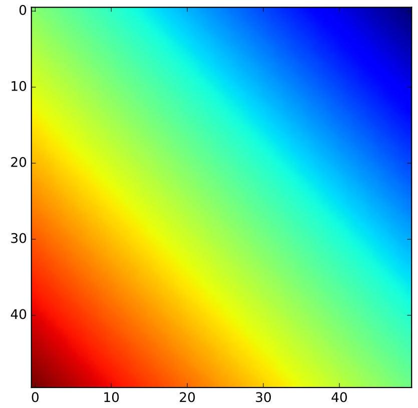
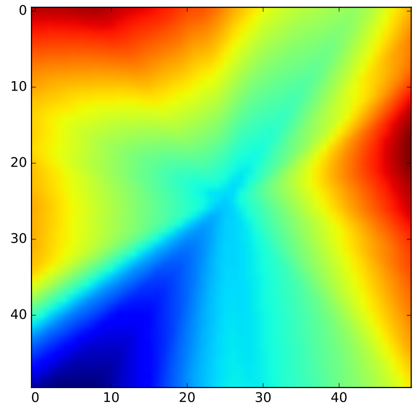
Multilayer perceptrons

- Key idea: build complex functions by composing simple functions

f(x) = Wx → g(x) = max(x,0) → f(x) = Wx → g(x) = max(x,0) → f(x) = Wx

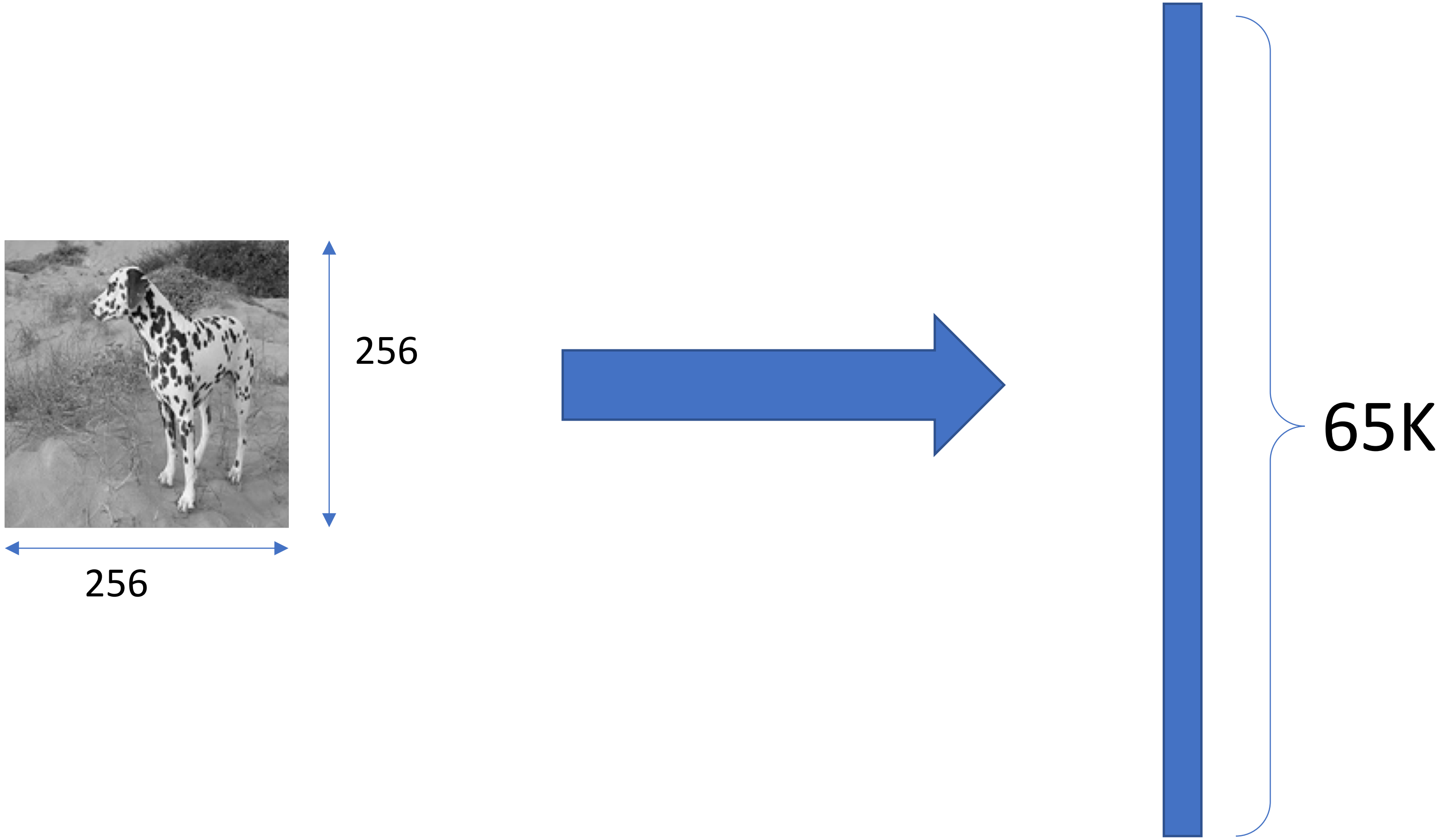




Multilayer perceptrons

- Key idea: build complex functions by composing simple functions
- Caveat: simple functions must include non-linearities
- $W(U(Vx)) = (WUV)x$

