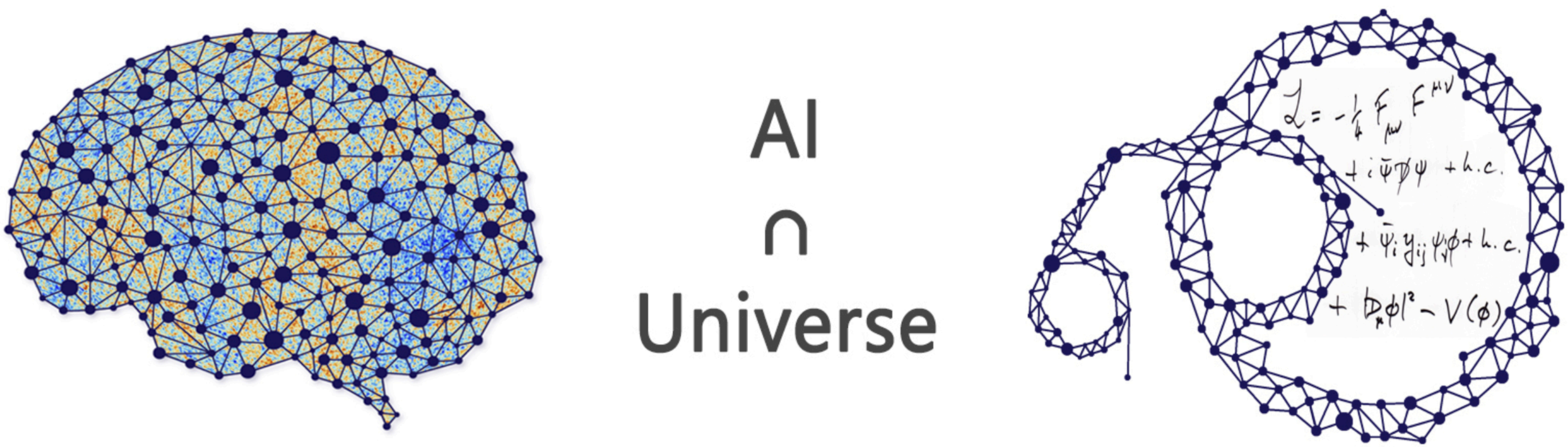
PHY 835: Machine Learning in Physics

Lecture 10: Convolutional Neural Network

February 22, 2024



Gary Shiu

Outline for today

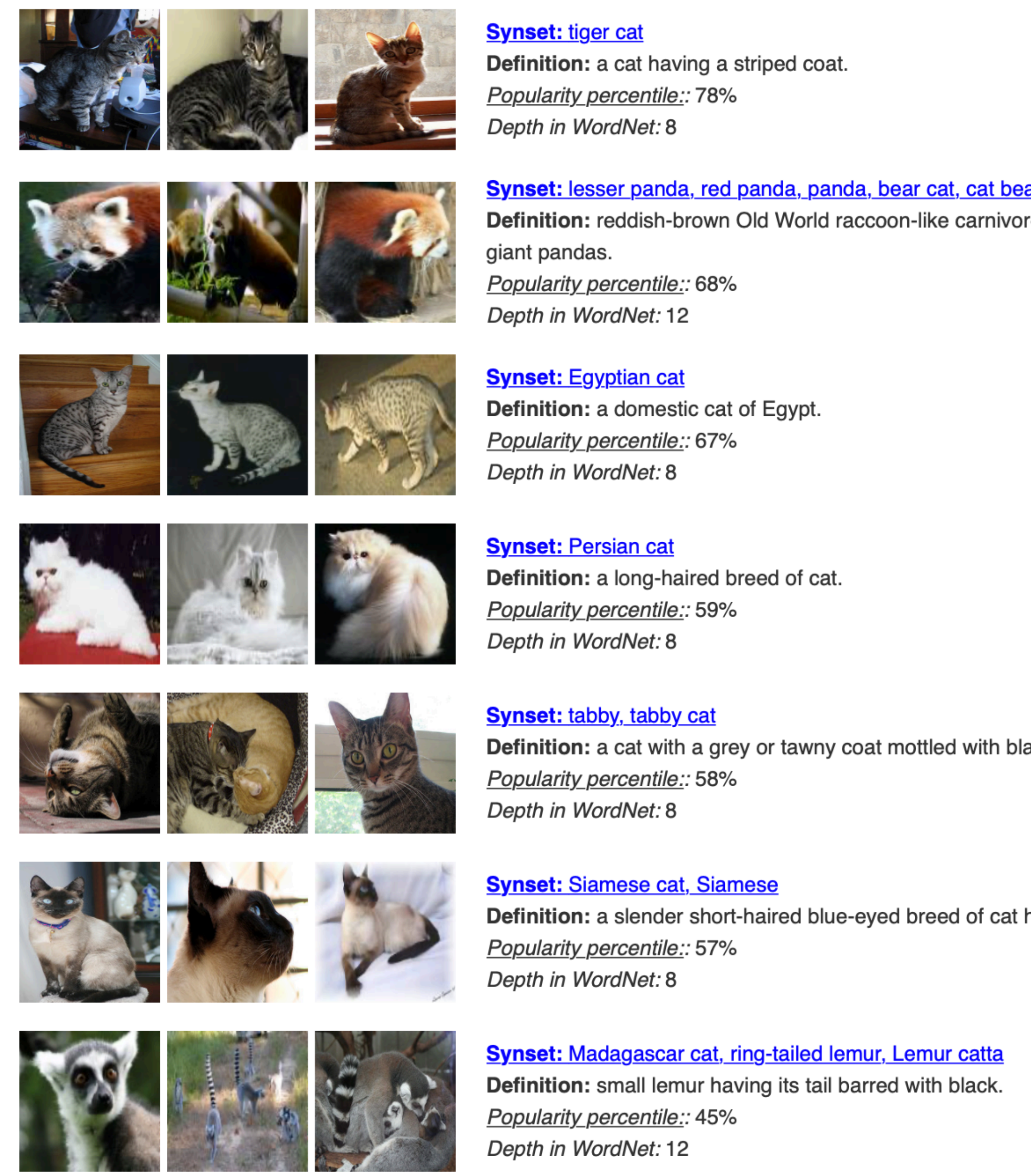
- Convolutional neural networks (CNNs)
- Convolutional Layer and Pooling layer
- Workflow for Deep Learning

References: Deep Learning Book, 1803.08823

Stanford CS23 (Andrej Karpathy & Fei-Fei Li): https://cs231n.github.io

Benchmark Datasets

- **MNIST** database: images of digits
- **ImageNet** database: $\geq 1.4 \times 10^7$ images (hand annotated), $\geq 20{,}000$ categories (e.g. screwdriver, each with $\mathcal{O}(1000)$ examples), http://image-net.org
- Performance of algorithms measured on benchmark datasets.
- Other datasets for different problems (e.g. 3D object recognition, Language: Wordnet)
- Think of some physics examples?

