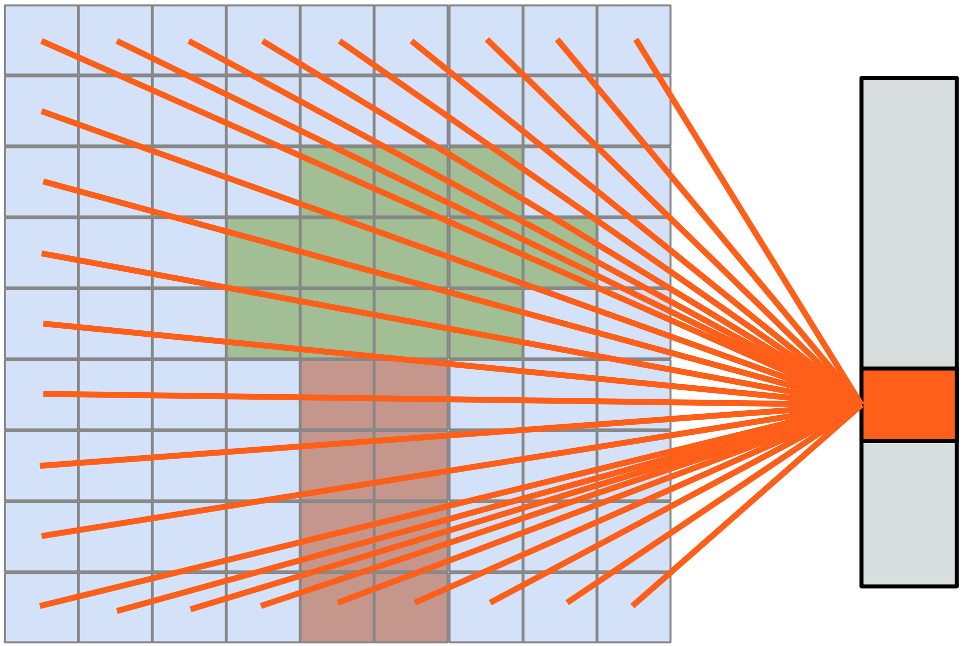
2 Building blocks

From fully connected to locally connected

From fully connected to locally connected



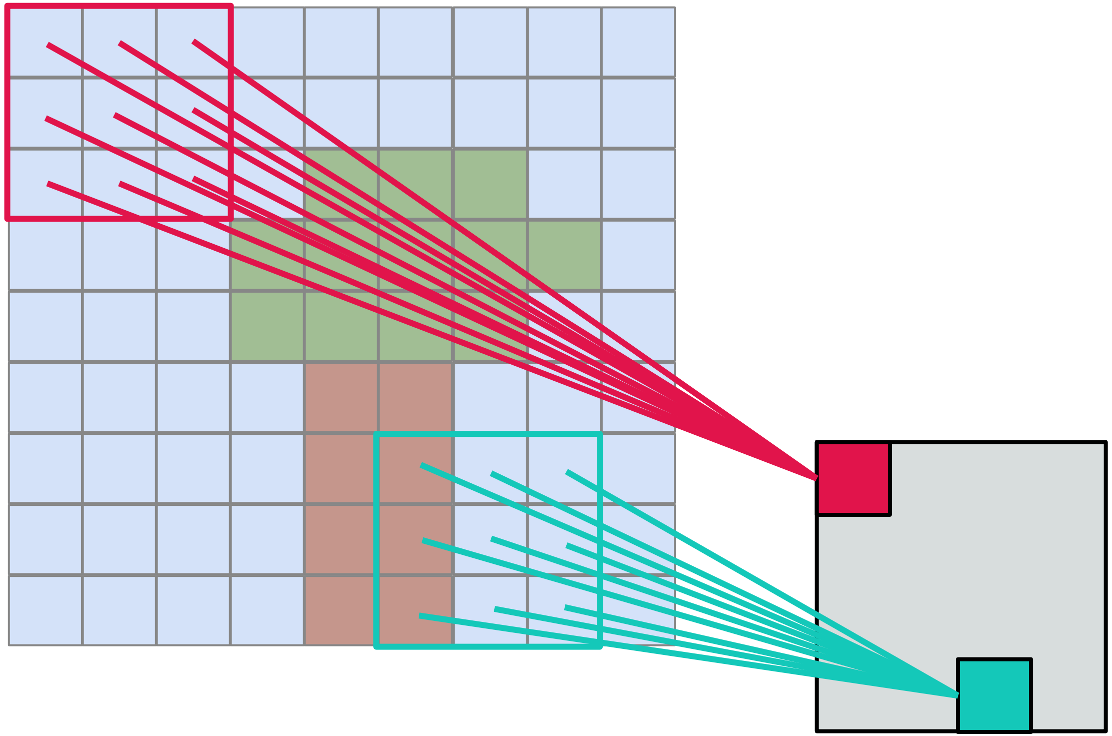
From fully connected to locally connected



fully-connected unit

$$y = \sum_{i \in \text{image}} \mathbf{w}_i \mathbf{x}_i + b$$

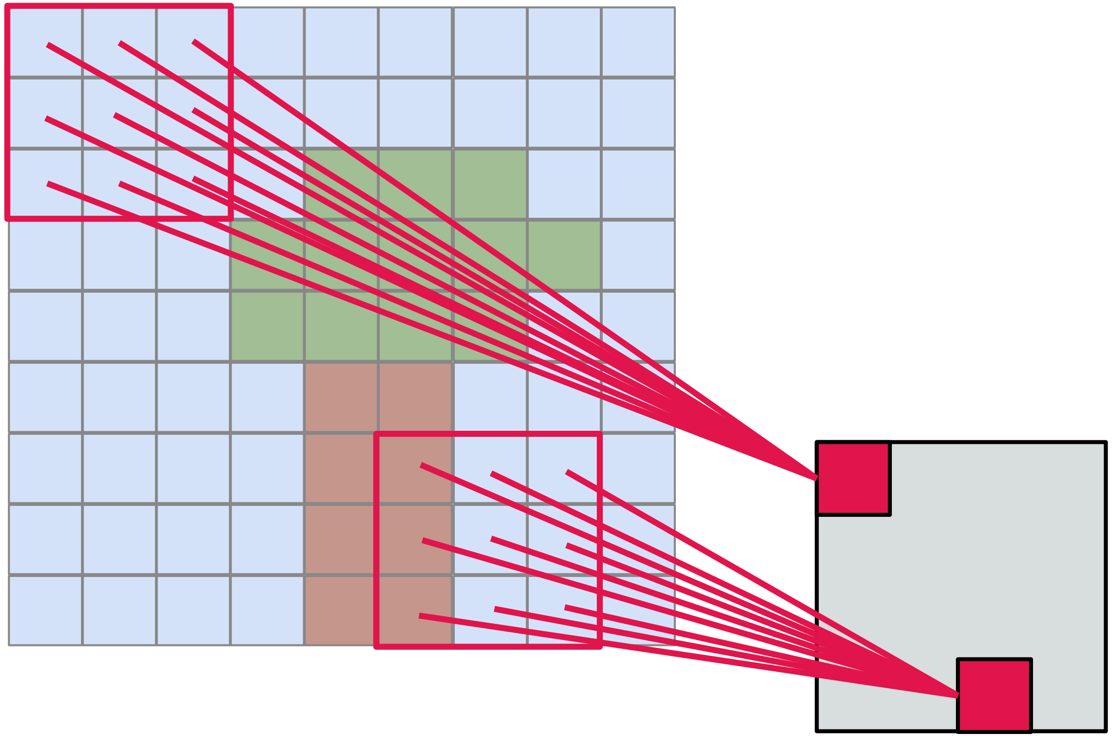
From fully connected to locally connected