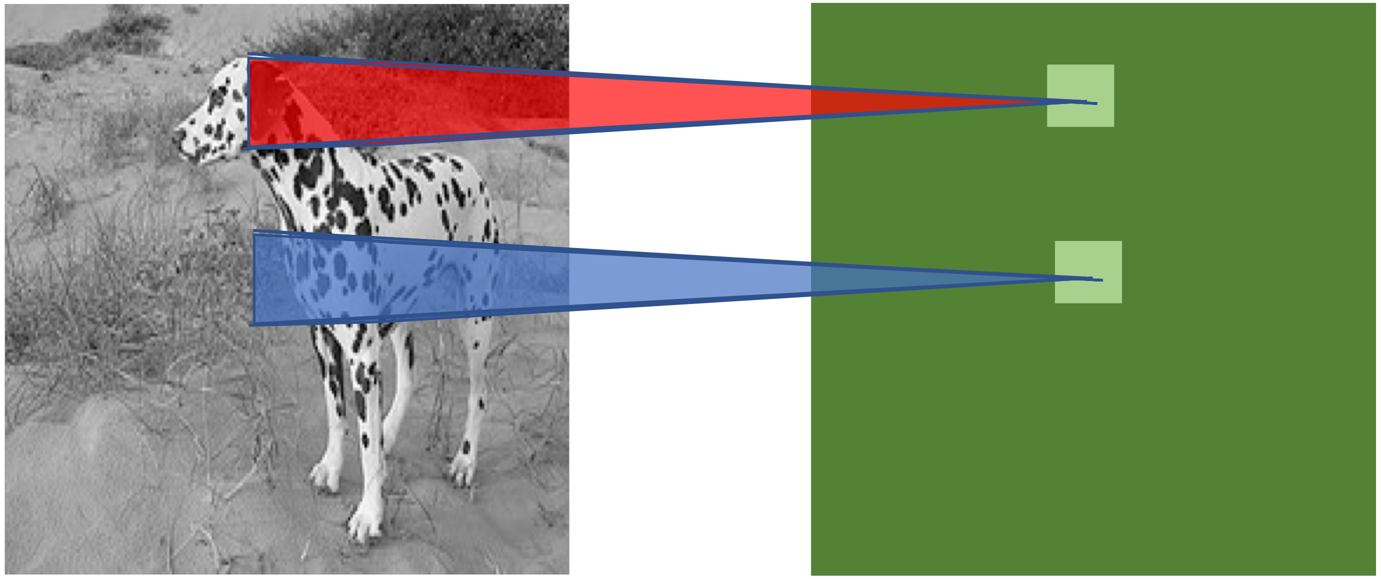
Reducing capacity



Reducing capacity

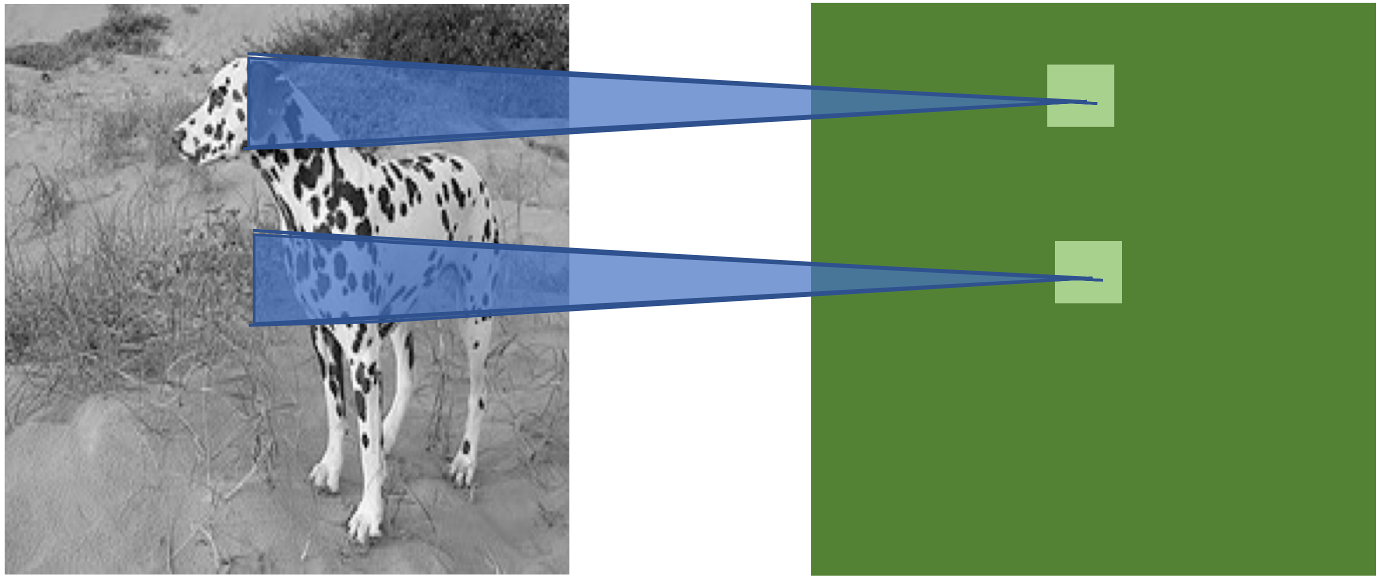
Idea 1: local connectivity

- Inputs and outputs are *feature maps*
- Pixels only related to nearby pixels



Idea 2: Translation invariance

- Pixels only related to nearby pixels

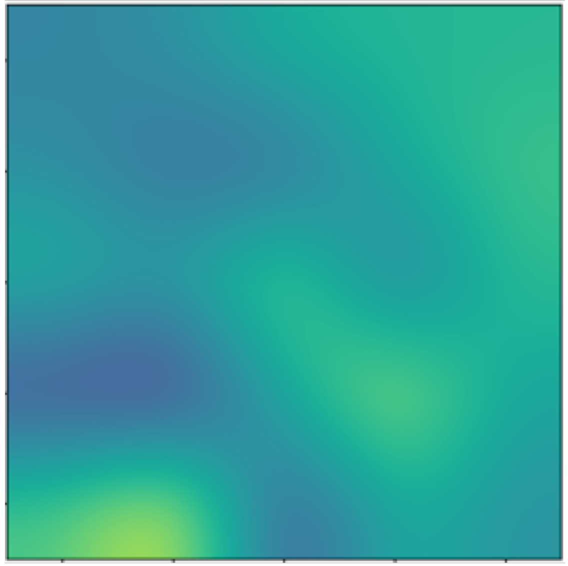


Local connectivity + translation invariance = *convolution*

5.4	0.1	3.6
1.8	2.3	4.5
1.1	3.4	7.2

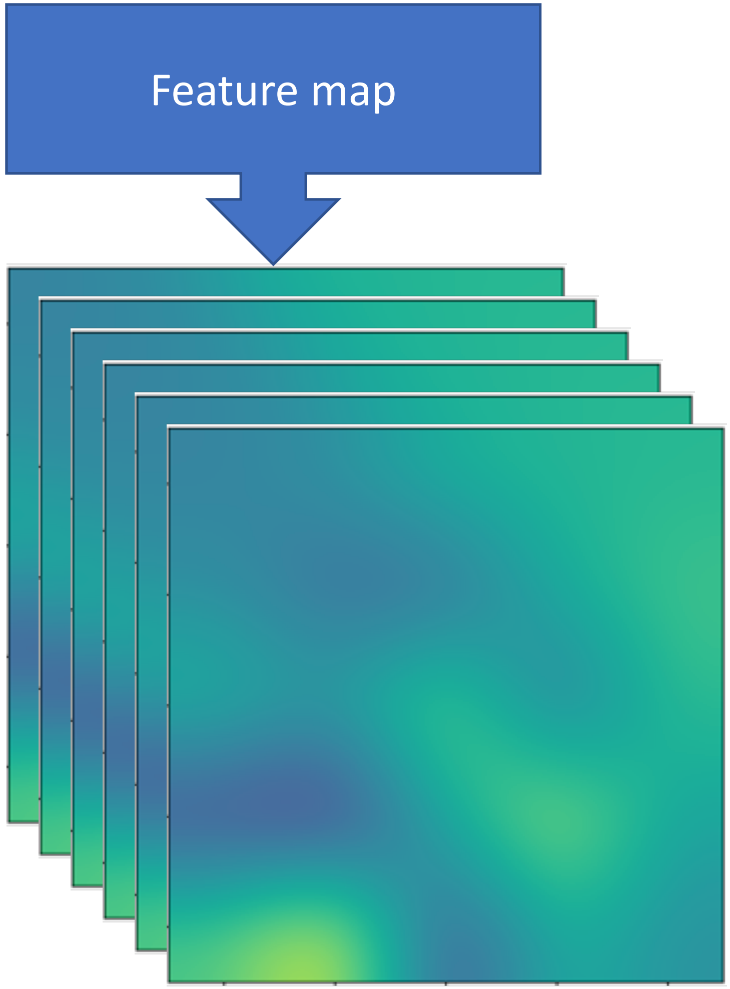
Local connectivity + translation invariance = *convolution*