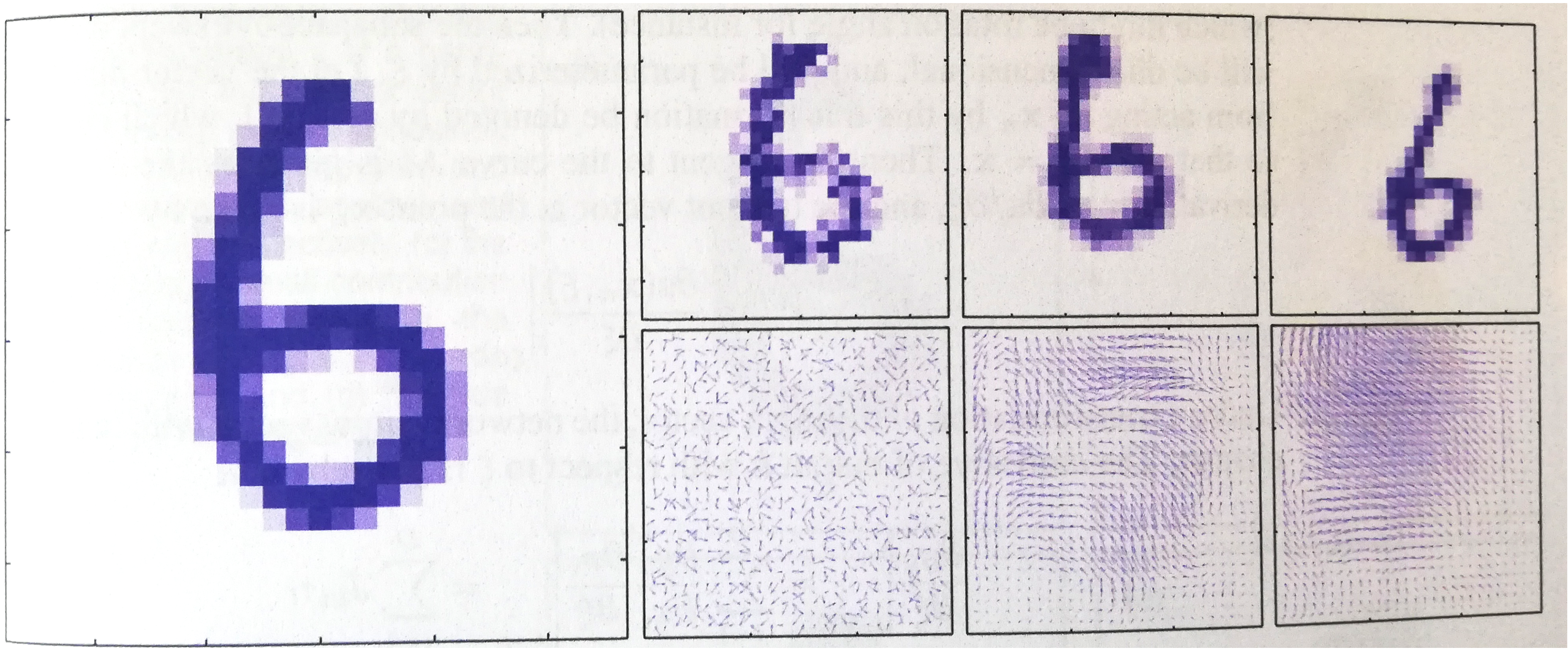


Learning with Symmetries

- **Locality:** features that define a "cat" are local in the picture: whiskers, tail, paws, …
- **Translational invariance:** Cats can be anywhere in the image.
- **Rotational invariance:** Relative position of features must be respected (e.g. whiskers and tail should appear on opposite sides)
- Our classifier should exhibit all these high-level structures.

Learning with Symmetries

- Consider classification of digits:

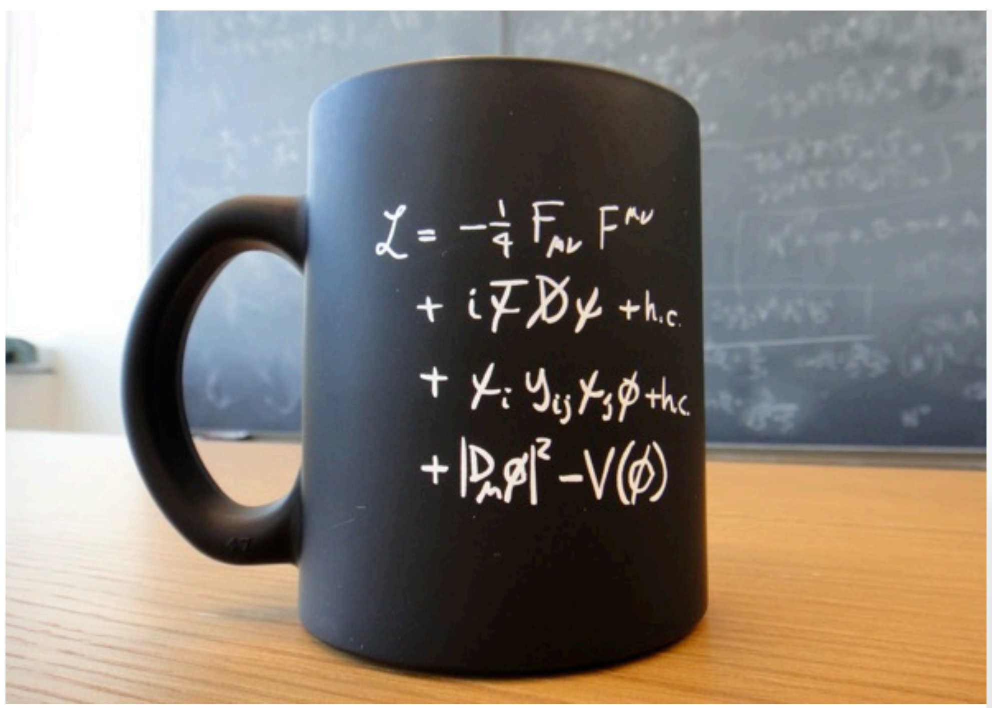


- What symmetries should be built-in in ML classifiers?

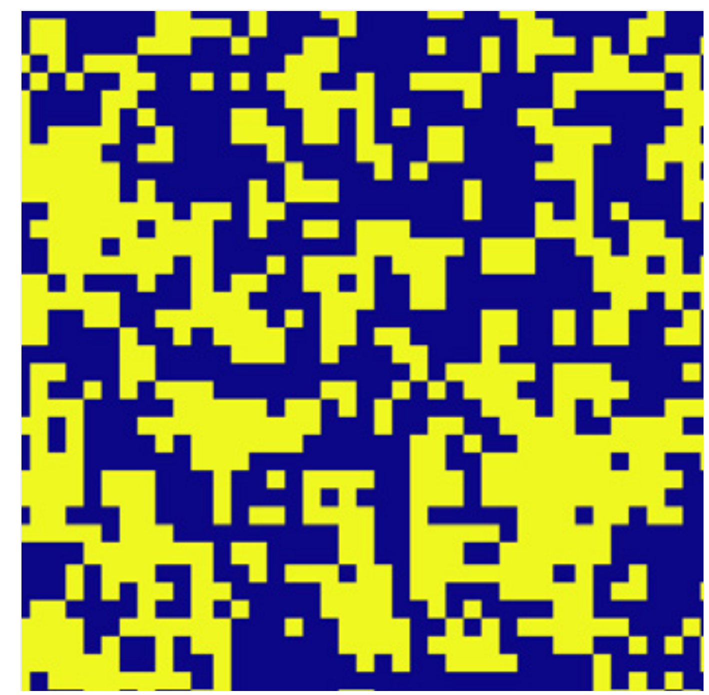
Translation, scaling, small rotations, smearing, elastic deformations.

Locality and Symmetries

- **Locality** & **Symmetries**: basic principles underlying physical laws.
- Physics is governed by **local interactions**. Think about QFT, relativity, and statistical physics:



Vision tasks: local features matters, e.g.. whiskers, edge of a table, …



$$H = -J \sum_{\langle ij \rangle} S_i S_j$$

Locality and Symmetries

- **Symmetries** are at the heart of physics. For example, translation invariance allows to work in momentum space→less parameters
- In relativity and quantum field theory, Poincare-symmetry (translations, rotations, boosts) is essential.
- Gauge symmetries are ubiquitous in QFT and gravity. Equivariant CNNs (Cohen, Welling 2016). We will come back to this...
- $f(x)$ is equivariant if we change the input in a particular way as $x' = g \cdot x$, the output changes in the same way: $f(g \cdot x) = g \cdot f(x)$:

$$\begin{tikzcd} X \arrow[r, "g\cdot"] \arrow[d, "f"'] & X \arrow[d, "f"] \\ Y \arrow[r, "g\cdot"'] & Y \end{tikzcd}$$

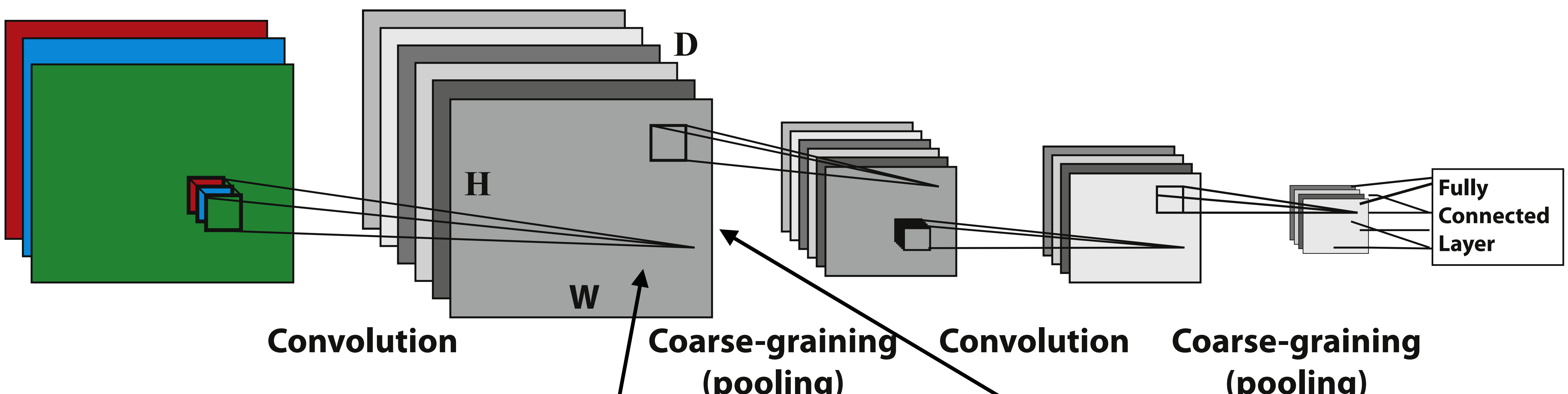
Convolutional Neural Networks

- The simplest approach would be to input the images to a **fully connected NN** which given enough training data (and time) would **learn the symmetries** by example.
- However, a crucial property is ignored: **nearby pixels are strongly correlated** we should aim instead first to **identify local features** that depend on small subregions.
- For example, treating the spin configuration of the 2d Ising model as a $L \times L$ dimensional vector ($L =$ number of sites in each linear direction) throws away spatial information (e.g., domain wall)
- Convolutional Neural Networks (CNNs) are architectures that t**ake advantage of this additional high-level structures** that all-to-all coupled networks fail to exploit.

Convolutional Neural Networks

A CNN is a translationally invariant neural network that respects locality of the input data.

Depth: number of input channels
(not depth of neural network)