$$y = \sum_{i \in 3\times3} \mathbf{w}_i \mathbf{x}_i + b$$

locally-connected units
3×3 receptive field

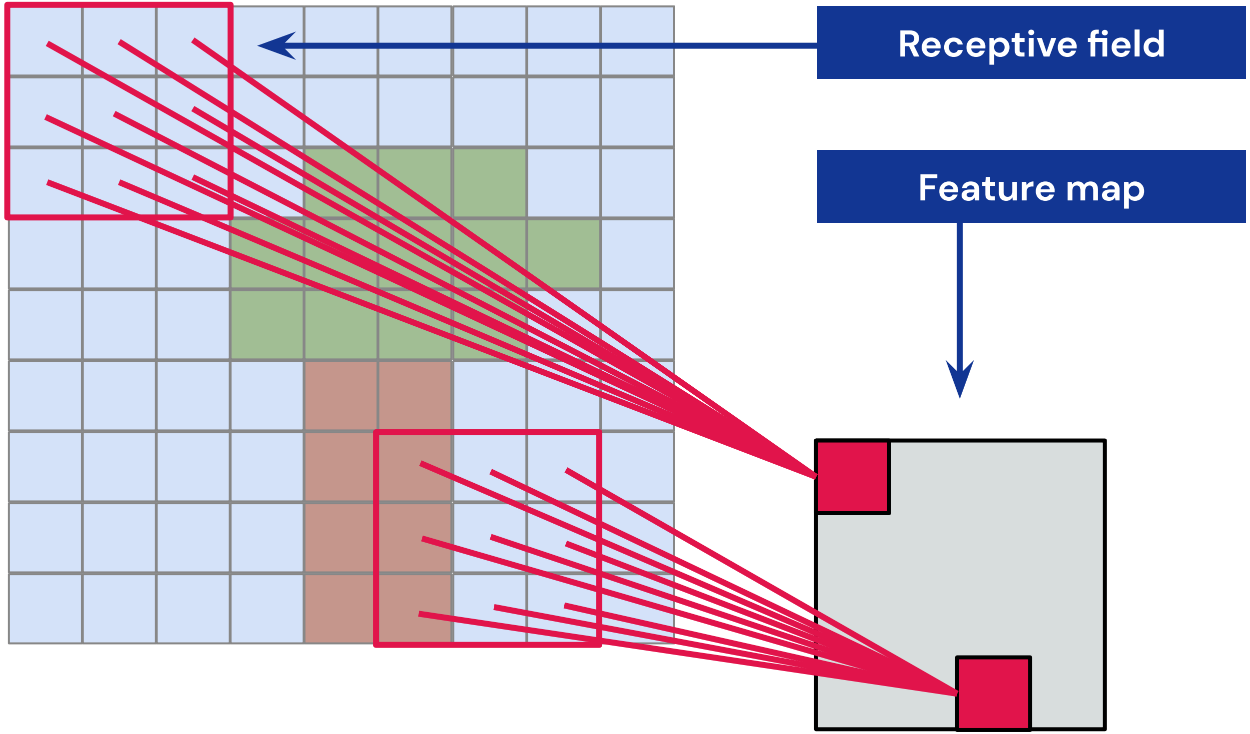
From locally connected to convolutional



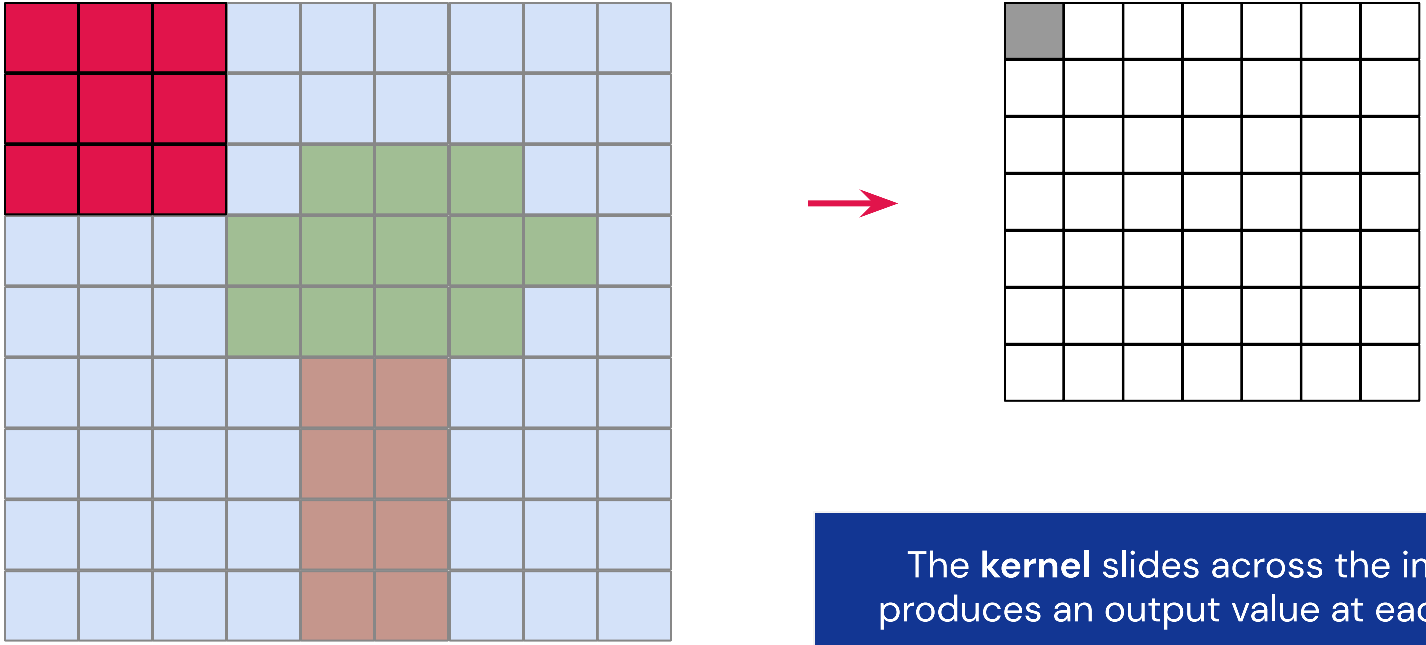
$$y = \mathbf{w} * \mathbf{x} + b$$

convolutional units
3×3 receptive field

From locally connected to convolutional

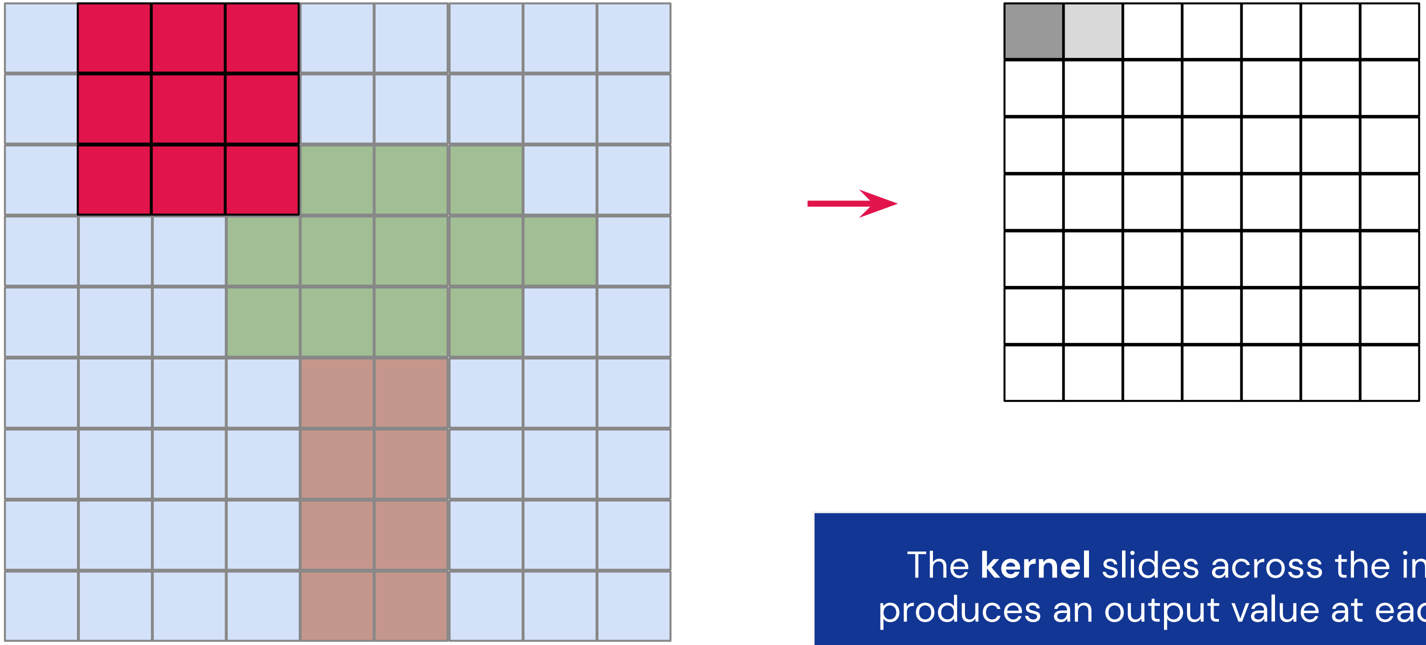


Implementation: the convolution operation



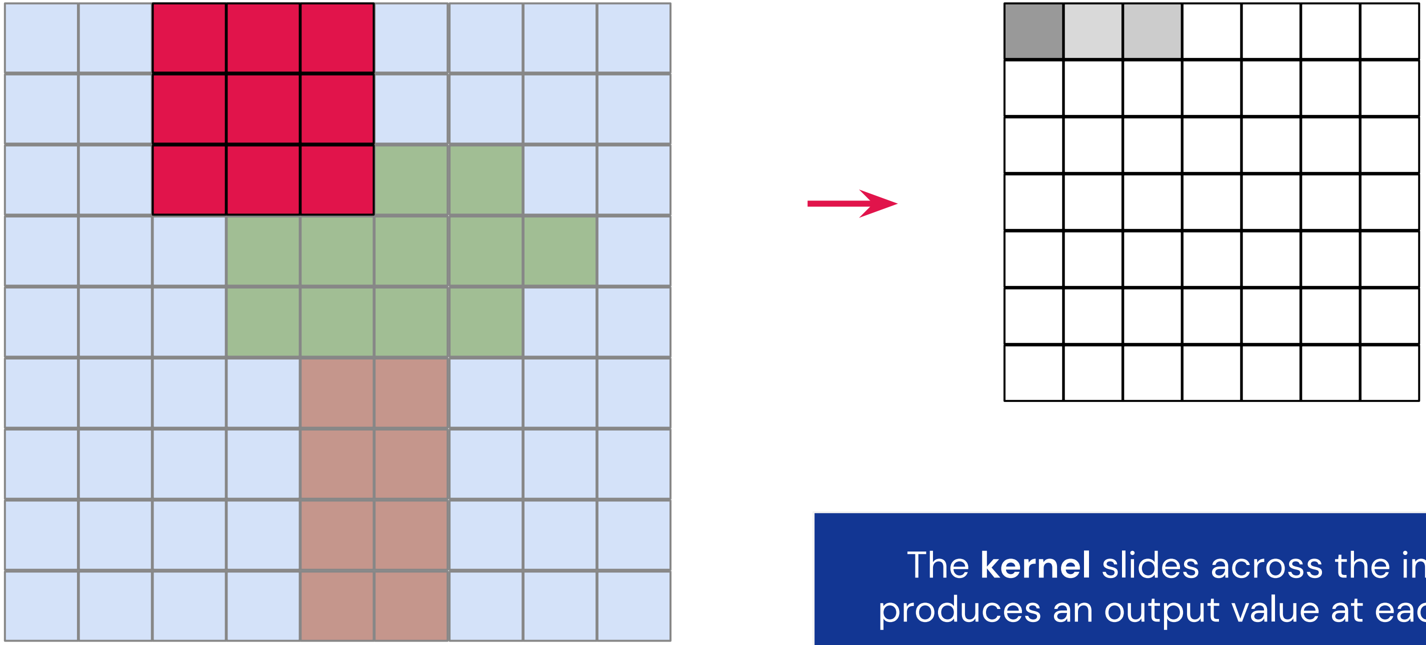
The **kernel** slides across the image and produces an output value at each position

Implementation: the convolution operation



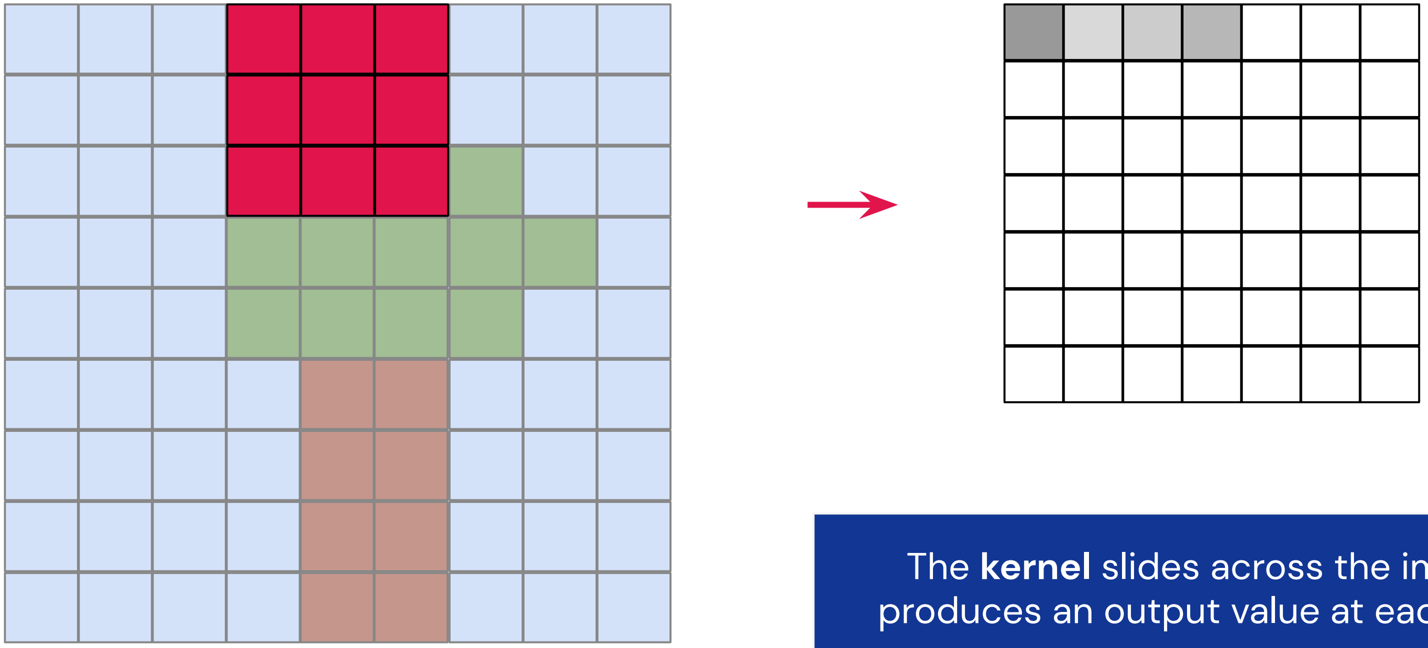
The **kernel** slides across the image and produces an output value at each position

Implementation: the convolution operation



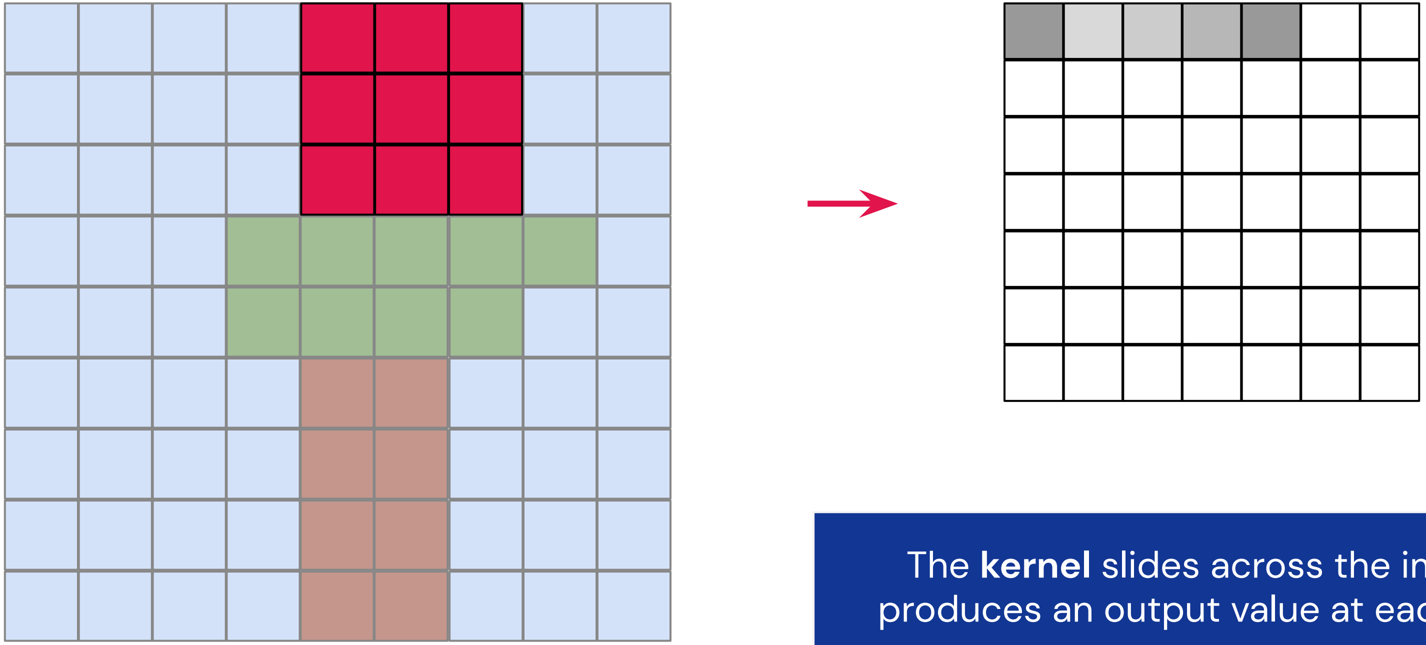
The **kernel** slides across the image and produces an output value at each position

Implementation: the convolution operation



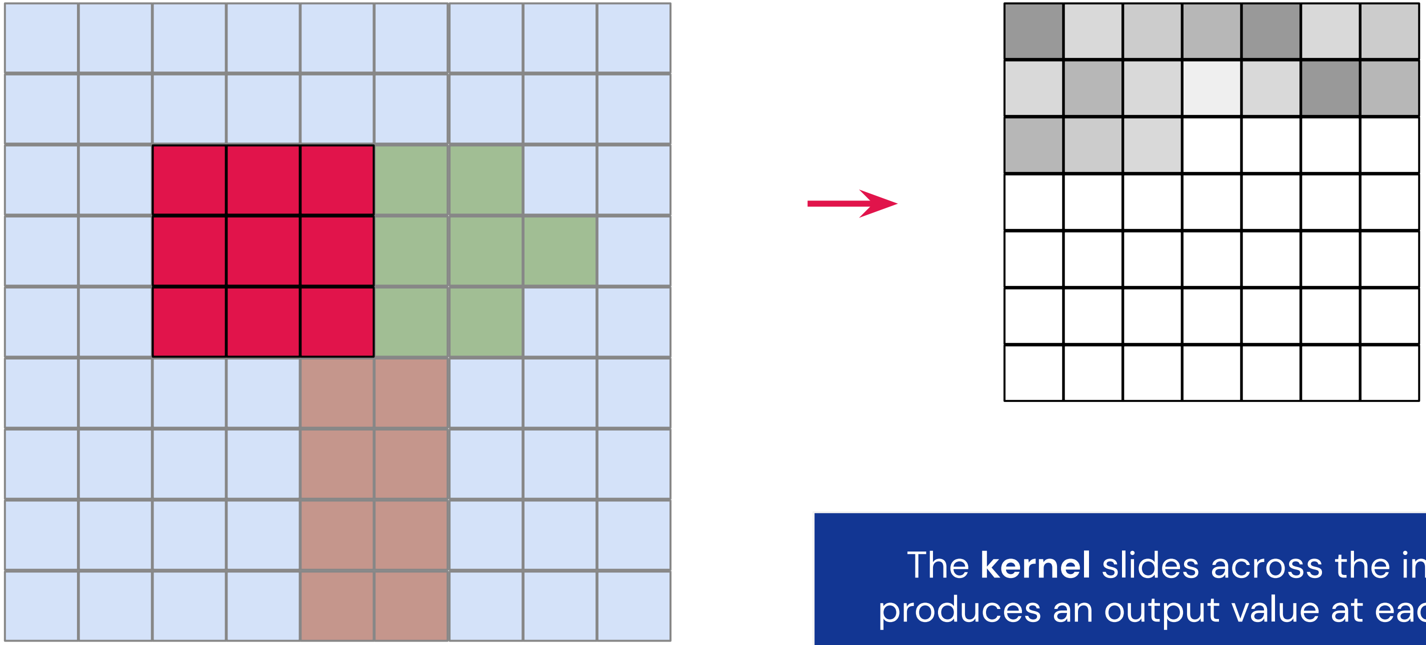
The **kernel** slides across the image and produces an output value at each position

Implementation: the convolution operation



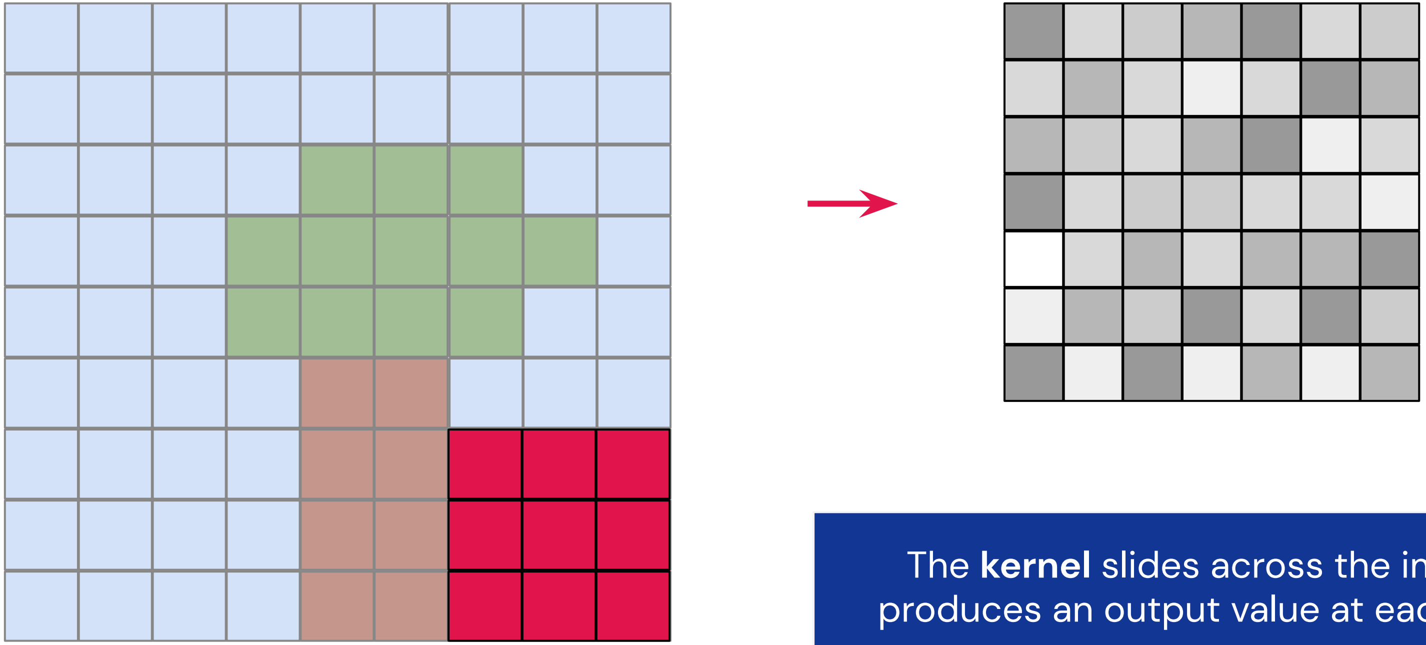
The **kernel** slides across the image and produces an output value at each position

Implementation: the convolution operation



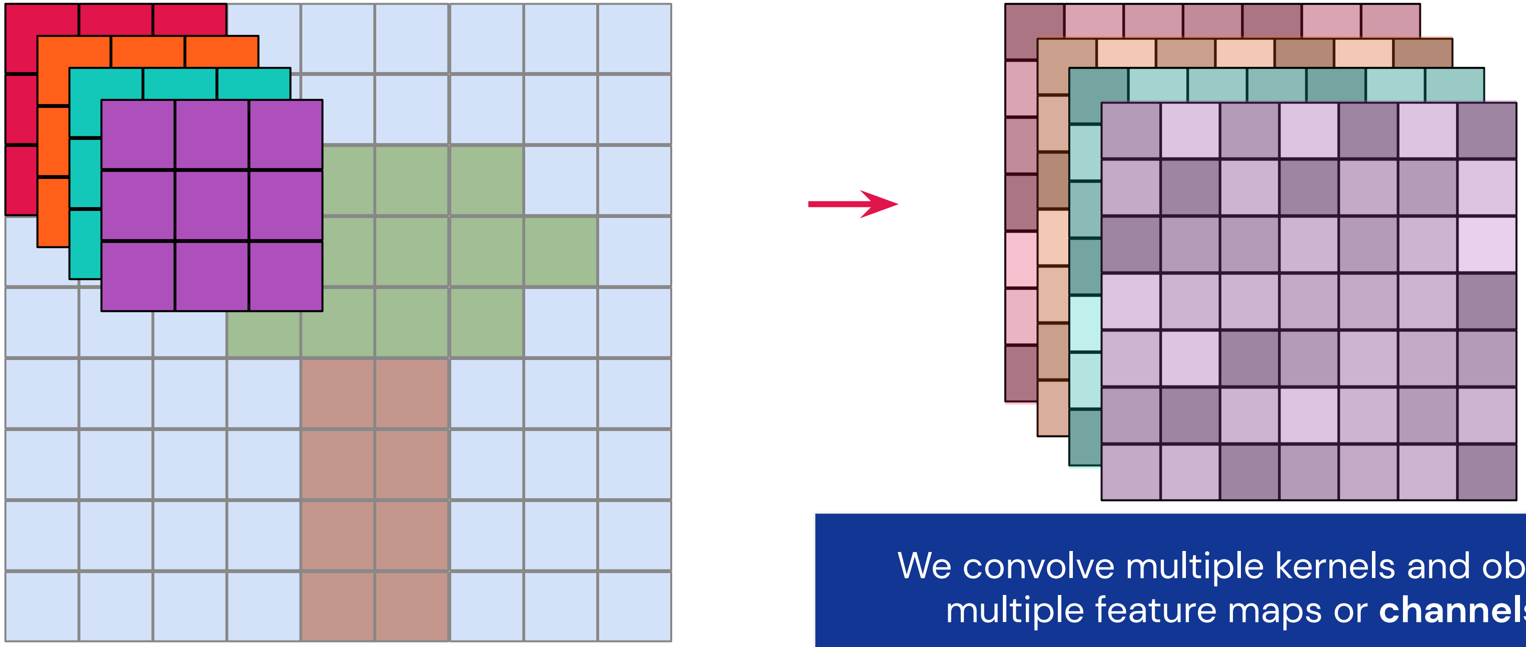
The **kernel** slides across the image and produces an output value at each position