5.4	0.1	3.6
1.8	2.3	4.5
1.1	3.4	7.2

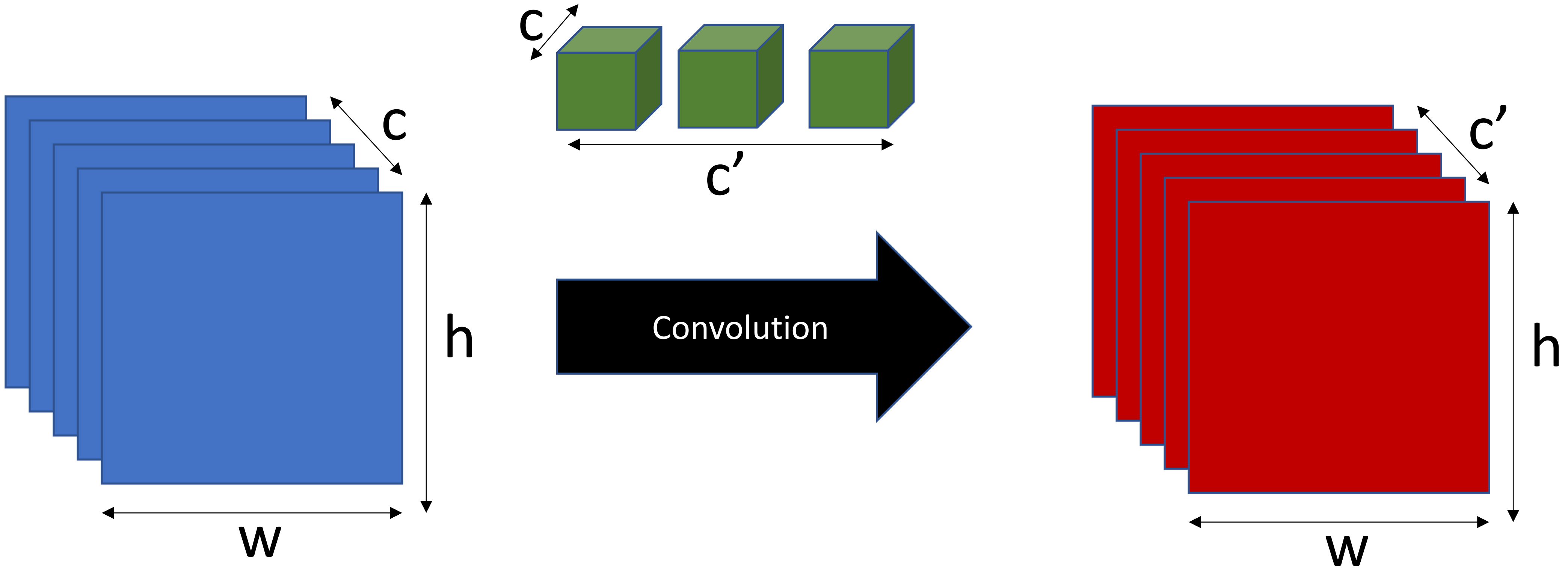


Local connectivity + translation invariance = *convolution*

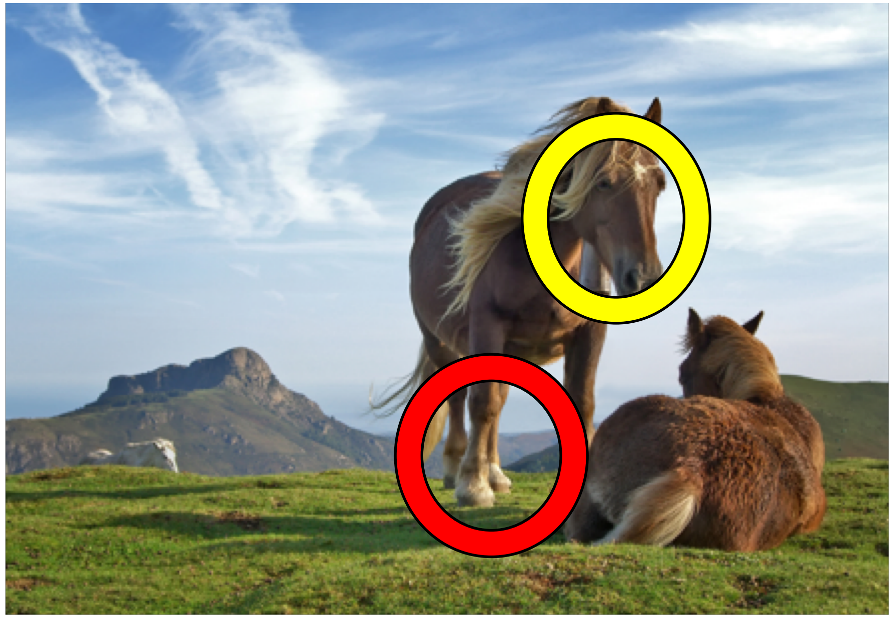
5.4	0.1	3.6
1.8	2.3	4.5
1.1	3.4	7.2



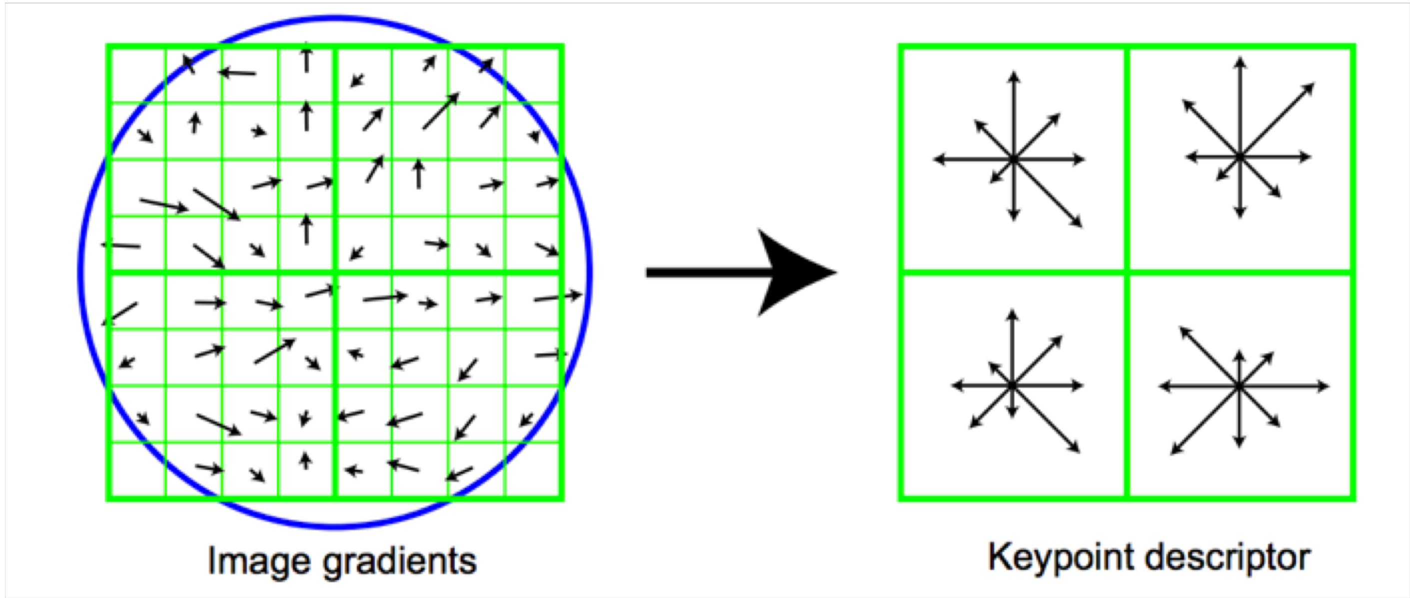
Convolution as a primitive



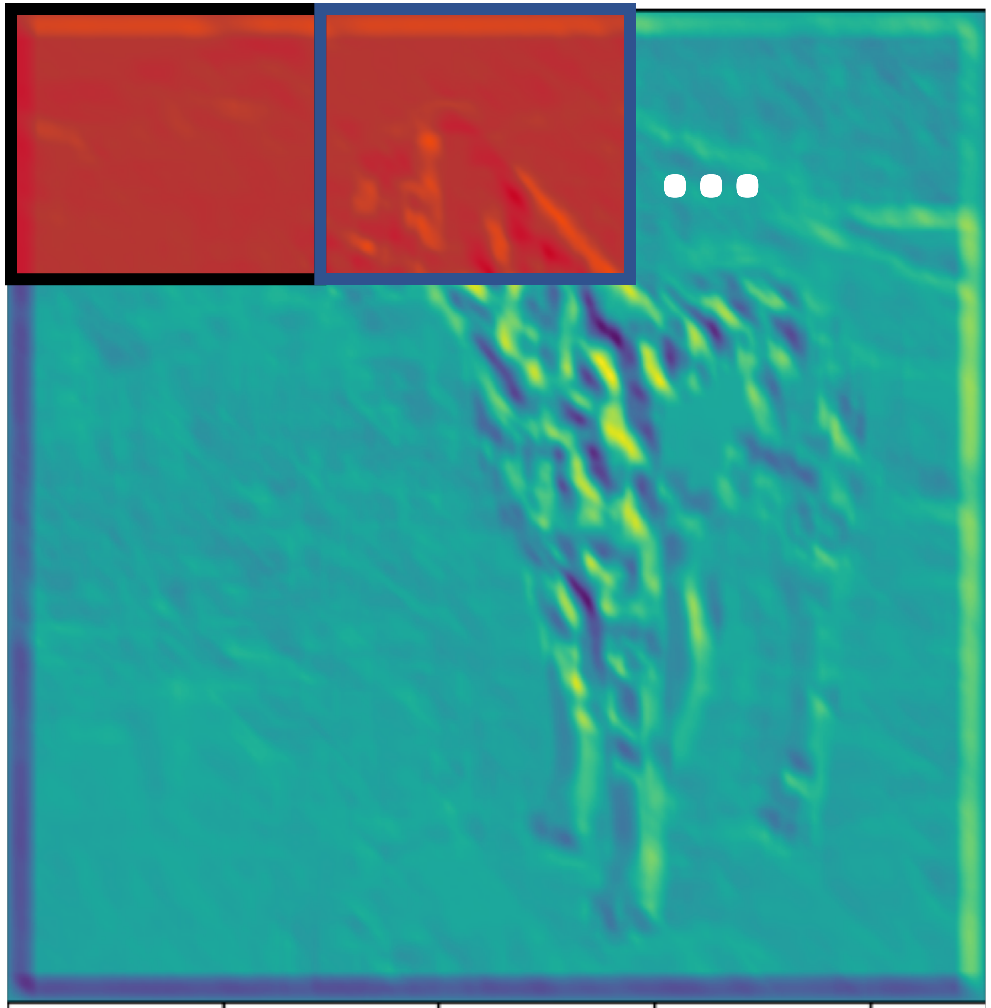
Invariance to distortions