Implementation: the convolution operation



The **kernel** slides across the image and produces an output value at each position

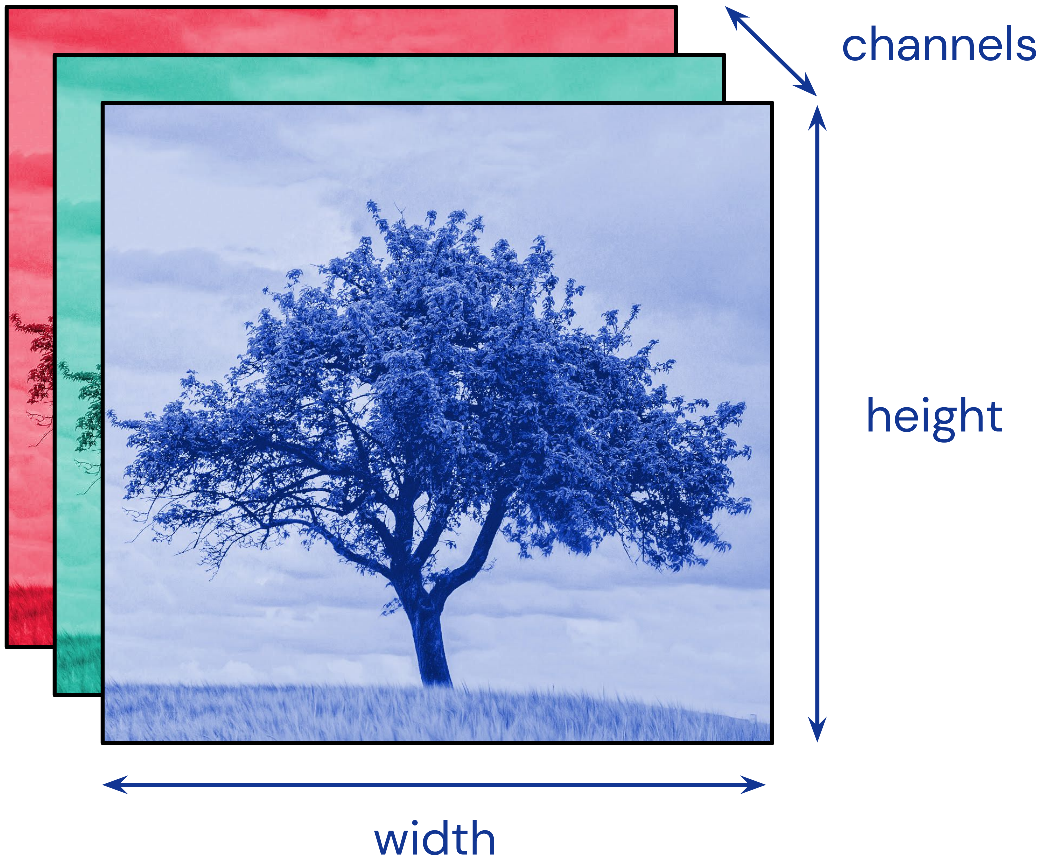
Implementation: the convolution operation



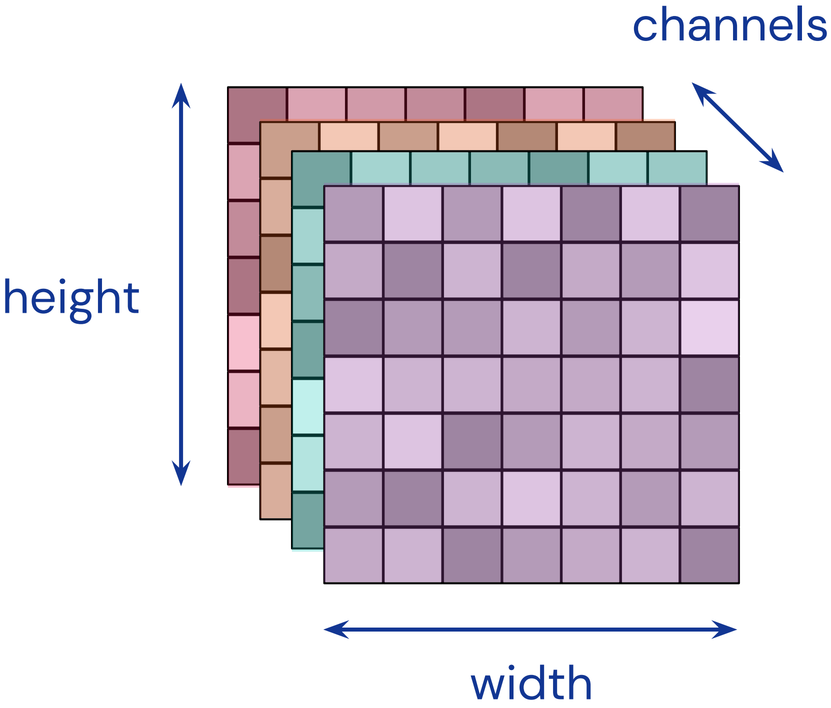
We convolve multiple kernels and obtain multiple feature maps or **channels**

Inputs and outputs are tensors

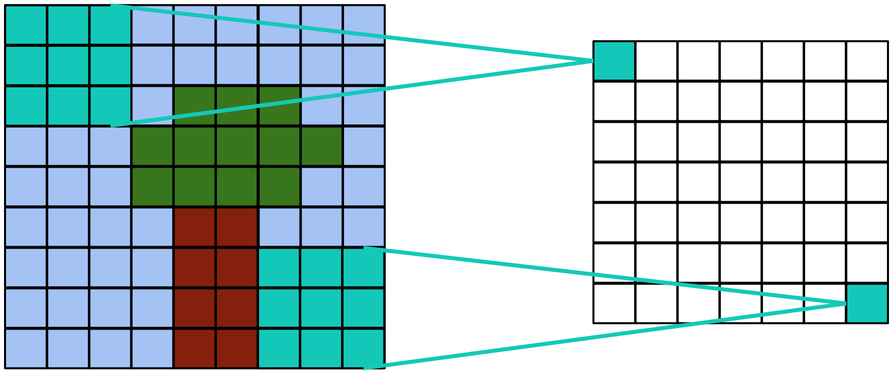
Inputs and outputs are tensors



Inputs and outputs are tensors

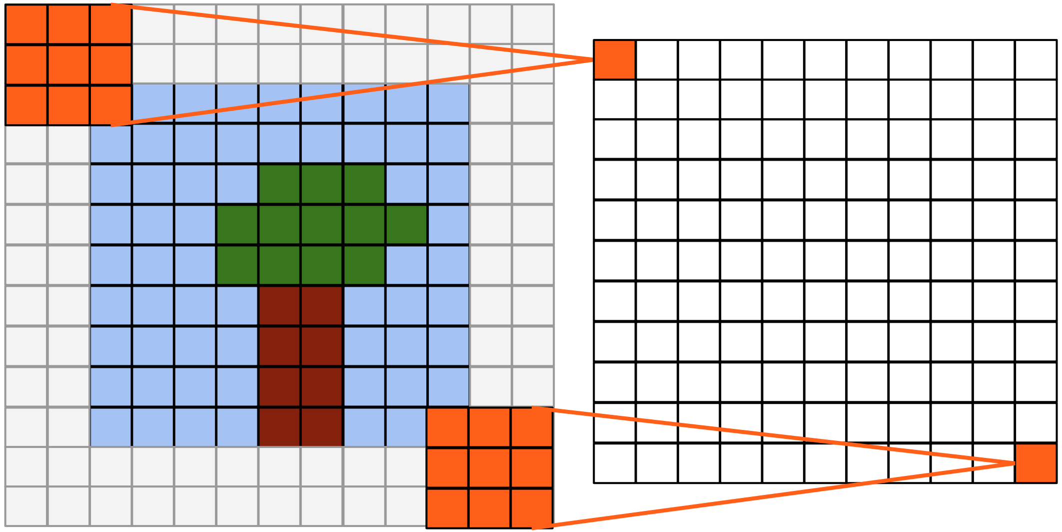


Variants of the convolution operation



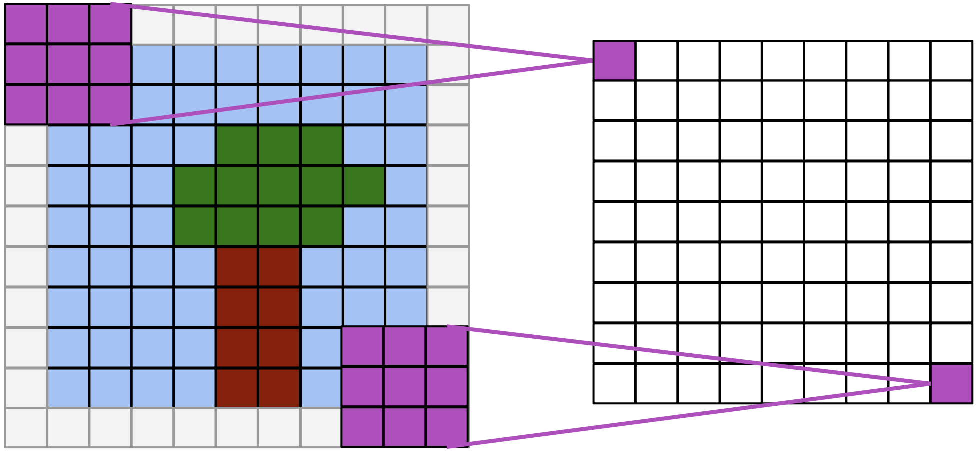
Valid convolution: output size = input size – kernel size + 1

Variants of the convolution operation



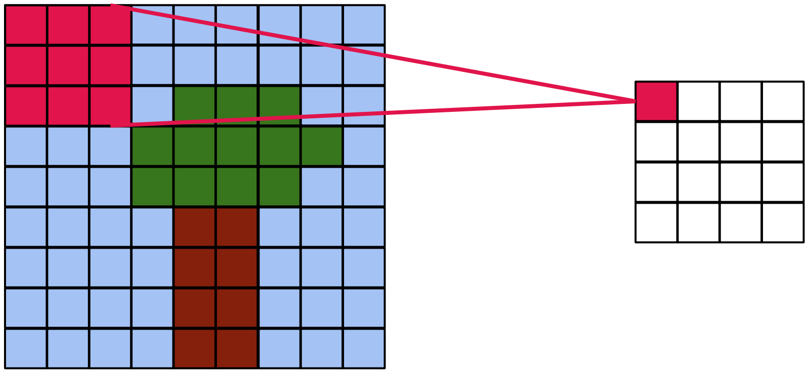
Full convolution: output size = input size + kernel size – 1

Variants of the convolution operation



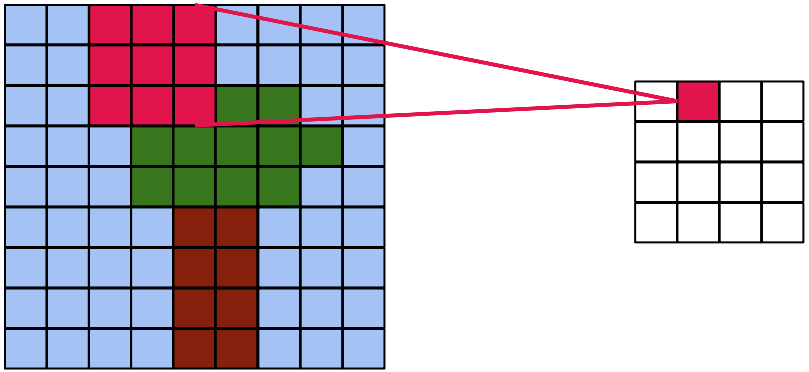
Same convolution: output size = input size

Variants of the convolution operation



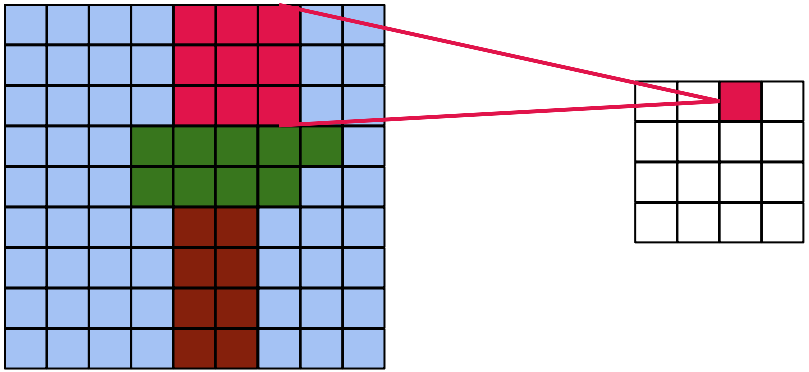
Strided convolution: kernel slides along the image with a step > 1

Variants of the convolution operation



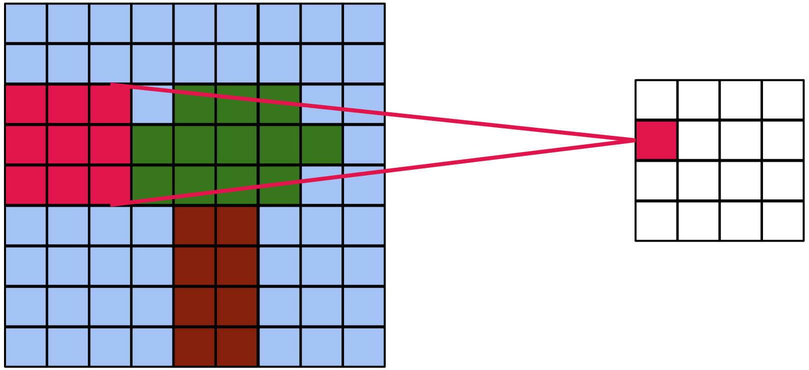
Strided convolution: kernel slides along the image with a step > 1

Variants of the convolution operation



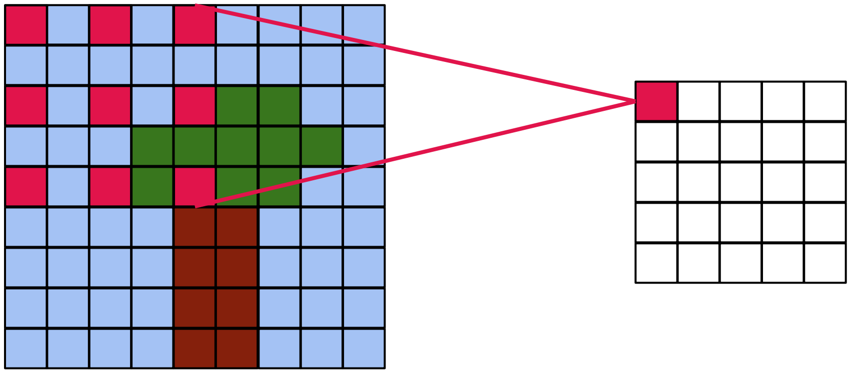
Strided convolution: kernel slides along the image with a step > 1

Variants of the convolution operation



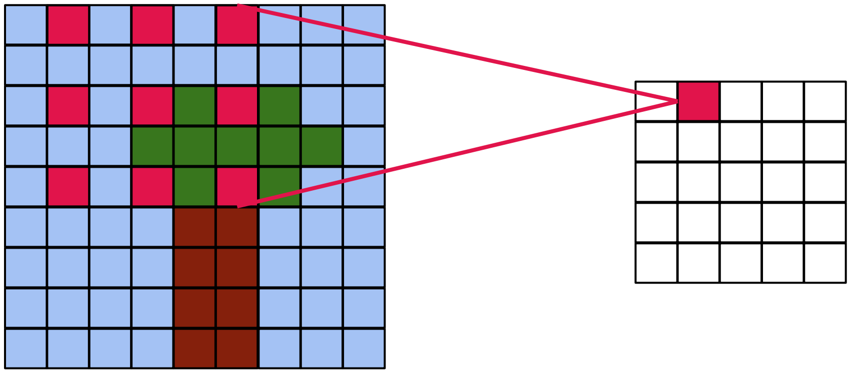
Strided convolution: kernel slides along the image with a step > 1

Variants of the convolution operation



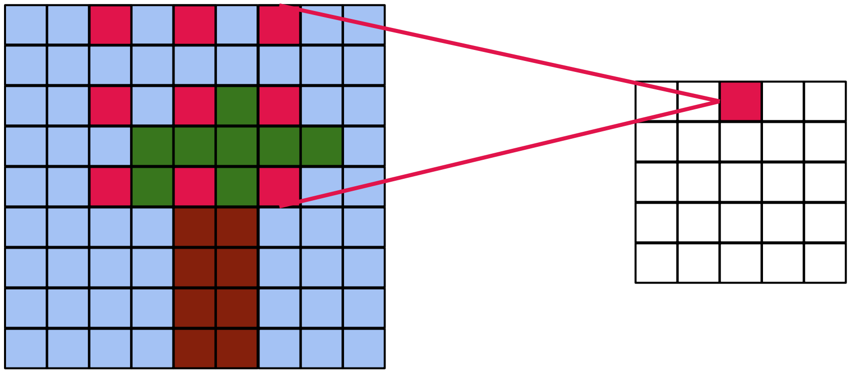
Dilated convolution: kernel is spread out, step > 1 between kernel elements

Variants of the convolution operation



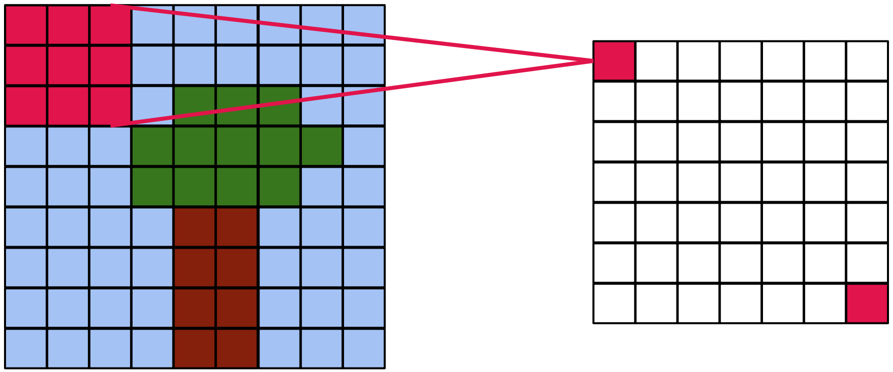
Dilated convolution: kernel is spread out, step > 1 between kernel elements

Variants of the convolution operation



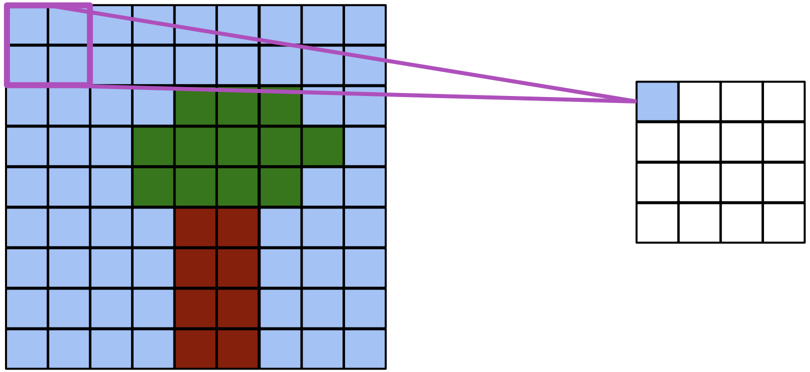
Dilated convolution: kernel is spread out, step > 1 between kernel elements

Variants of the convolution operation



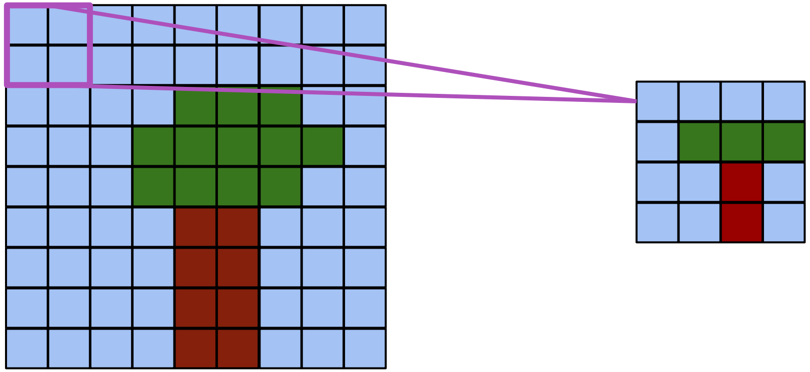
Depthwise convolution: each output channel is connected only to one input channel

Pooling



Pooling: compute mean or max over small windows to reduce resolution

Pooling



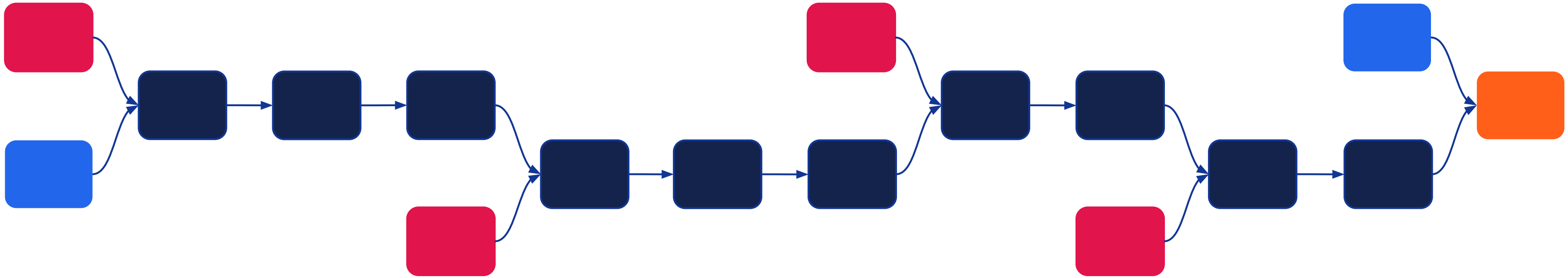
Pooling: compute mean or max over small windows to reduce resolution

3 Convolutional neural networks

Stacking the building blocks

- CNNs or “convnets”
- Up to 100s of layers
- Alternate convolutions and pooling to create a hierarchy

Recap: neural networks as computational graphs



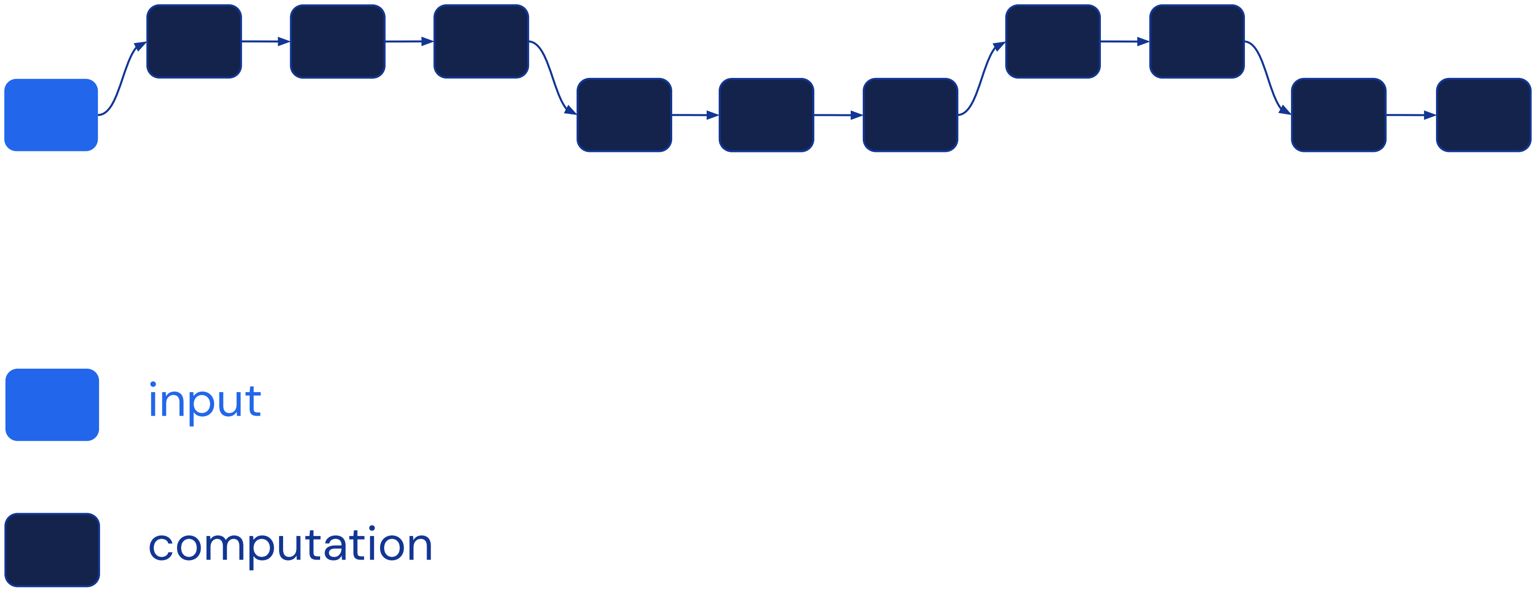
input

loss

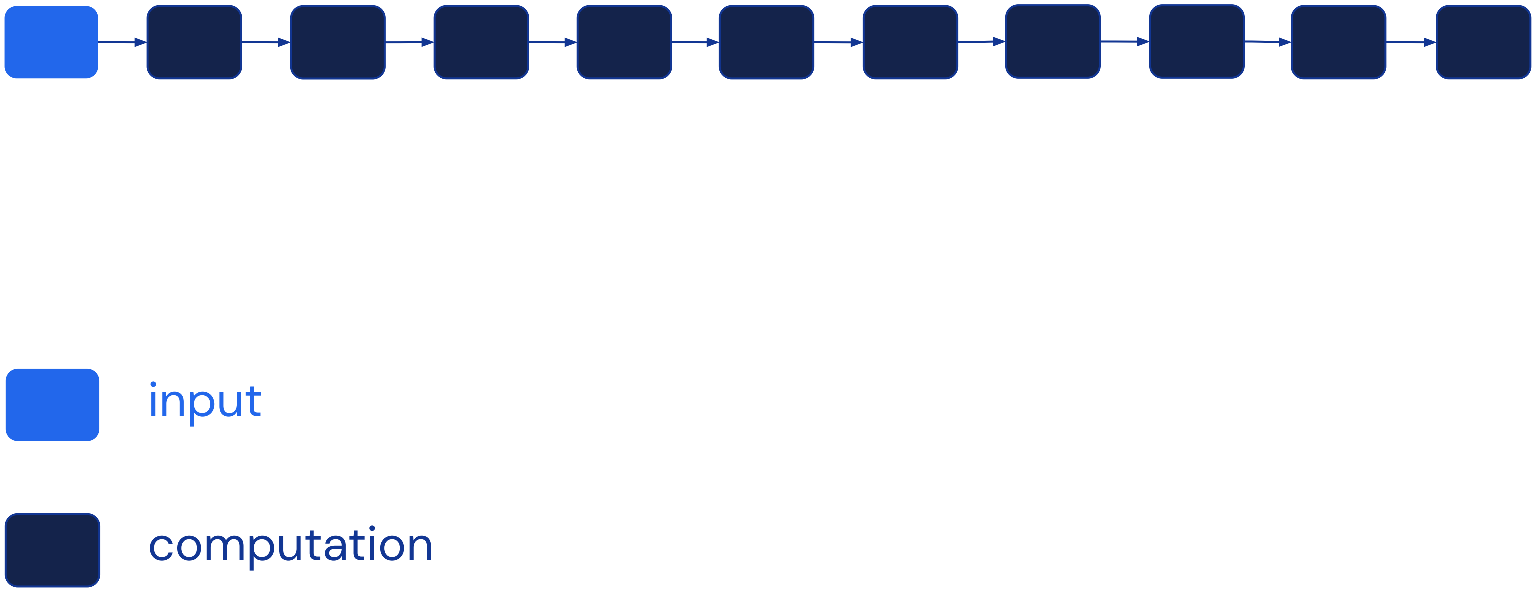
computation

parameters

Simplified diagram: implicit parameters and loss



Simplified diagram: implicit parameters and loss



Computational building blocks of convnets

fully connected

input

convolution

nonlinearity

pooling

4 Going deeper: Case studies

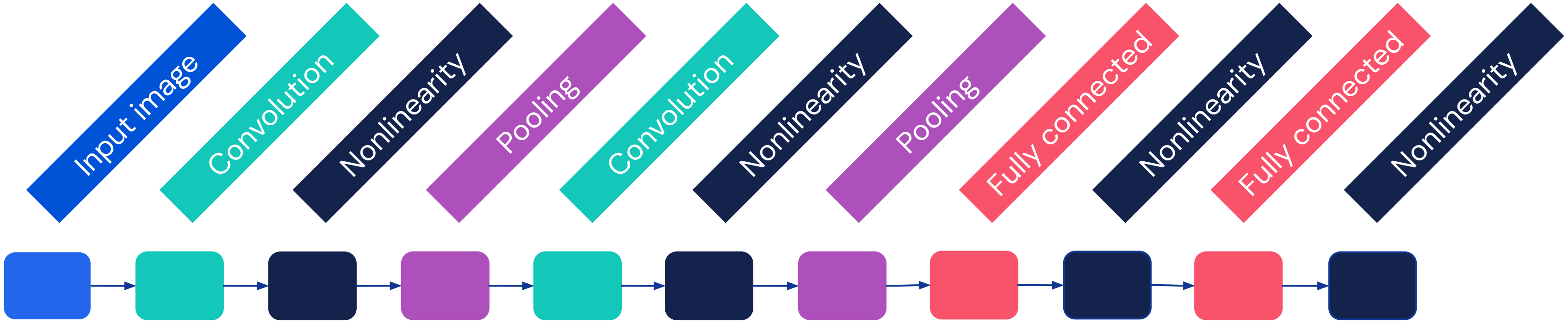
LeNet-5 (1998)

Architecture of **LeNet-5**, a convnet for handwritten digit recognition

Want to learn more?

Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.
Gradient-based learning applied to document recognition
Proceedings of the IEEE 86(11) (**1998**)

LeNet-5 (1998)



Input image
Convolution
Nonlinearity
Pooling
Convolution
Nonlinearity
Pooling
Fully connected
Nonlinearity
Fully connected
Nonlinearity

AlexNet (2012)

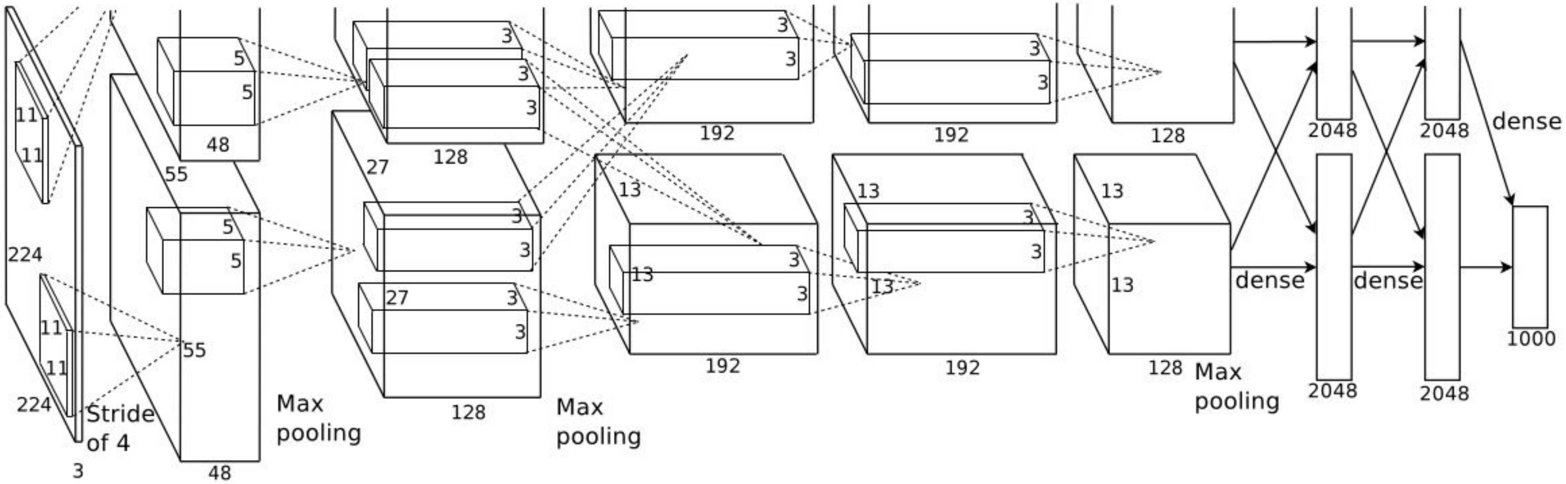


Figure from Krizhevsky et al. (2012)

Want to learn more?