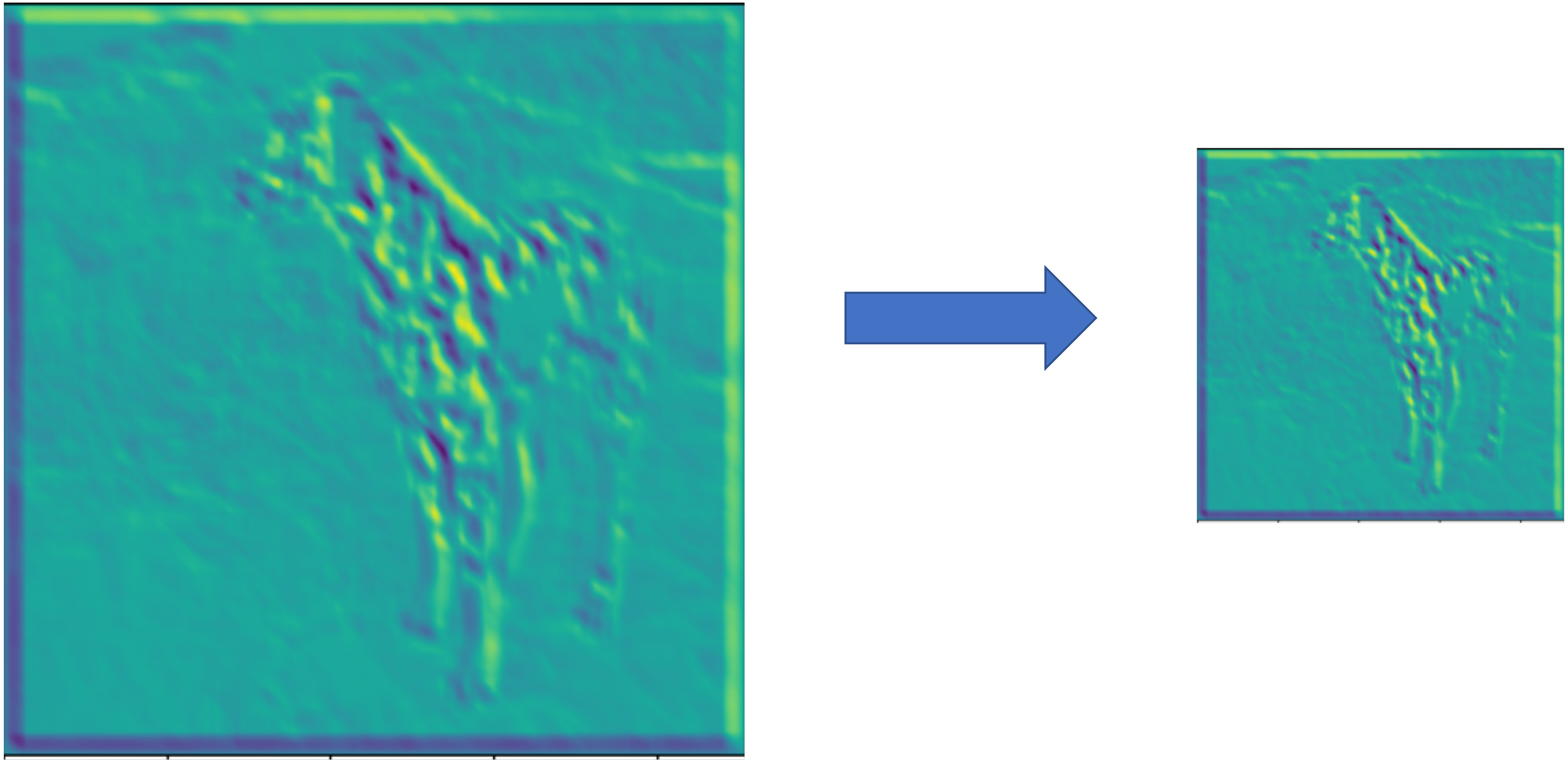
Invariance to distortions



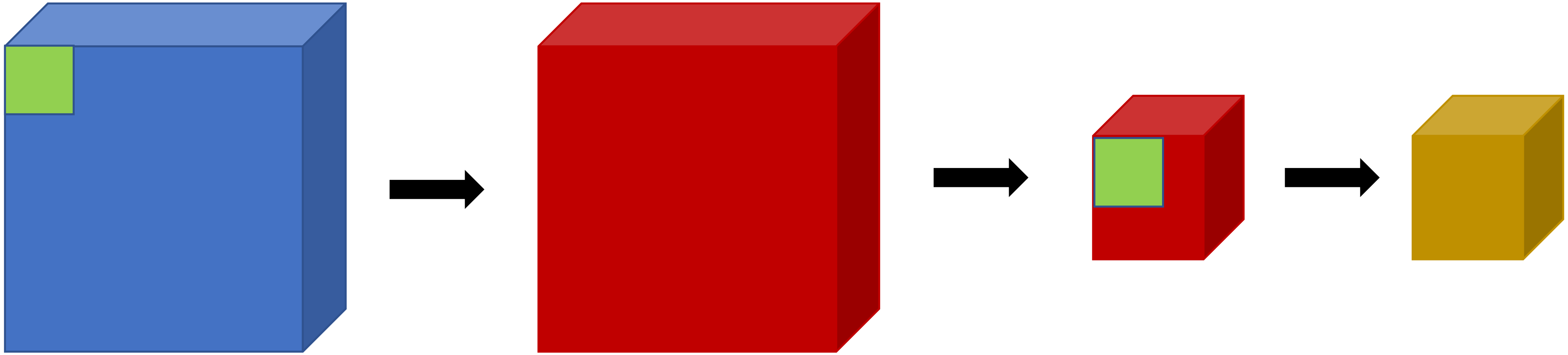
Invariance to distortions: Pooling



Invariance to distortions: Subsampling



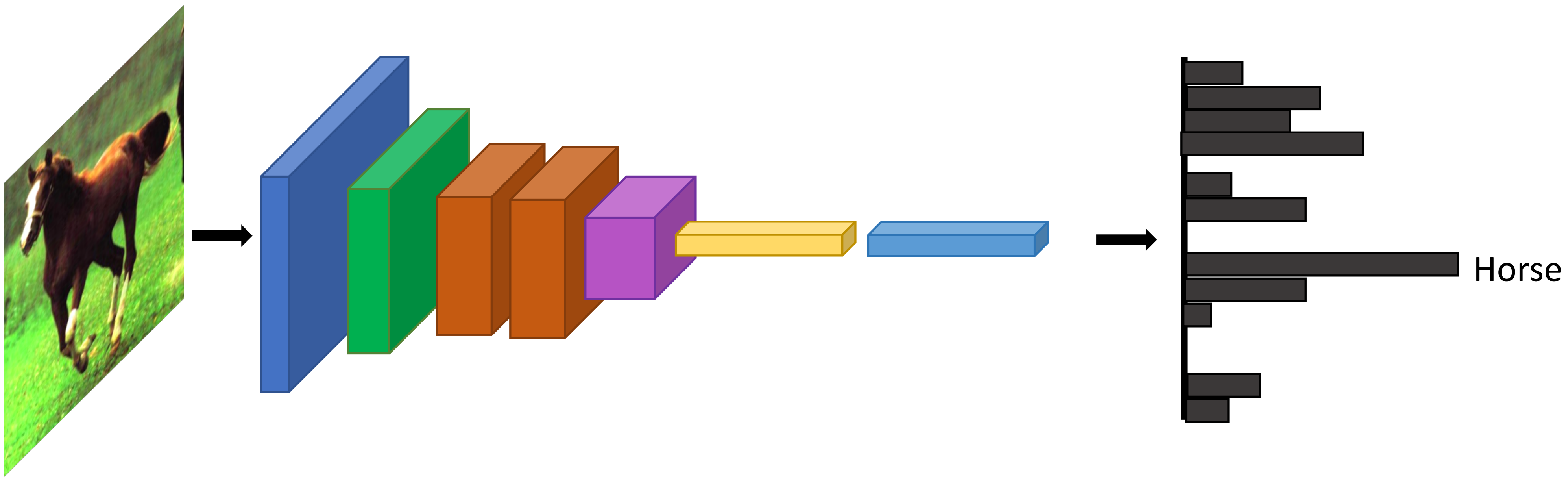
Convolution subsampling convolution



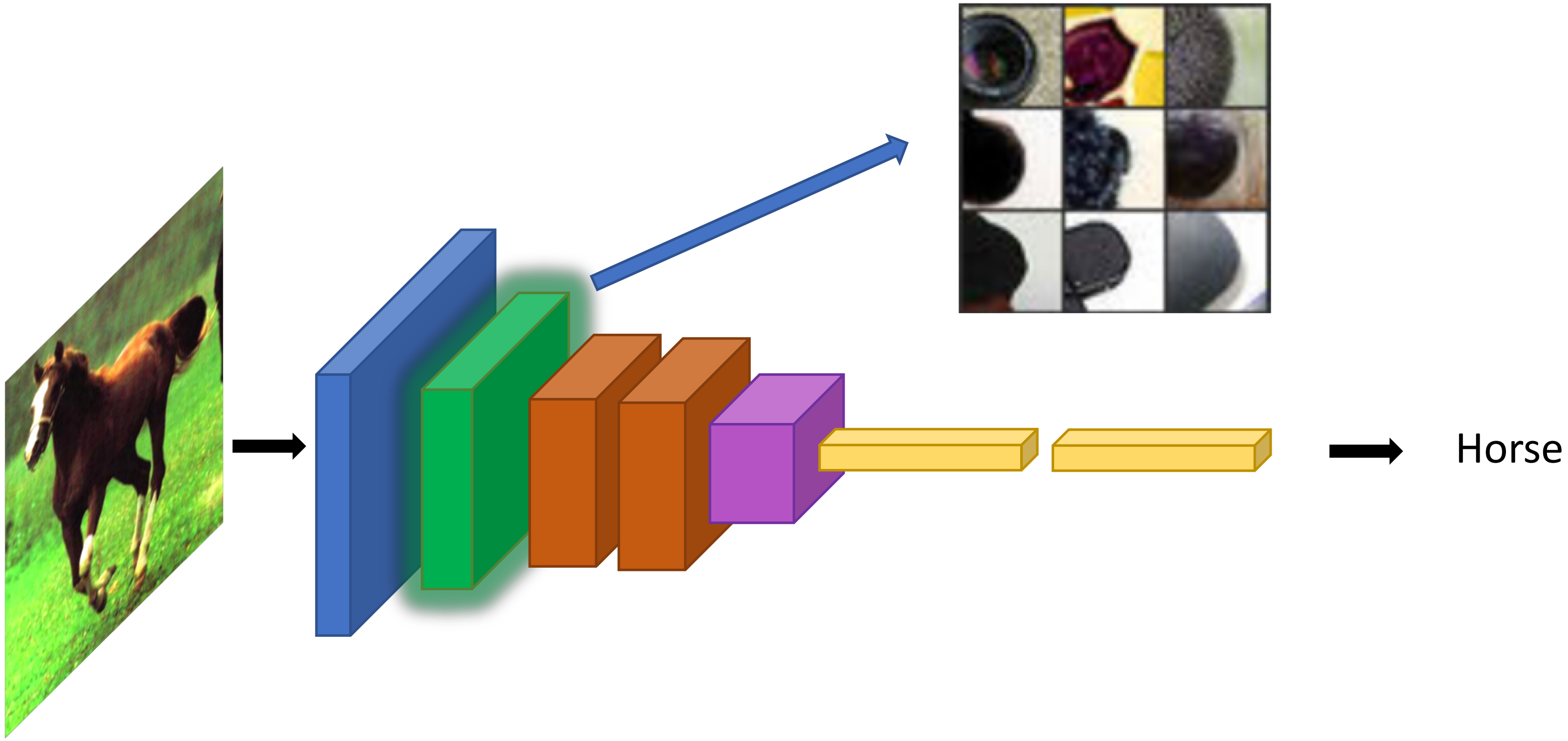
Convolution subsampling convolution

- Convolution in earlier steps detects *more local* patterns *less resilient* to distortion