Krizhevsky, A.; Sutskever, I.; Hinton, G.E.
ImageNet classification with deep convolutional neural networks
Neural Information Processing Systems (**2012**)

Architecture: 8 layers, ReLU, dropout, weight decay

Infrastructure: large dataset, trained 6 days on 2 GPUs

AlexNet (2012)

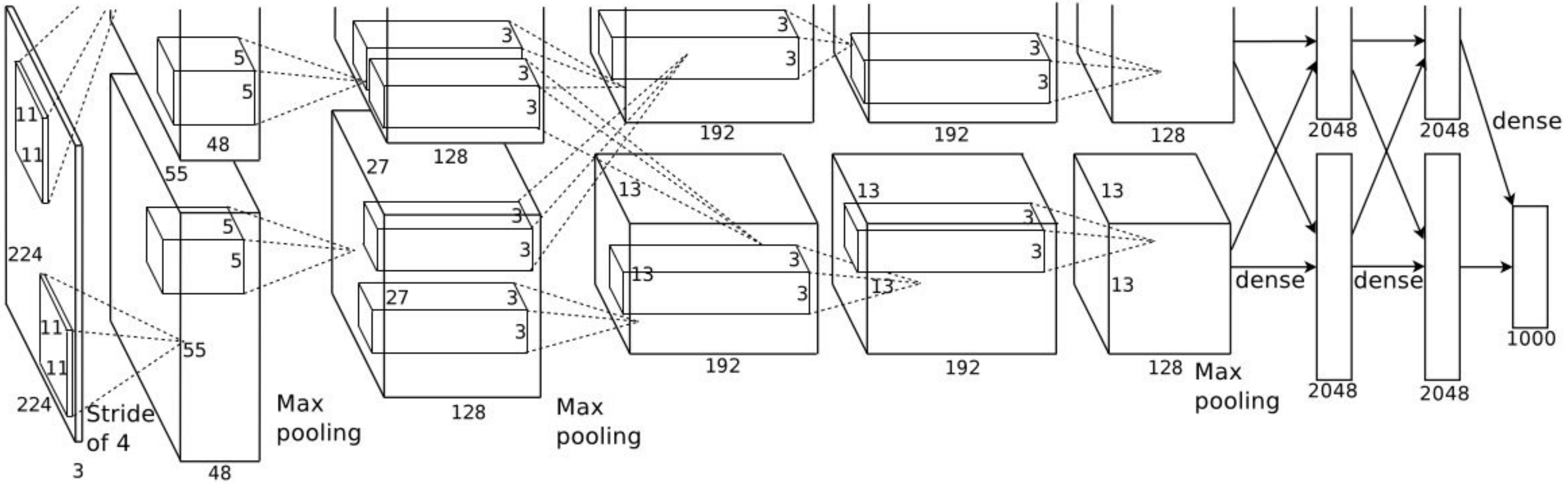
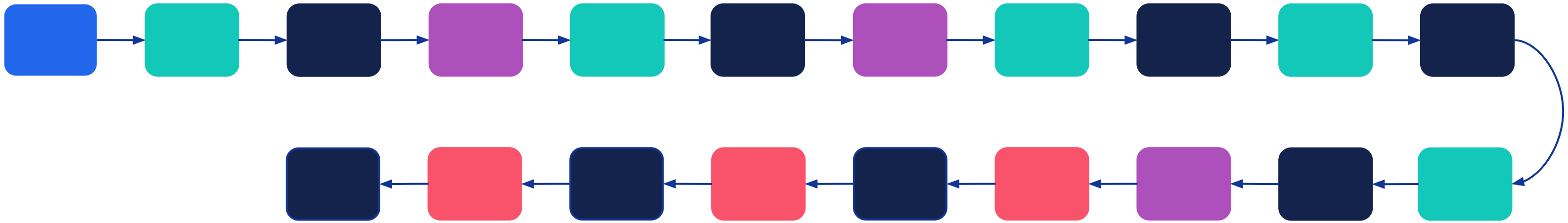
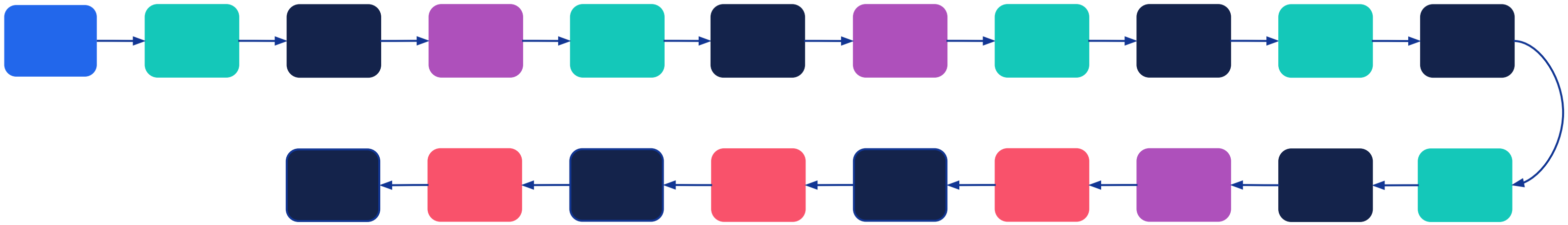


Figure from Krizhevsky et al. (2012)



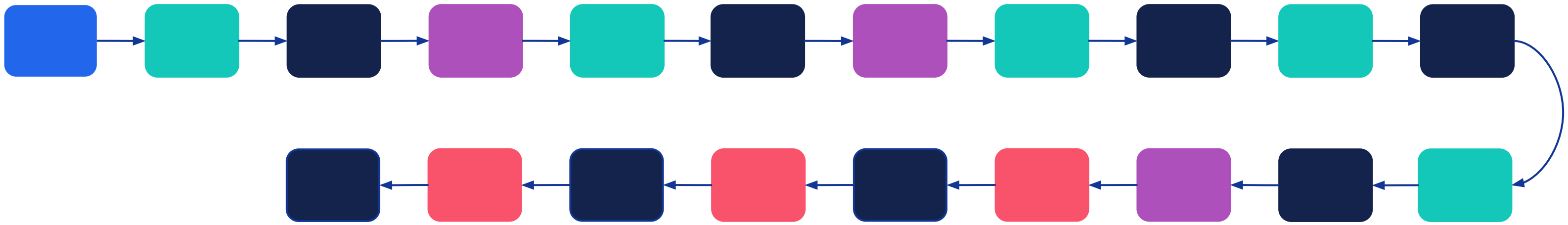
AlexNet (2012)

Input image:
$\longrightarrow$ 224×224×3

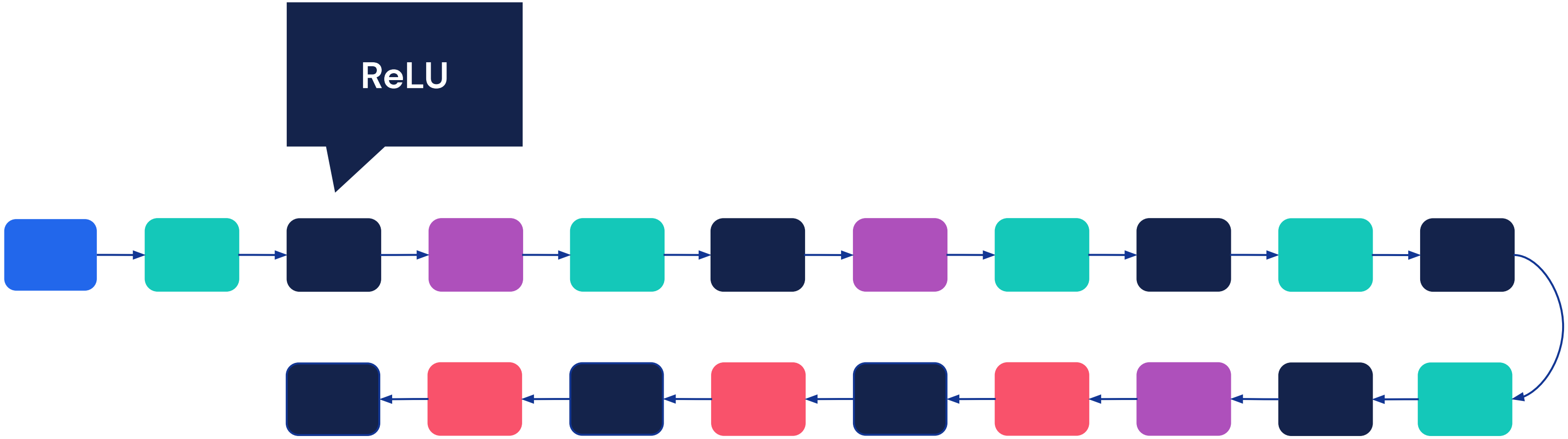


AlexNet (2012)

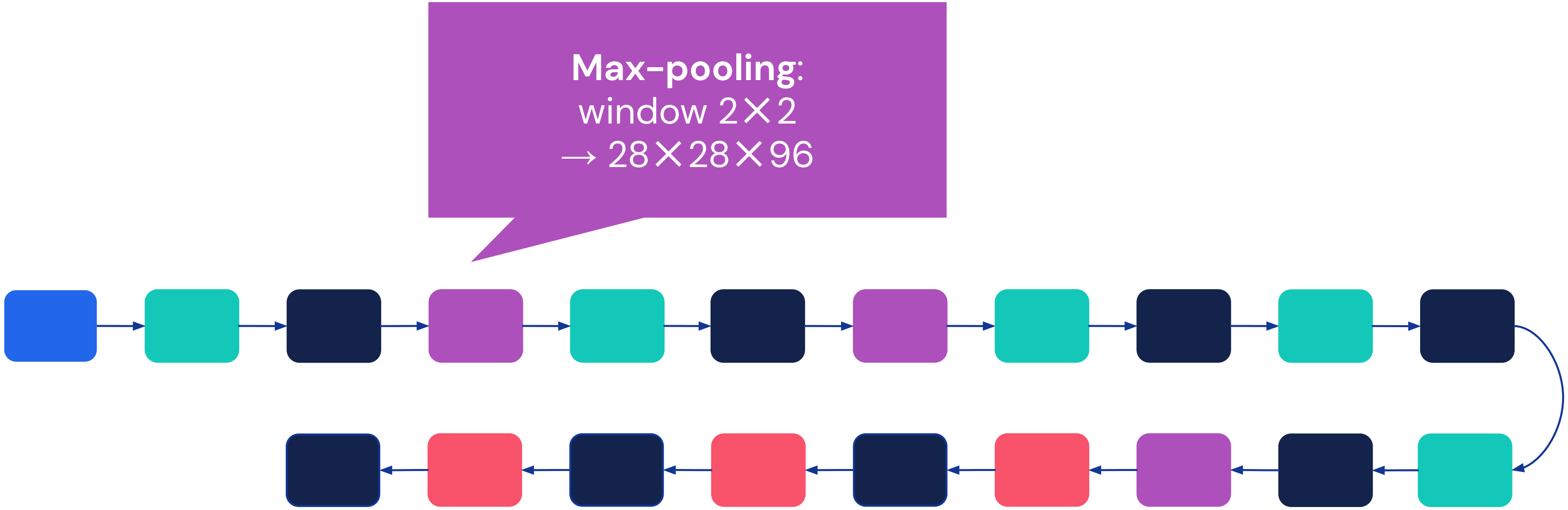
Layer 1 **convolution**:
kernel 11×11, 96 channels, stride 4
→ 56×56×96



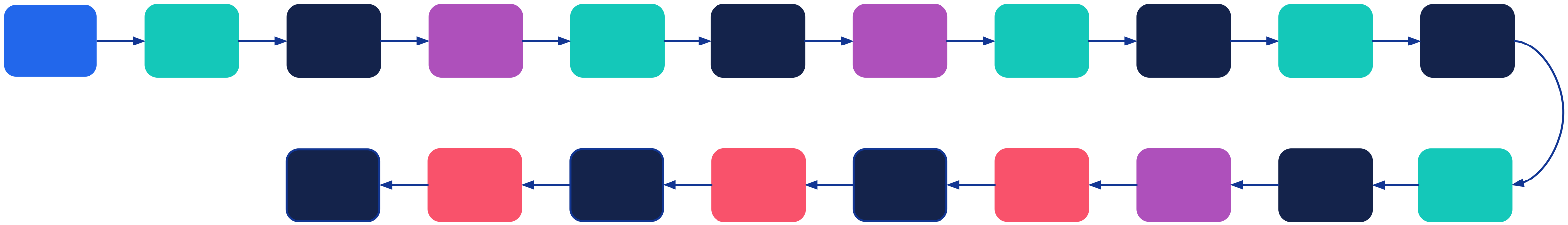
AlexNet (2012)



AlexNet (2012)

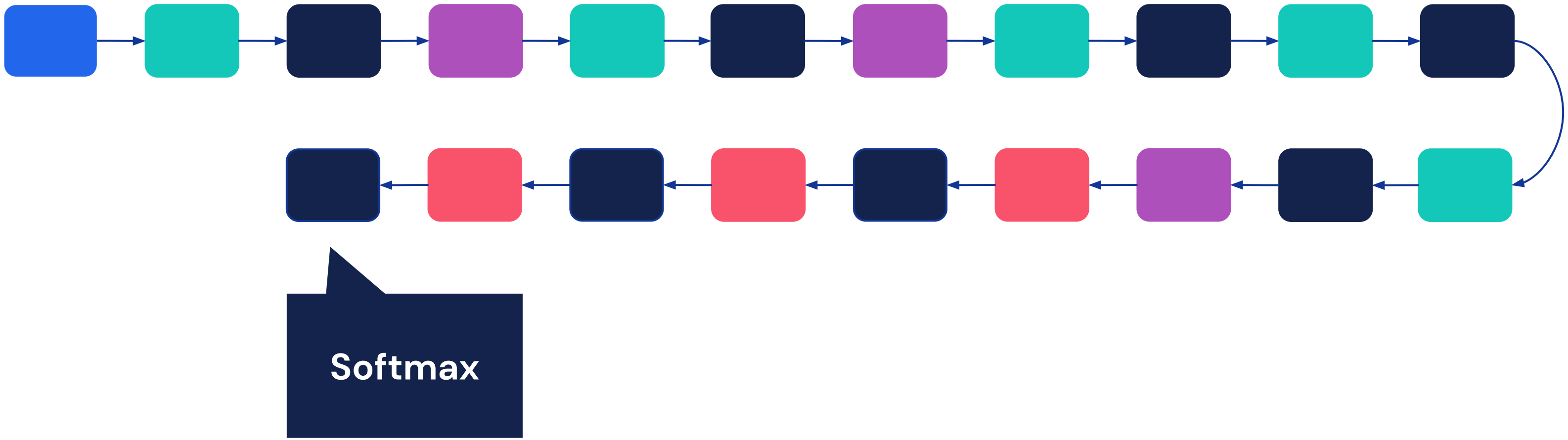


AlexNet (2012)

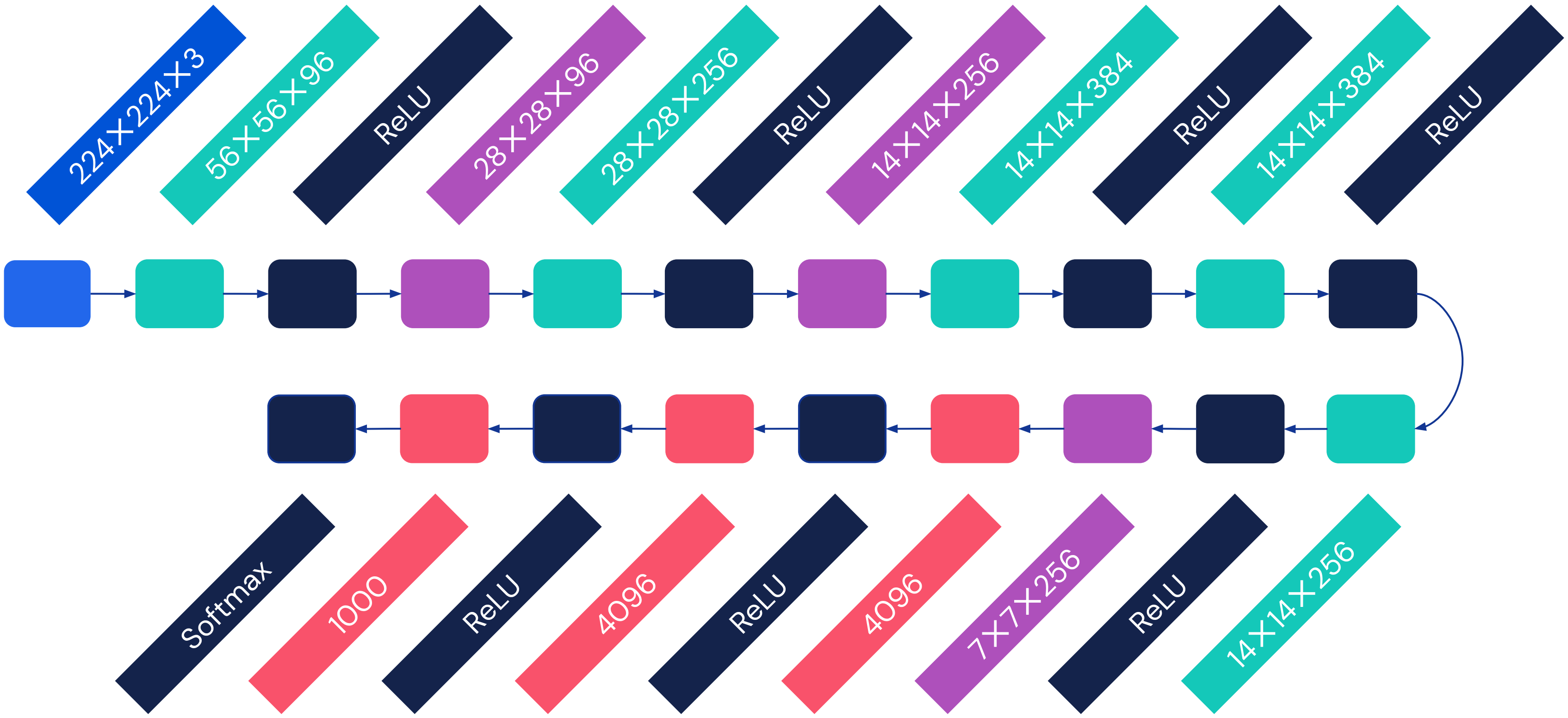


Layer 8 **fully connected:**
→ 1000

AlexNet (2012)

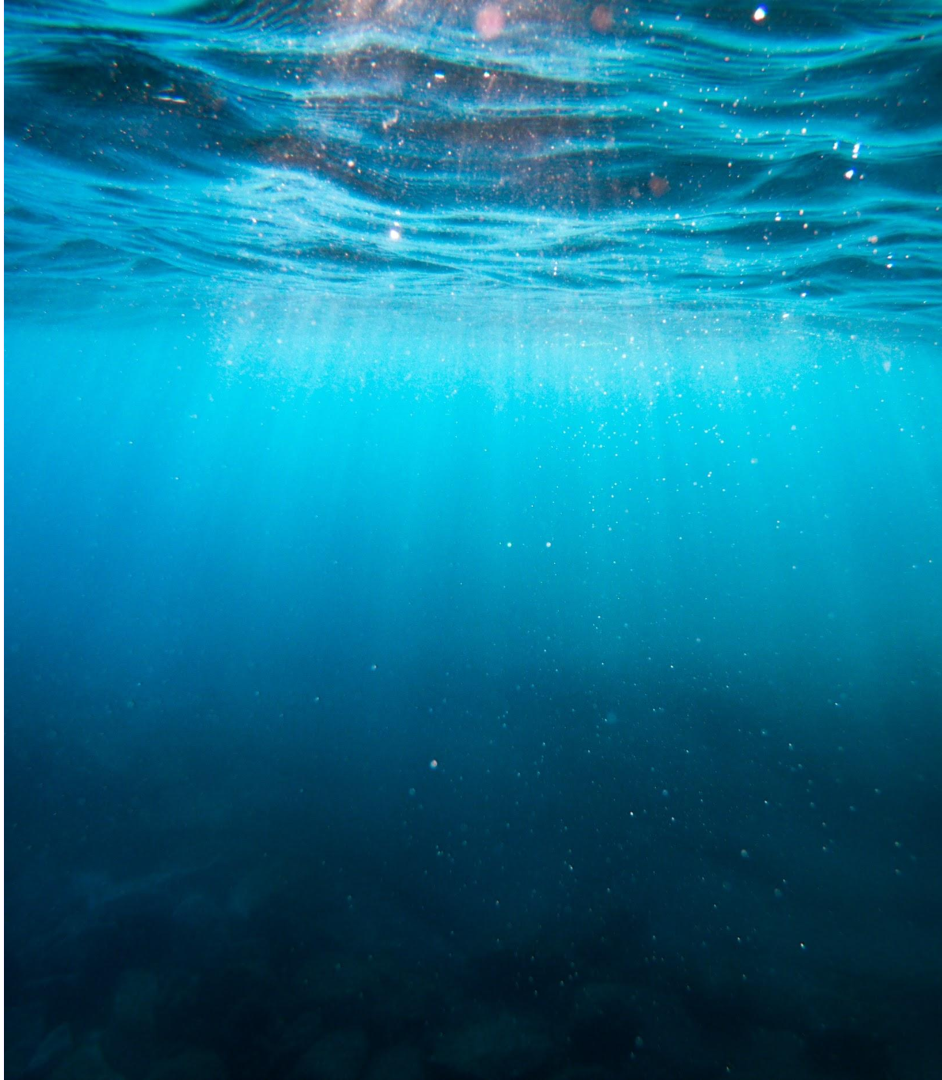


AlexNet (2012)

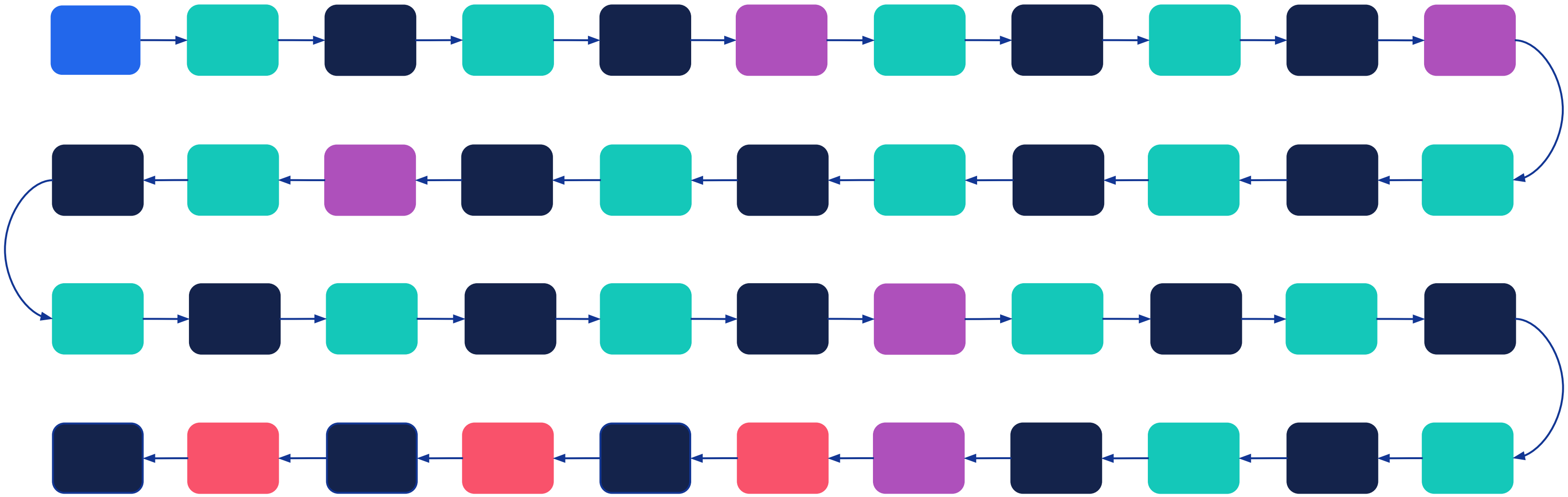


Deeper is better

- Each layer is a linear classifier by itself
- More layers – more nonlinearities
- What limits the number of layers in convnets?



VGGNet (2014): building very deep convnets

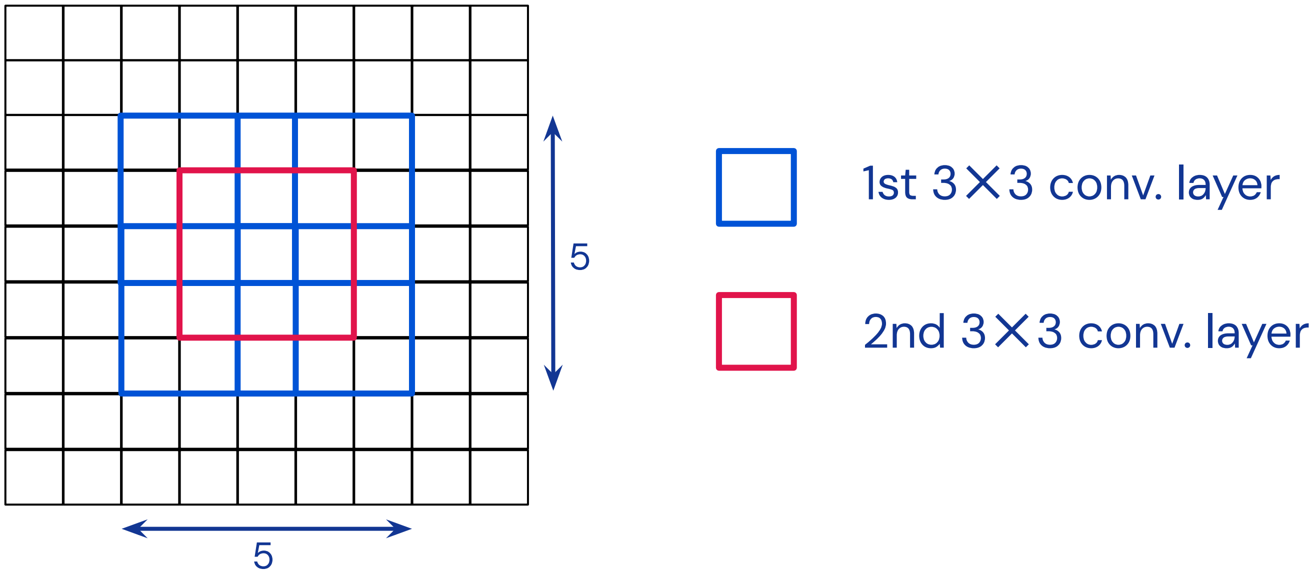


Want to learn more?

Simonyan, K.; Zisserman, A.
Very deep convolutional networks for large-scale image recognition International Conference on Learning Representations (2015)

Stack many convolutional layers before pooling
Use “same” convolutions to avoid resolution reduction

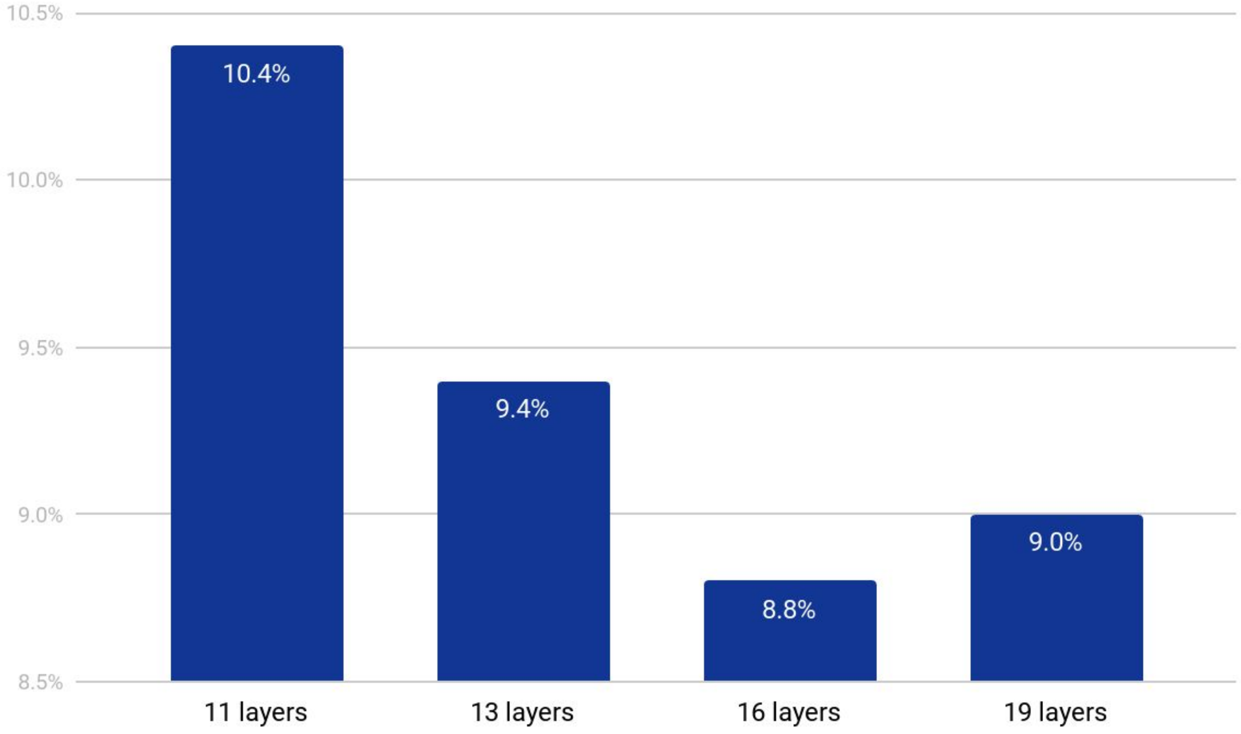
VGGNet (2014): stacking 3×3 kernels



Architecture: up to 19 layers, 3×3 kernels only, “same” convolutions

Infrastructure: trained for 2–3 weeks on 4 GPUs (data parallelism)

VGGNet (2014): error plateaus after 16 layers



Challenges of depth

- Computational complexity
- Optimisation difficulties

Improving optimisation

- Careful initialisation
- Sophisticated optimisers
- Normalisation layers
- Network design

GoogLeNet (2014)

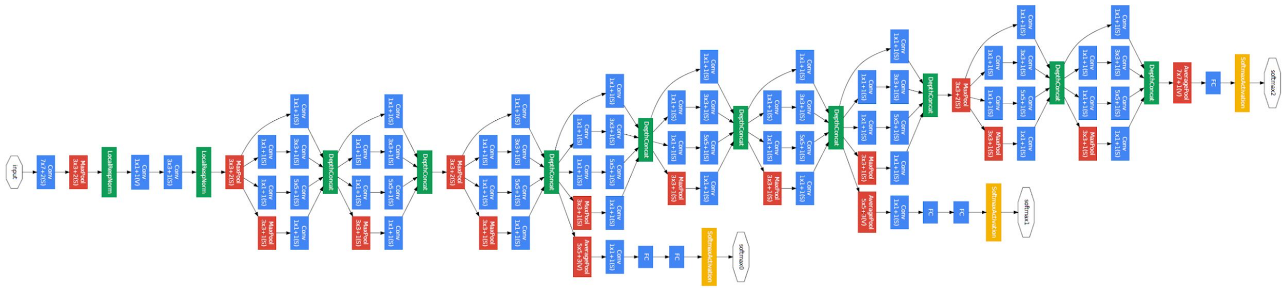
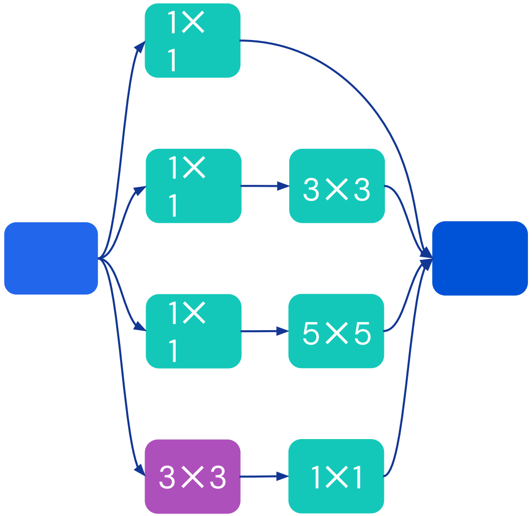


Figure from Szegedy et al. (2015)



Want to learn more?

Szegedy, C. et al.
Going deeper with convolutions IEEE conference on computer vision and pattern recognition (**2015**)

Batch normalisation

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Figure from Ioffe et al. (2015)

Want to learn more?