- Convolution in later steps detects *more global* patterns *more resilient* to distortion
- Subsampling allows capture of *larger, more invariant* patterns

Convolutional networks

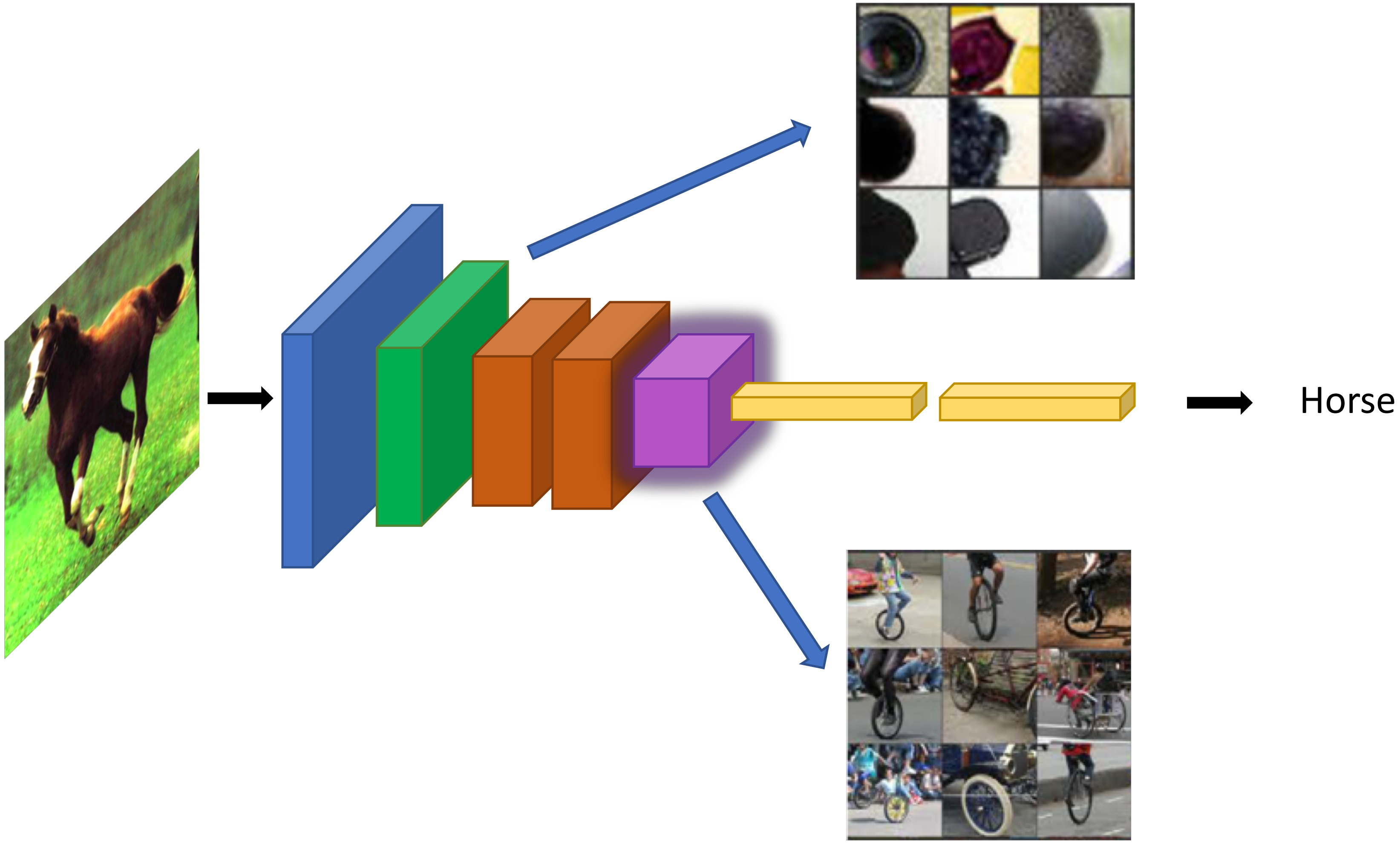


Convolutional networks



Visualizations from : M. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In ECCV 2014.