Ioffe, S.; Szegedy, C.
Batch normalization: Accelerating deep network training by reducing internal covariate shift International conference on machine learning (2015)

Reduces sensitivity to **initialisation**

Introduces stochasticity and acts as a **regulariser**

Batch normalisation

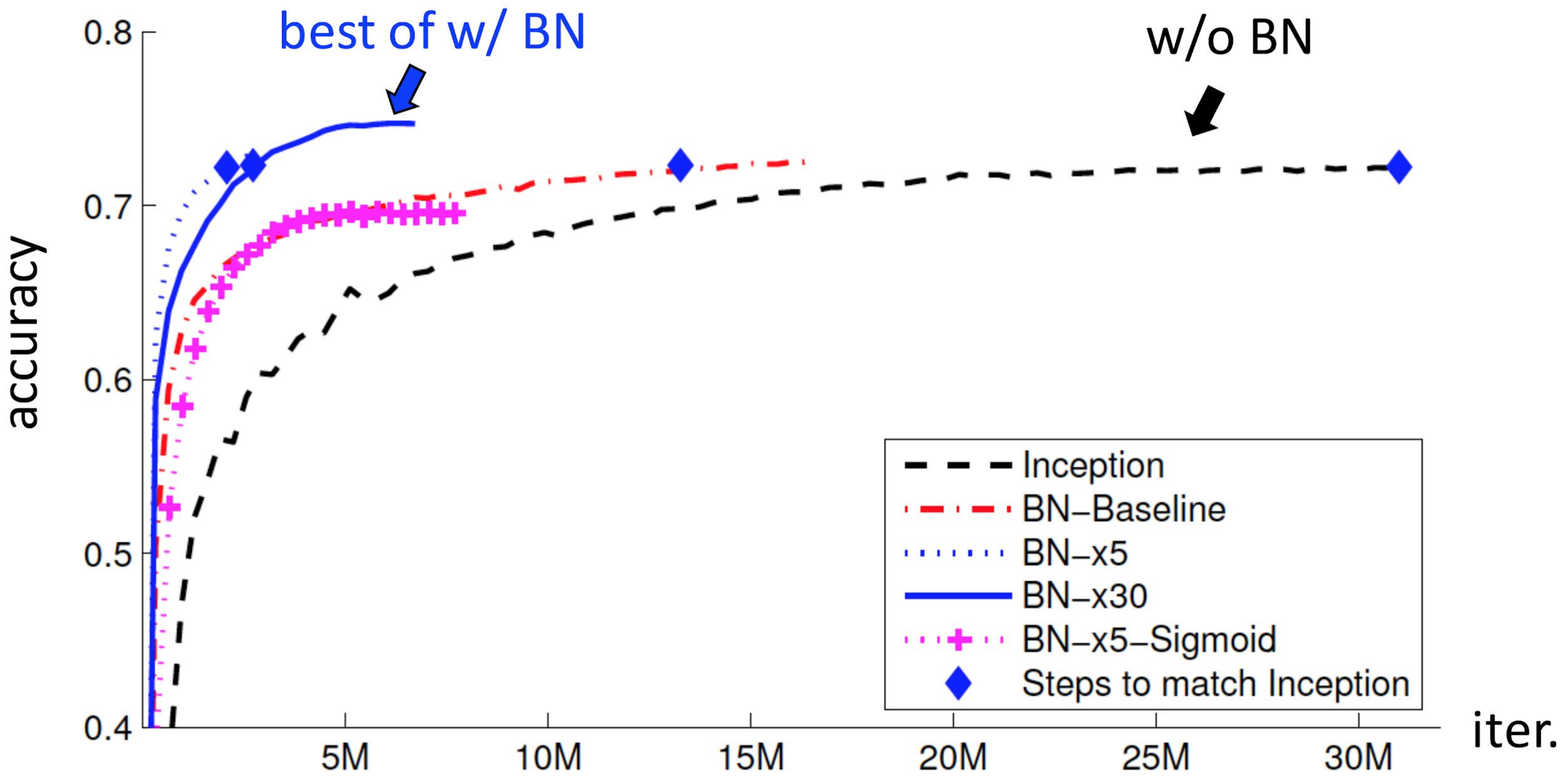
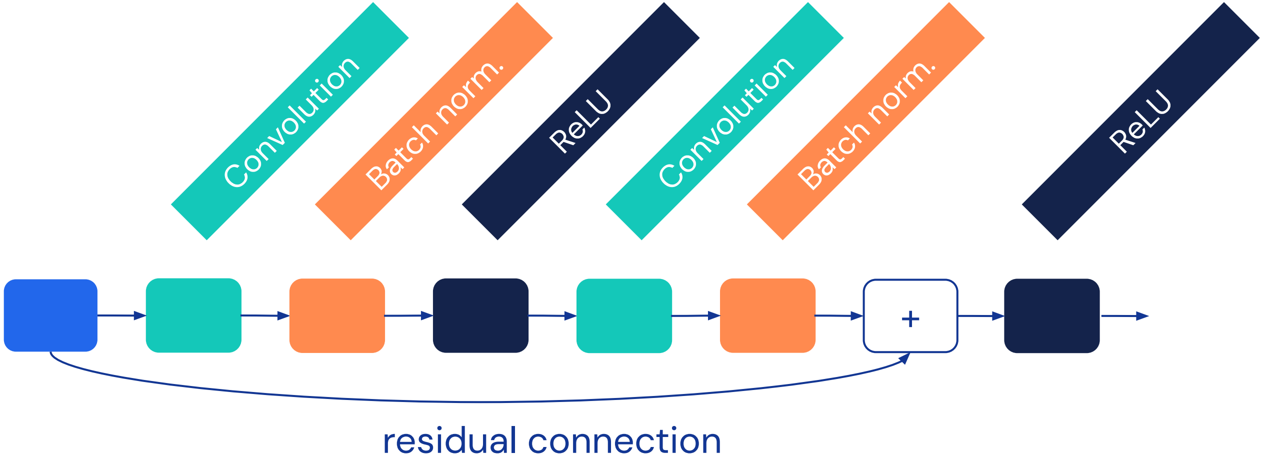


Figure from Ioffe et al. (2015)

ResNet (2015): residual connections

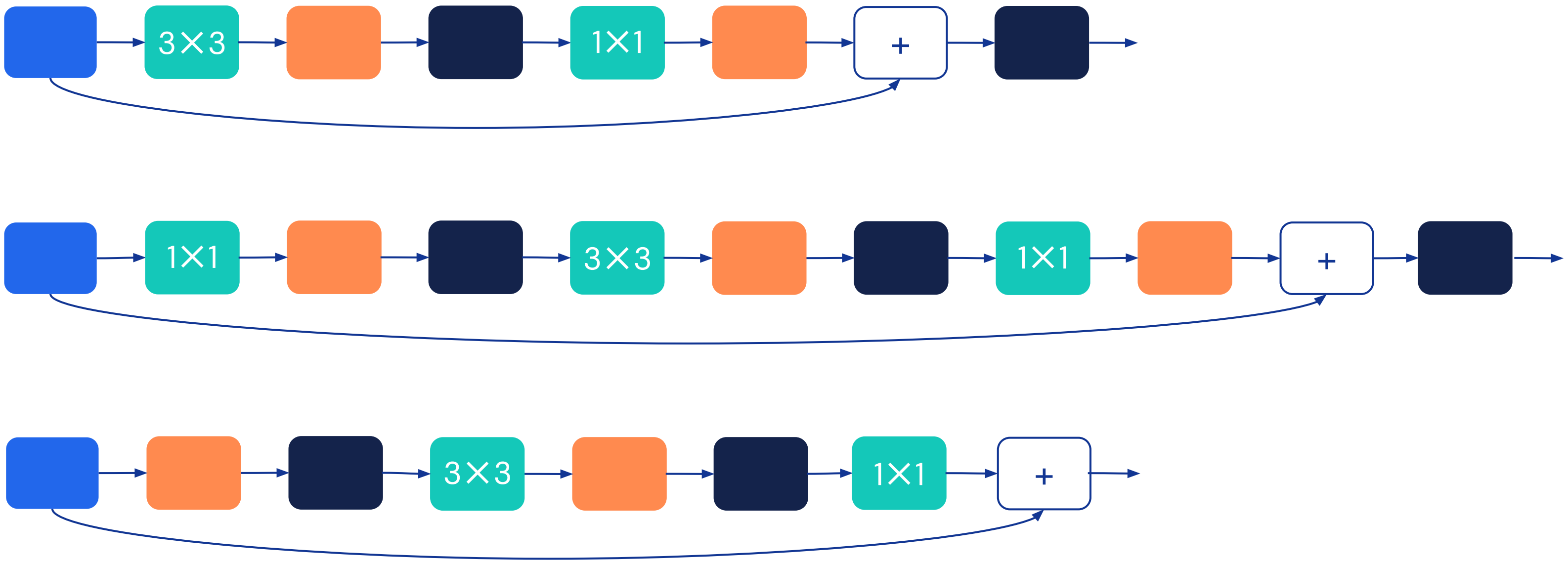


Want to learn more?

He, K. et al.
Deep residual learning for image recognition
IEEE conference on computer vision and pattern recognition (**2016**)

Residual connections facilitate training deeper networks

ResNet (2015): different flavours



Want to learn more?

He, K. et al.
Identity mappings in deep residual networks
European conference on computer vision
(2016)

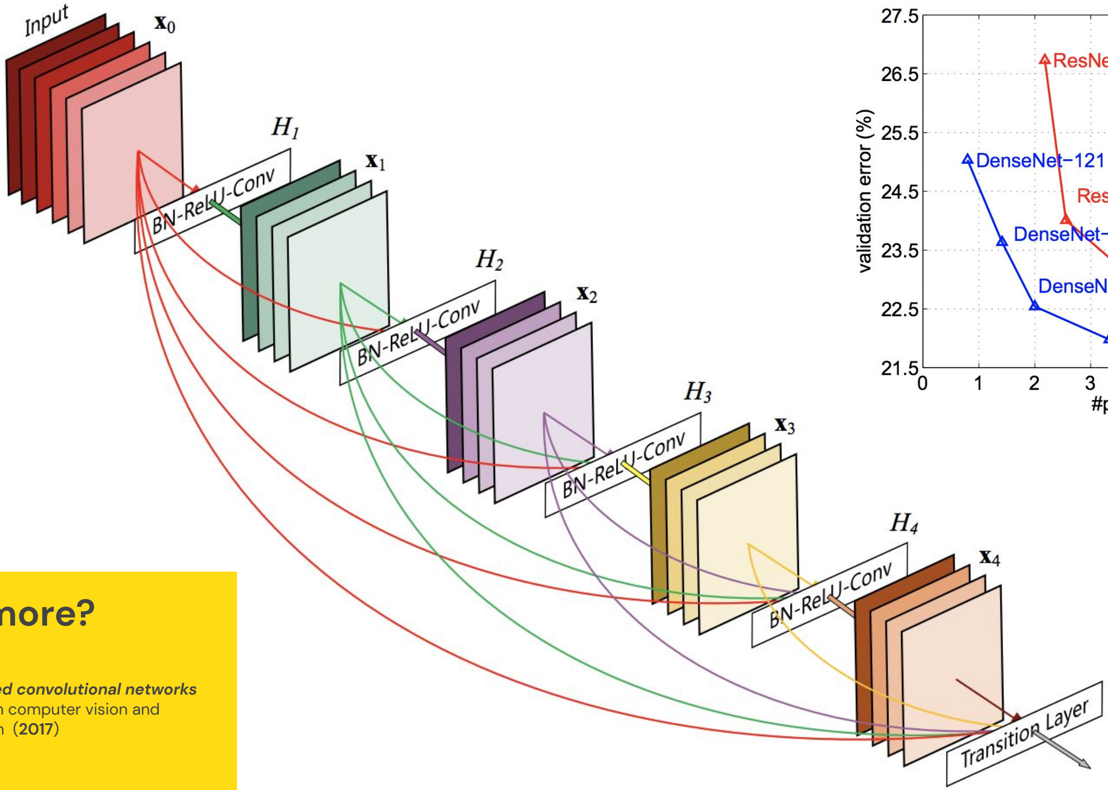
ResNet V2 (bottom) avoids all nonlinearities in the residual pathway

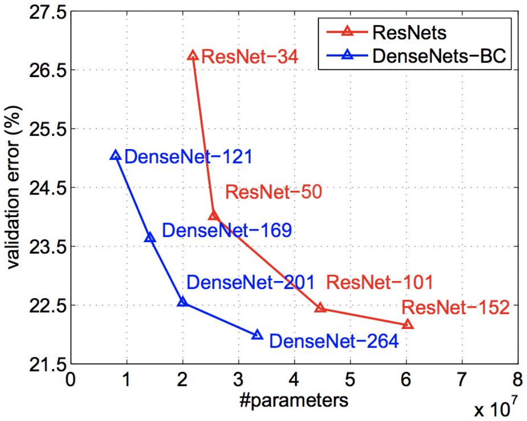
ResNet (2015): up to 152 layers

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table from He et al. (2015)

DenseNet (2016): connect layers to all previous layers





Want to learn more?

Huang, G. et al.
Densely connected convolutional networks
IEEE conference on computer vision and pattern recognition (2017)

Figures from Huang et al. (2015)

Squeeze-and-excitation networks (2017)

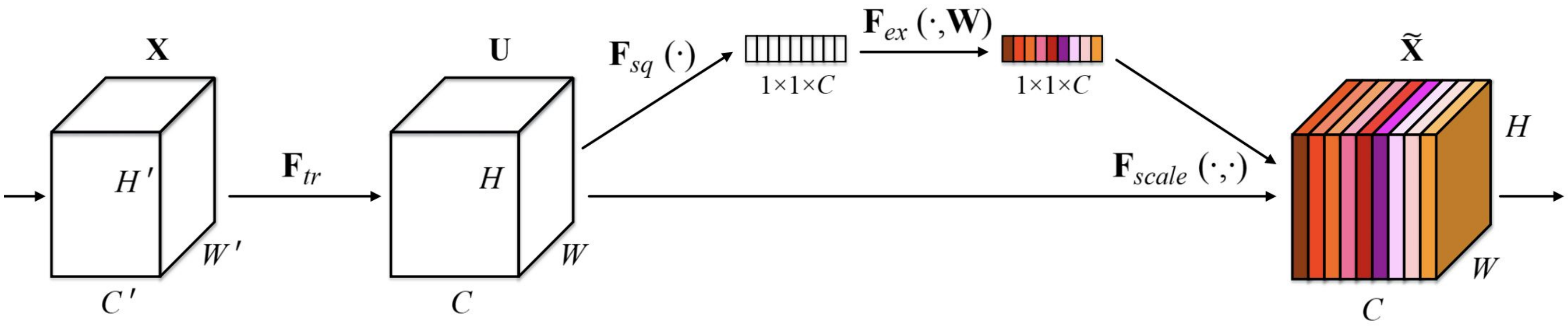


Figure from Hu et al. (2018)

Want to learn more?

Hu, J.; Shen, L.; Sun, G.
Squeeze-and-excitation networks *IEEE conference on computer vision and pattern recognition* (2018)

Features can incorporate **global context**

AmoebaNet (2018): neural architecture search

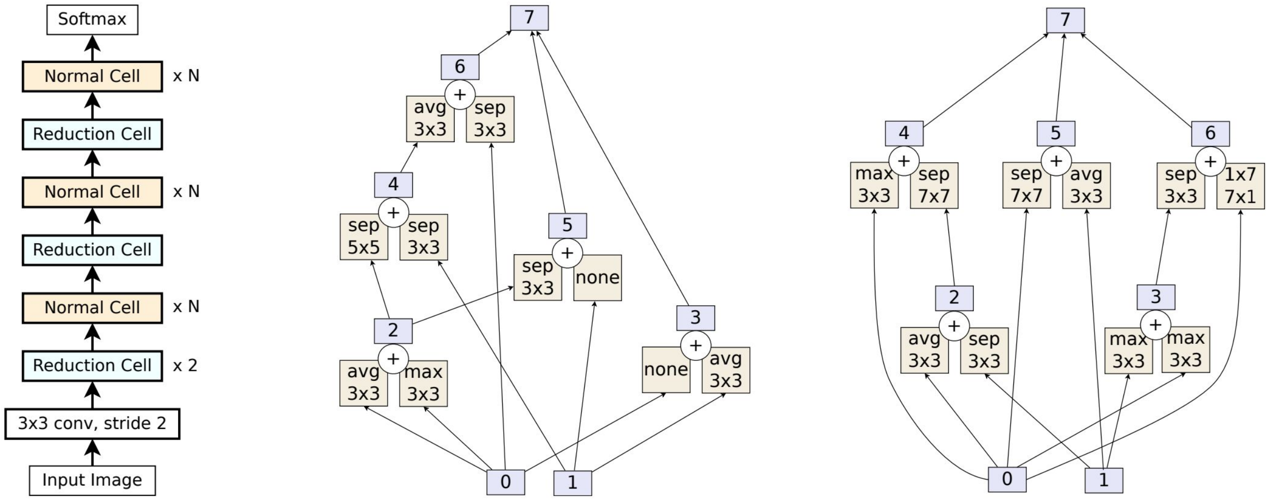


Figure from Real et al. (2019)

Want to learn more?

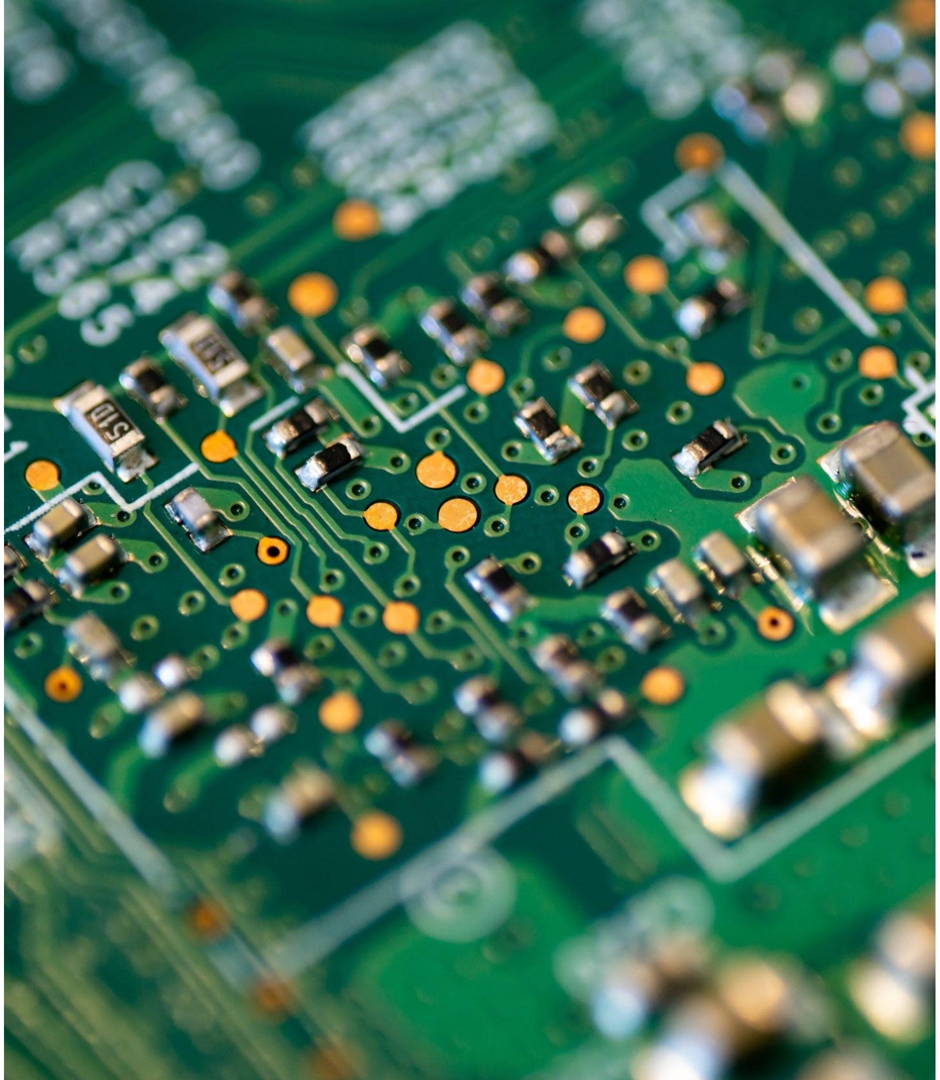
Real, E. et al.
Regularized evolution for image classifier architecture search *AAAI conference on artificial intelligence* (2019)

Architecture found by **evolution**

Search **acyclic graphs** composed of predefined layers

Reducing complexity

- Depthwise convolutions
- Separable convolutions
- Inverted bottlenecks (MobileNetV2, MNasNet, EfficientNet)



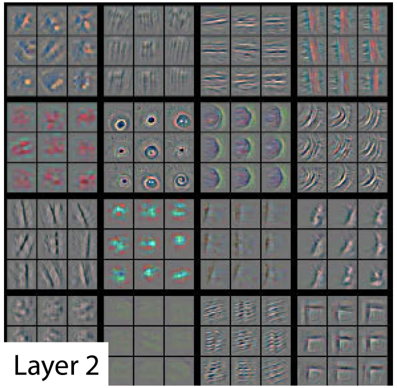
5 Advanced topics

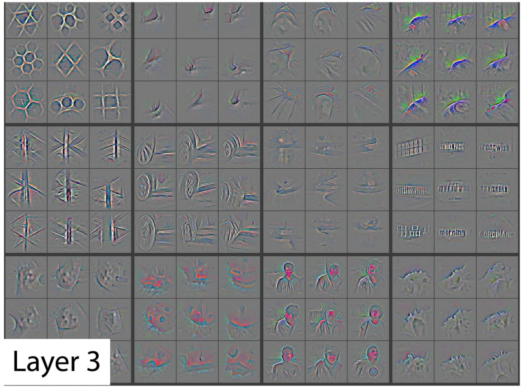
Data augmentation

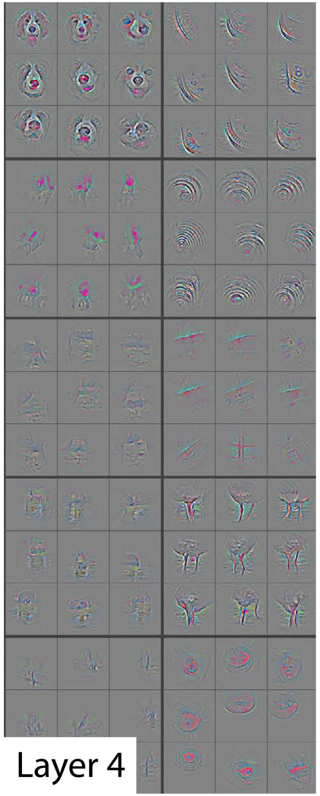
By design, convnets are only robust against **translation**

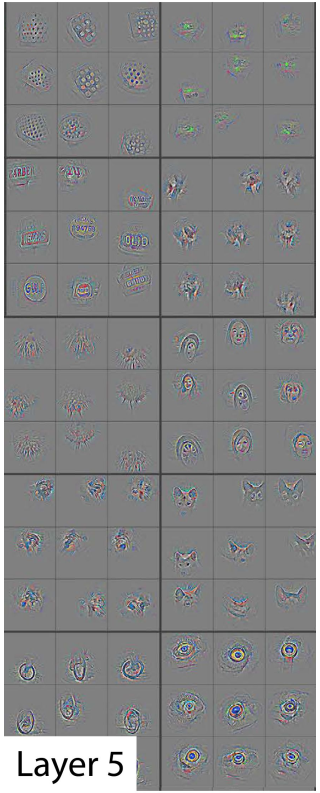
Data augmentation makes them robust against other transformations: rotation, scaling, shearing, warping, ...

Visualising what a convnet learns









Figures from Zeiler et al. (2014)

Want to learn more?

Zeiler, M.D.; Fergus, R.
Visualizing and understanding convolutional networks *European conference on computer vision* (2014)

Visualising what a convnet learns

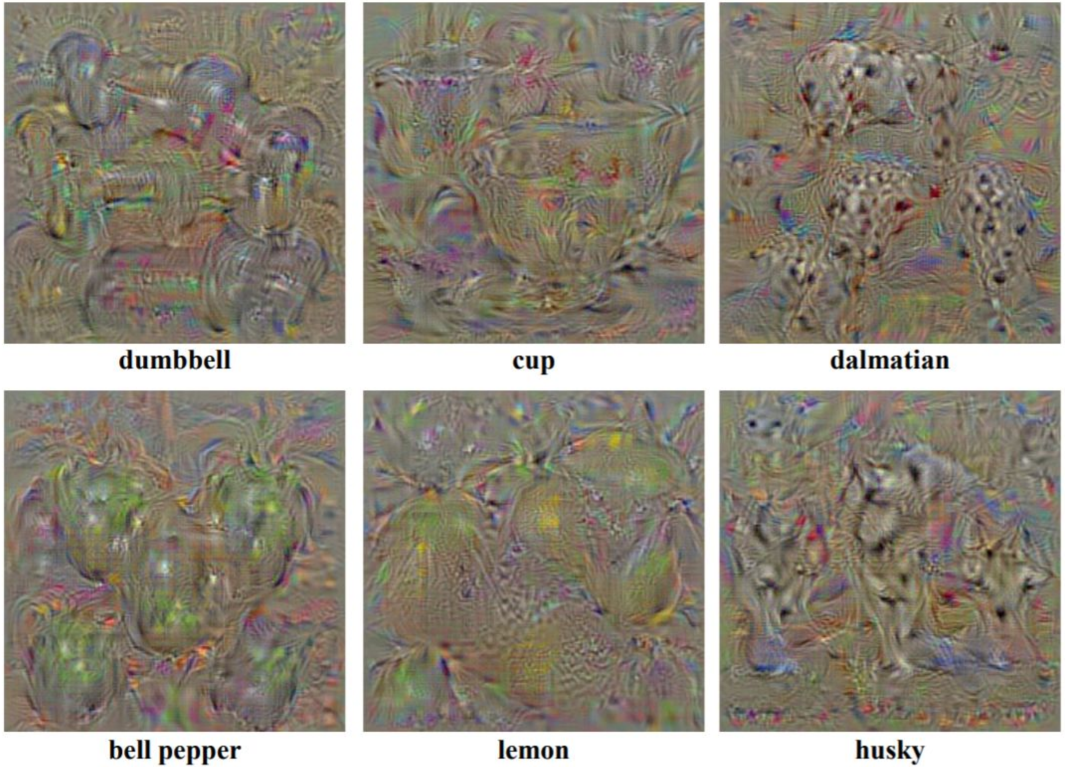


Figure from Simonyan et al. (2013)

Visualising what a convnet learns

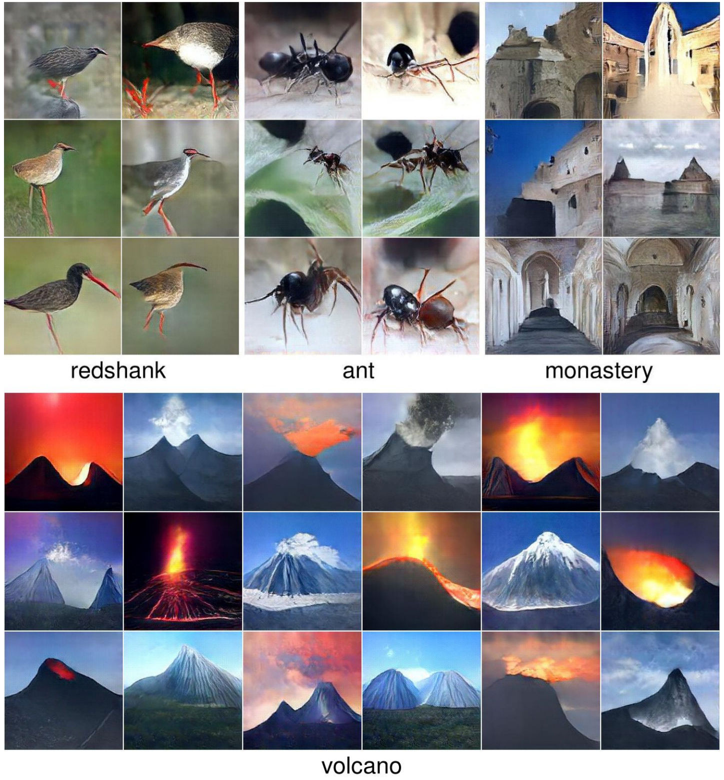
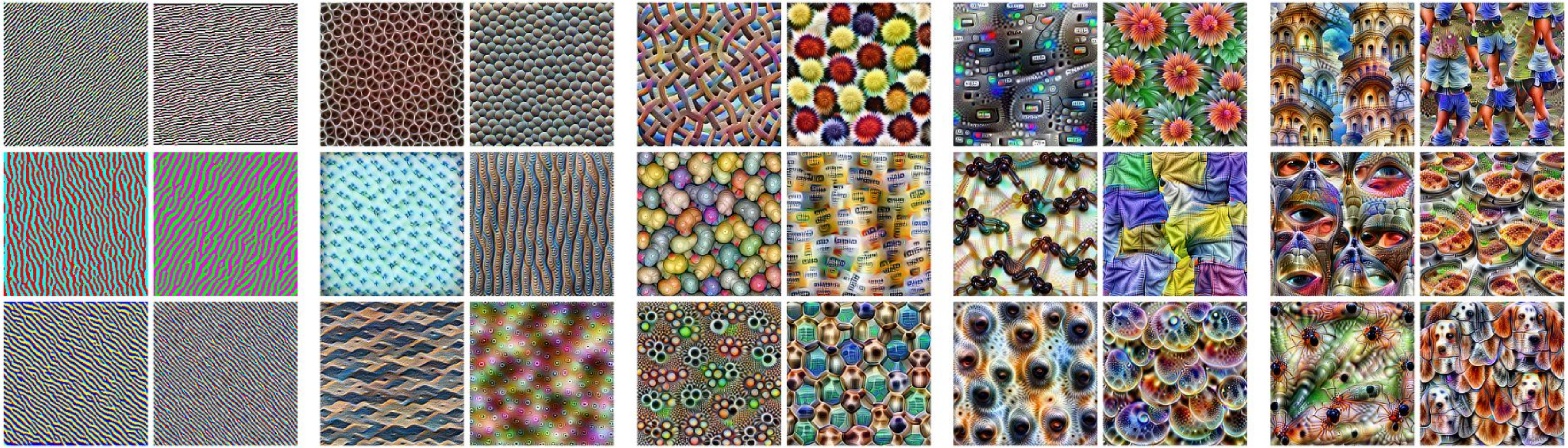


Figure from Nguyen et al. (2016)

https://distill.pub/2017/feature-visualization/ by **Chris Olah**, **Alexander Mordvintsev** and **Ludwig Schubert**

Feature Visualization

How neural networks build up their understanding of images



Edges (layer conv2d0) **Textures** (layer mixed3a) **Patterns** (layer mixed4a) **Parts** (layers mixed4b & mixed4c) **Objects** (layers mixed4d & mixed4e)

Feature visualization allows us to see how GoogLeNet [1], trained on the ImageNet [2] dataset, builds up its understanding of images over many layers. Visualizations of all channels are available in the appendix.

AUTHORS
Chris Olah
Alexander Mordvintsev
Ludwig Schubert

AFFILIATIONS
Google Brain Team
Google Research
Google Brain Team

PUBLISHED
Nov. 7, 2017

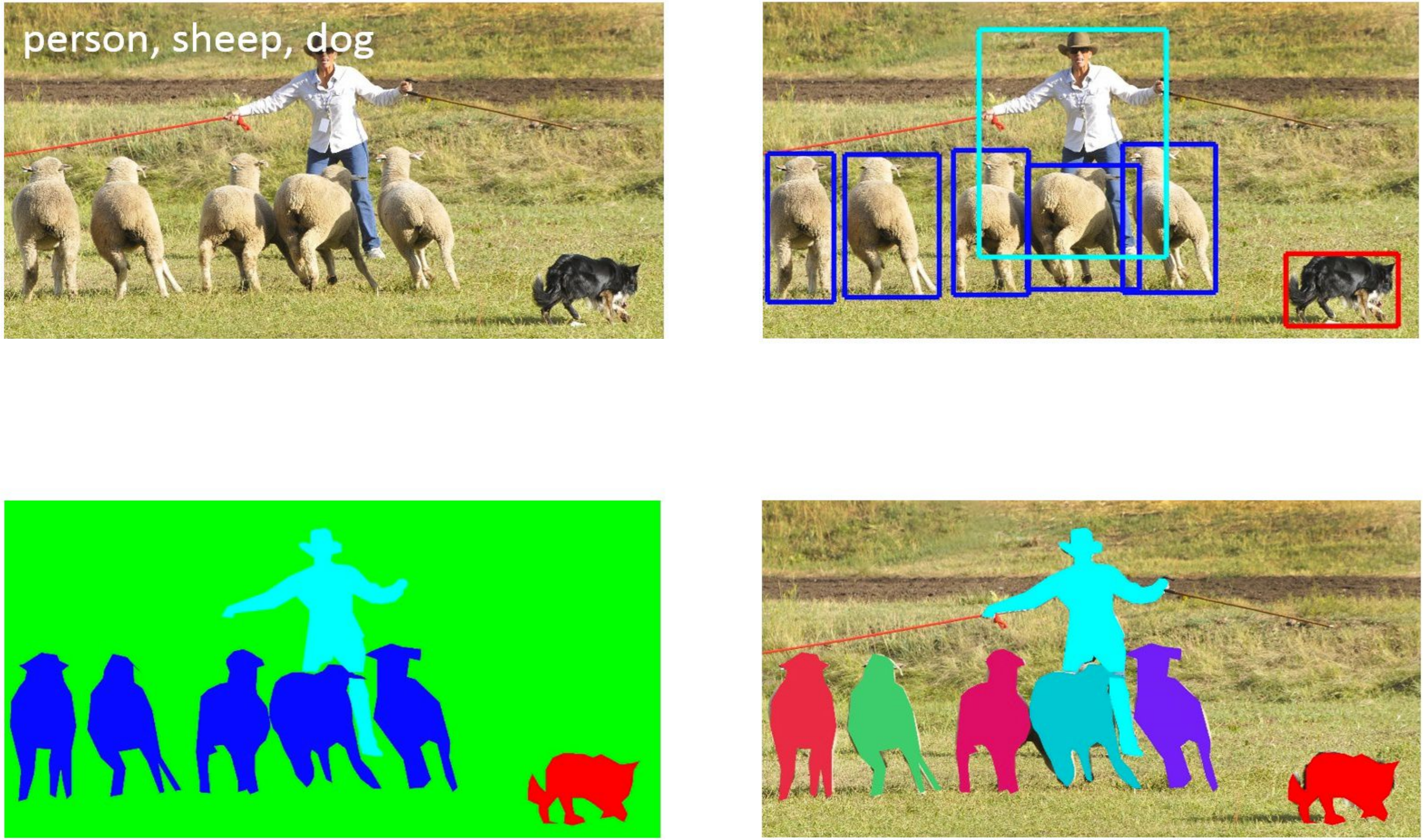
DOI
10.23915/distill.00007

Other topics to explore

- Pre-training and fine-tuning
- Group equivariant convnets: invariance to e.g. rotation
- Recurrence and attention: other building blocks to exploit topological structure

6 Beyond image recognition

What else can we do with convnets?



Figures from Lin et al. (2015)

Generative models of images

- Generative adversarial nets
- Variational autoencoders
- Autoregressive models (PixelCNN)




More convnets

- Representation learning and self-supervised learning
- Convnets for video, audio, text, graphs, …

Convolutional neural networks replaced **handcrafted features** With **handcrafted architectures.**

Prior knowledge is not obsolete: it is merely incorporated at a higher level of abstraction.

Thank you

Questions