Convolutional networks



Visualizations from : M. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In *ECCV* 2014.

Vagaries of optimization

- Non-convex
  - Local optima
  - Sensitivity to initialization
- Vanishing / exploding gradients

$$\frac{\partial z}{\partial z_i} = \frac{\partial z}{\partial z_{n-1}} \frac{\partial z_{n-1}}{\partial z_{n-2}} \cdots \frac{\partial z_{i+1}}{\partial z_i}$$

  - If each term is (much) greater than 1 → *explosion of gradients*
  - If each term is (much) less than 1 → *vanishing gradients*

Vanishing and exploding gradients

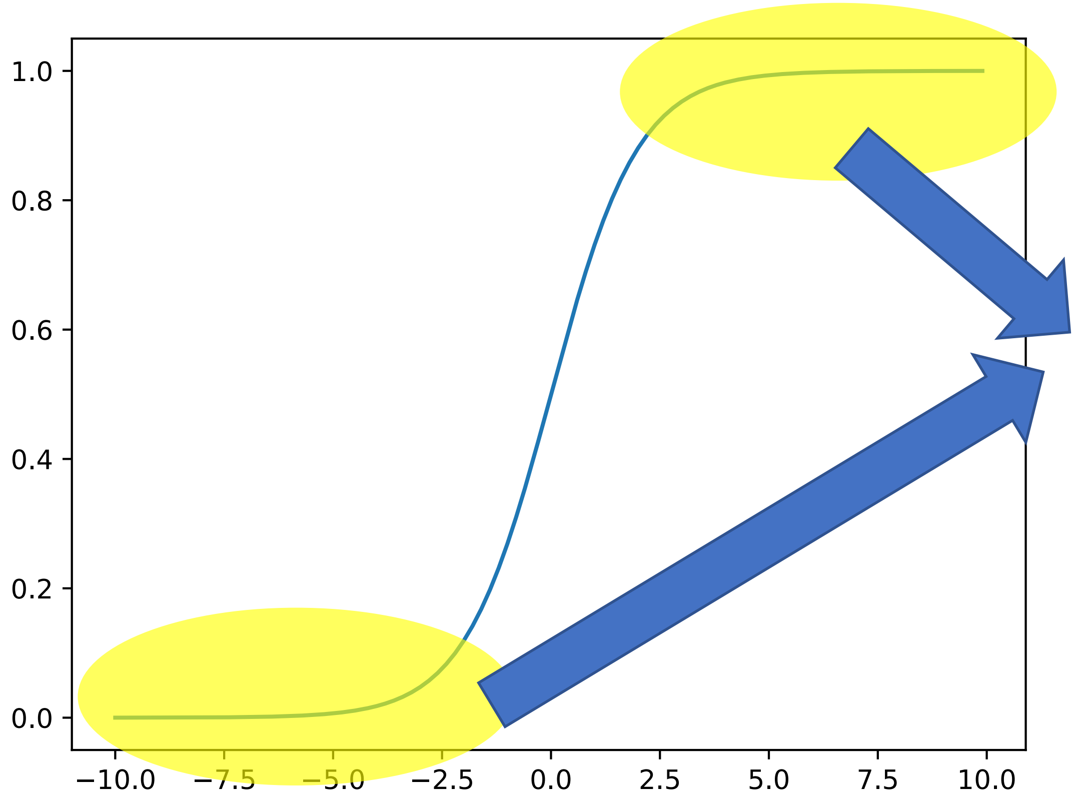
$$\frac{\partial \mathbf{z}}{\partial \mathbf{z}_i} = \frac{\partial \mathbf{z}}{\partial \mathbf{z}_{n-1}} \frac{\partial \mathbf{z}_{n-1}}{\partial \mathbf{z}_{n-2}} \cdots \frac{\partial \mathbf{z}_{i+1}}{\partial \mathbf{z}_i}$$

$$\frac{\partial L}{\partial \mathbf{z}_i} = \frac{\partial L}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{z}_i}$$

$$\lambda_{min}(\frac{\partial \mathbf{z}}{\partial \mathbf{z}_i})\frac{\partial L}{\partial \mathbf{z}} \leq \frac{\partial L}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{z}_i} \leq \lambda_{max}(\frac{\partial \mathbf{z}}{\partial \mathbf{z}_i})\frac{\partial L}{\partial \mathbf{z}}$$

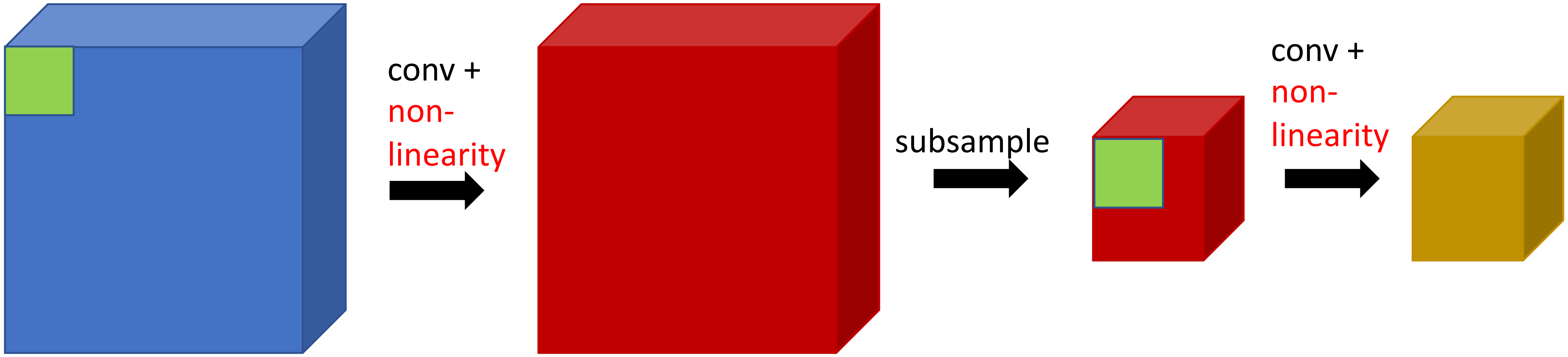
$$\lambda_{max}(UV) \leq \lambda_{max}(U)\lambda_{max}(V)$$
$$\lambda_{min}(UV) \geq \lambda_{min}(U)\lambda_{min}(V)$$
$$\lambda_{max}(A^n) = \lambda_{max}(A)^n$$
$$\lambda_{min}(A^n) = \lambda_{min}(A)^n$$

Sigmoids cause vanishing gradients



Gradient close to 0

Convolution subsampling convolution



Rectified Linear Unit (ReLU)

- max (x,0)
- Also called half-wave rectification (signal processing)

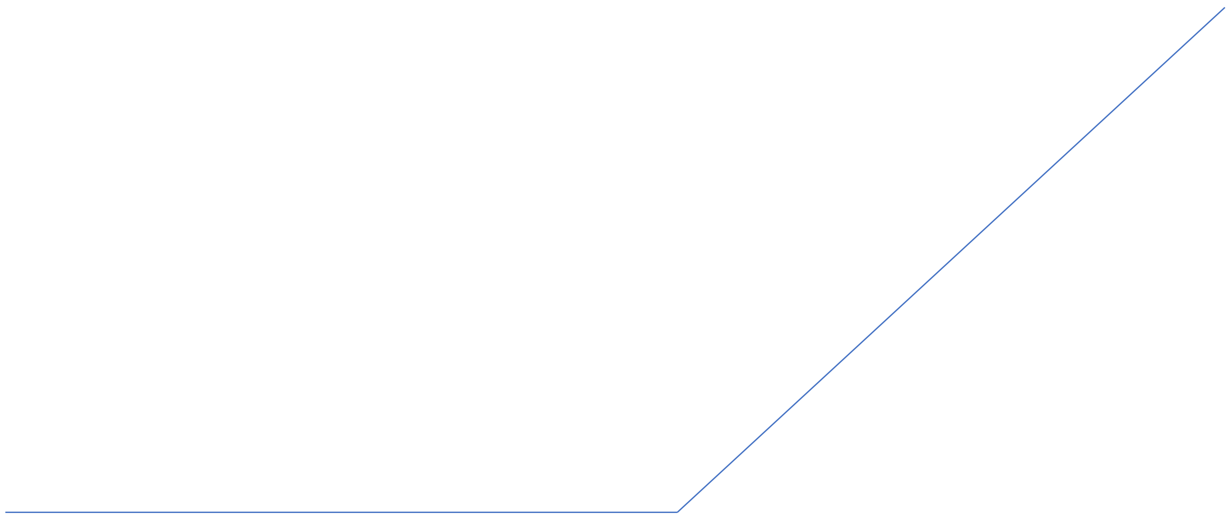
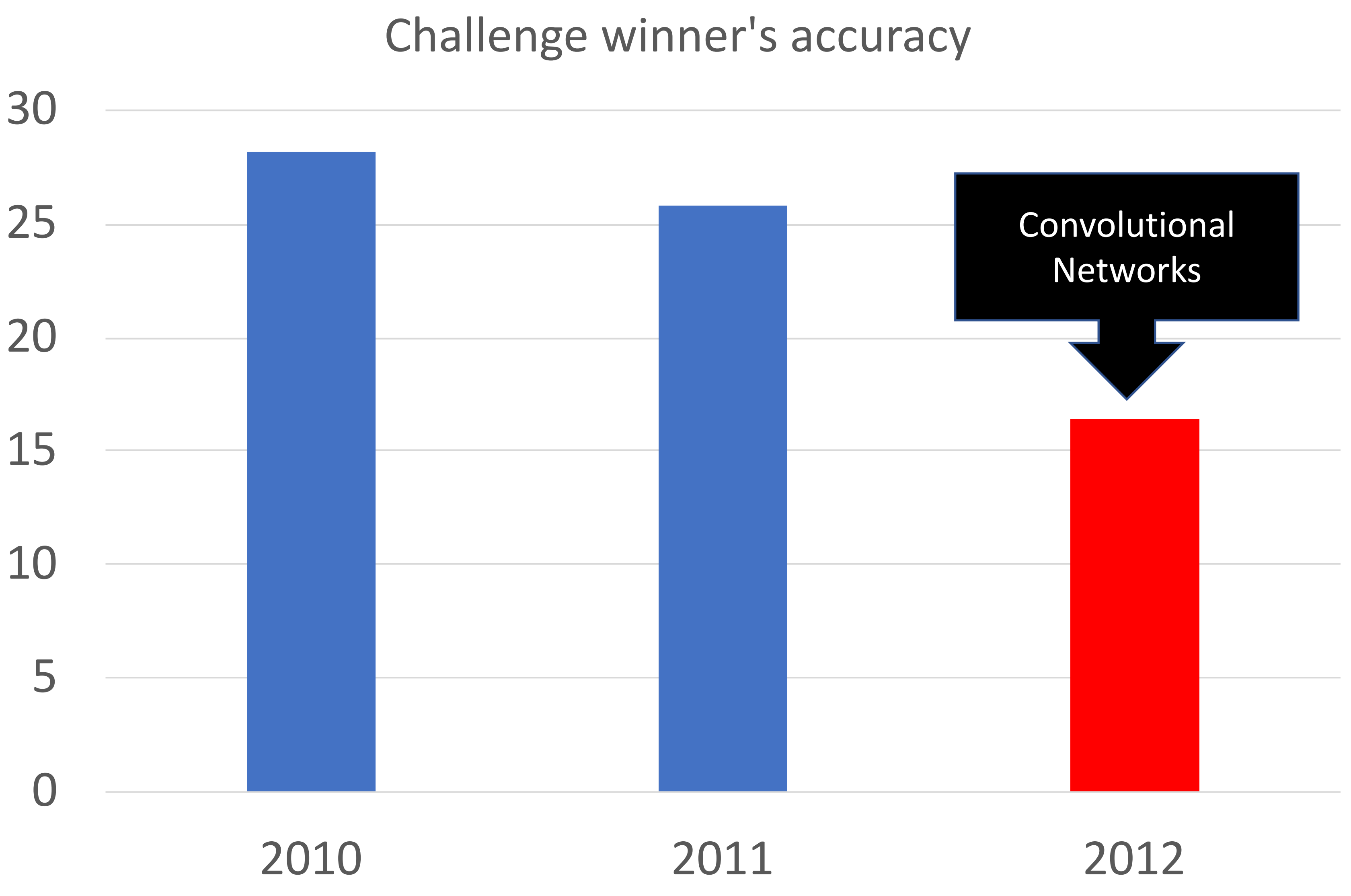


Image Classification

ImageNet

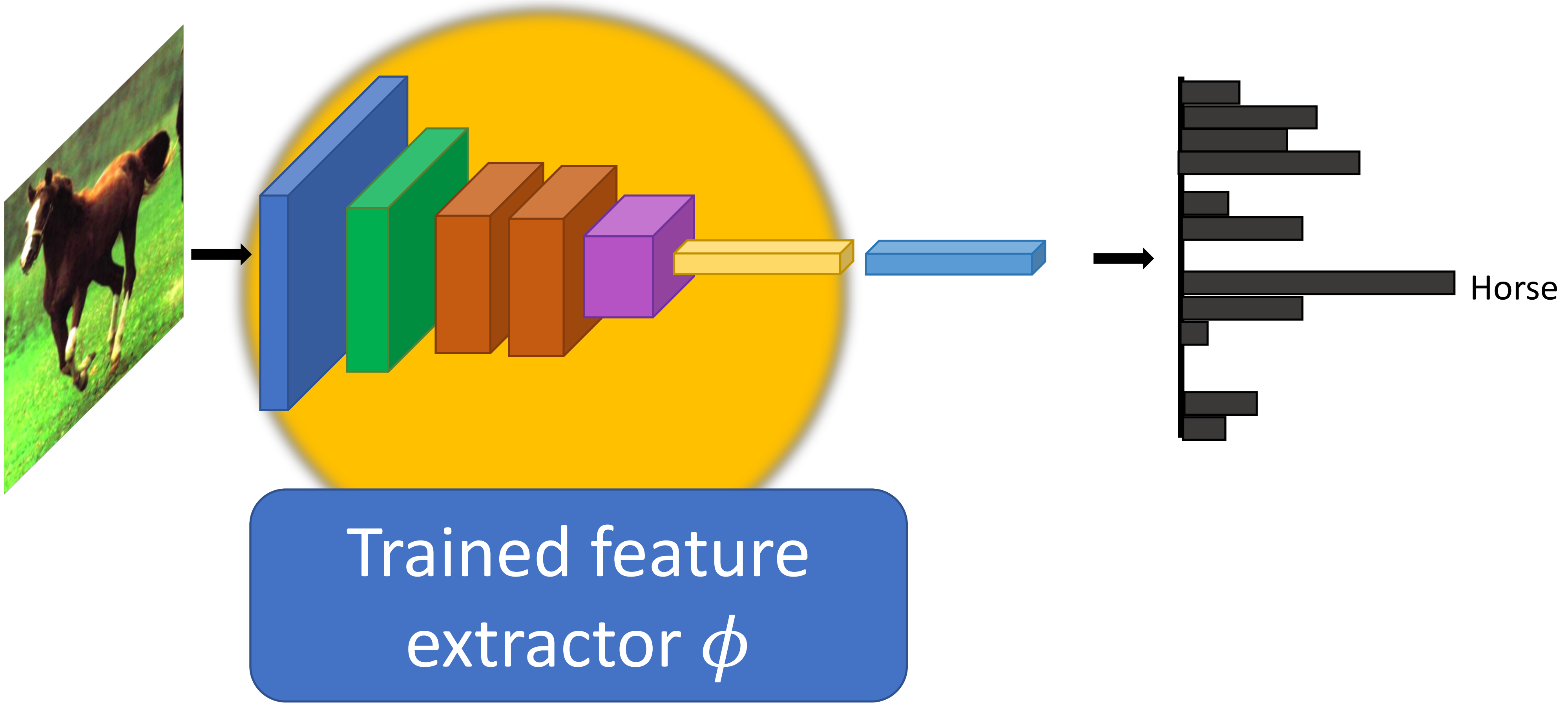
- 1000 categories
- ~1000 instances per category

Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) **ImageNet Large Scale Visual Recognition Challenge**. *International Journal of Computer Vision*, 2015.


Challenge winner's accuracy
30
25
20
15
10
5
0
Convolutional Networks
2010
2011
2012

Transfer learning

Transfer learning with convolutional networks



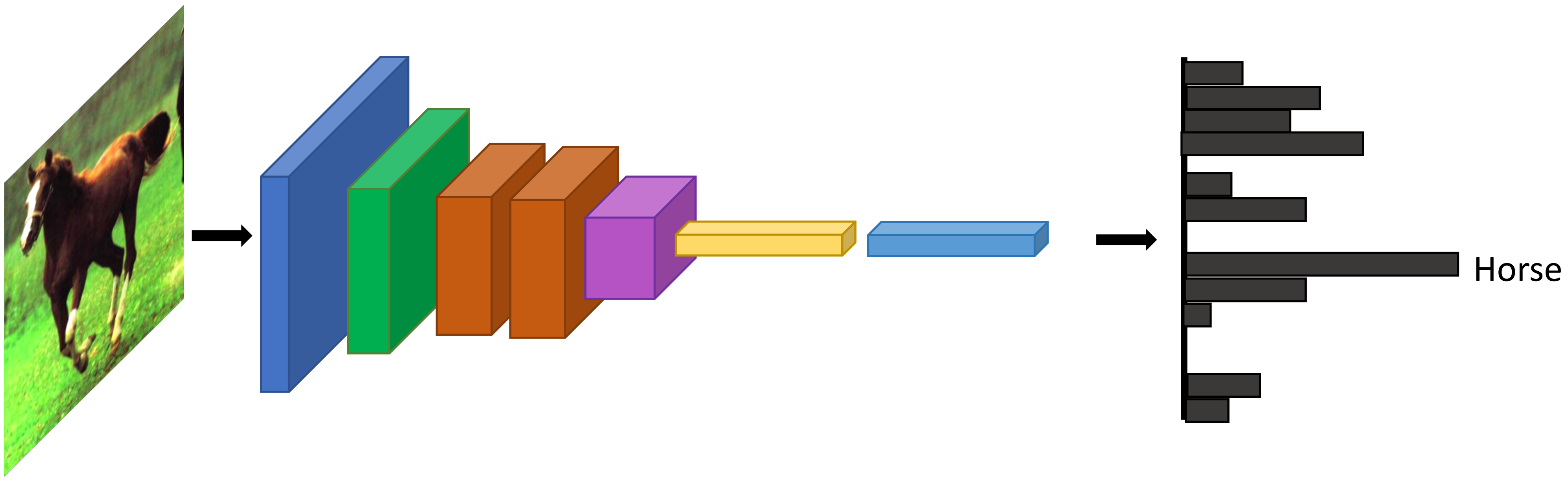
Transfer learning with convolutional networks

Dataset	Non-Convnet Method	Non-Convnet perf	Pretrained convnet + classifier	Improvement
Caltech 101	MKL	84.3	87.7	+3.4
VOC 2007	SIFT+FK	61.7	79.7	+18
CUB 200	SIFT+FK	18.8	61.0	+42.2
Aircraft	SIFT+FK	61.0	45.0	-16
Cars	SIFT+FK	59.2	36.5	-22.7

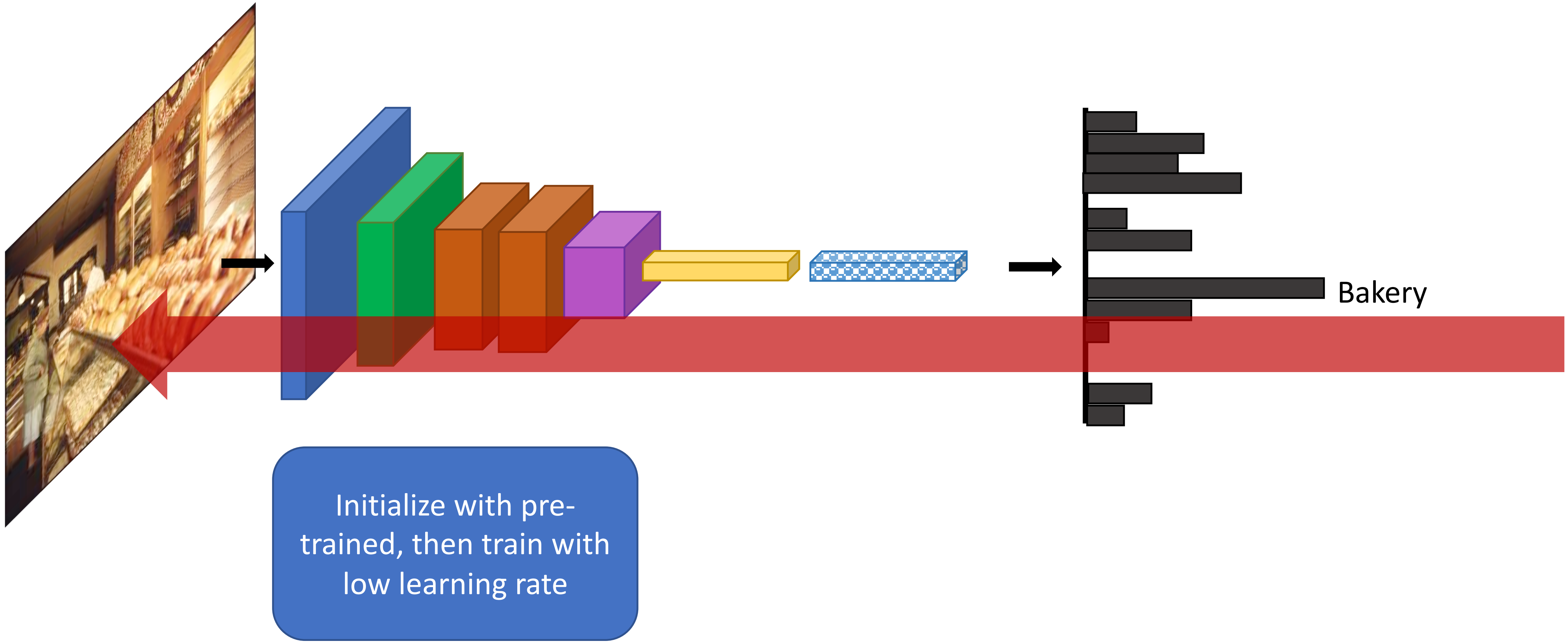
Why transfer learning?

- Availability of training data
- Computational cost
- Ability to pre-compute feature vectors and use for multiple tasks
- *Con: NO end-to-end learning*

Finetuning



Finetuning


Bakery
Initialize with pre-trained, then train with low learning rate

Finetuning

Dataset	Non-Convnet Method	Non-Convnet perf	Pretrained convnet + classifier	Finetuned convnet	Improvement
Caltech 101	MKL	84.3	87.7	88.4	+4.1
VOC 2007	SIFT+FK	61.7	79.7	82.4	+20.7
CUB 200	SIFT+FK	18.8	61.0	70.4	+51.6
Aircraft	SIFT+FK	61.0	45.0	74.1	+13.1
Cars	SIFT+FK	59.2	36.5	79.8	+20.6