D=3 for RGB images
Height (H) and Width (W)
determined by # of pixels

pooling layers reduce H, W
while preserving D

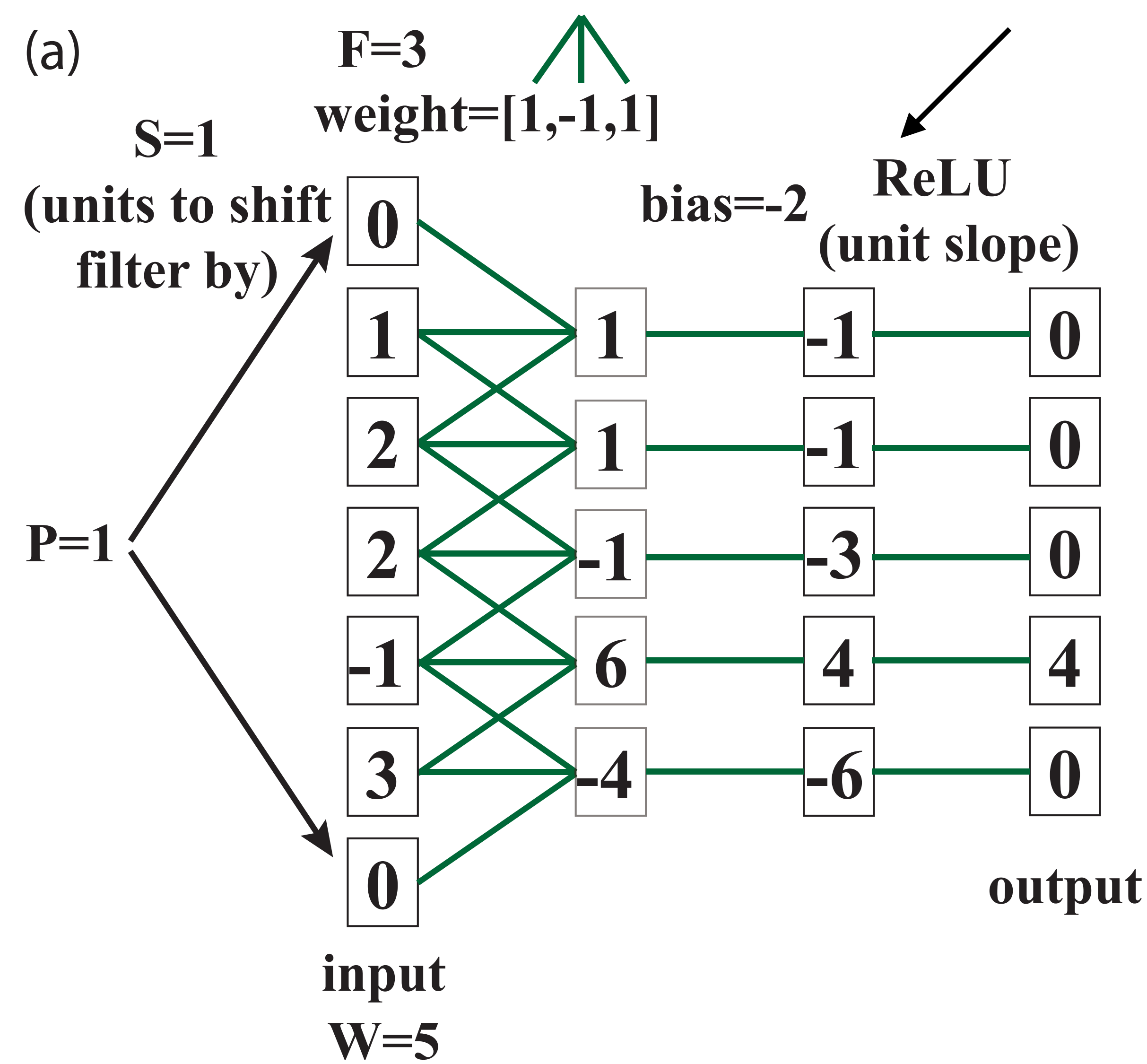
neuron activation state:
convolution with local spatial filter
(e.g., 3 x 3 pixel grid)

Convolutional Neural Networks

CNNs are composed by **two** kinds of layers

Convolution of input with filters

non-linearity



example of convolutional layer

e.g. $0 \times 1 + 1 \times (-1) + 2 \times 1 - 2 = -1$

Convolutional Neural Networks

another example
(with no padding)

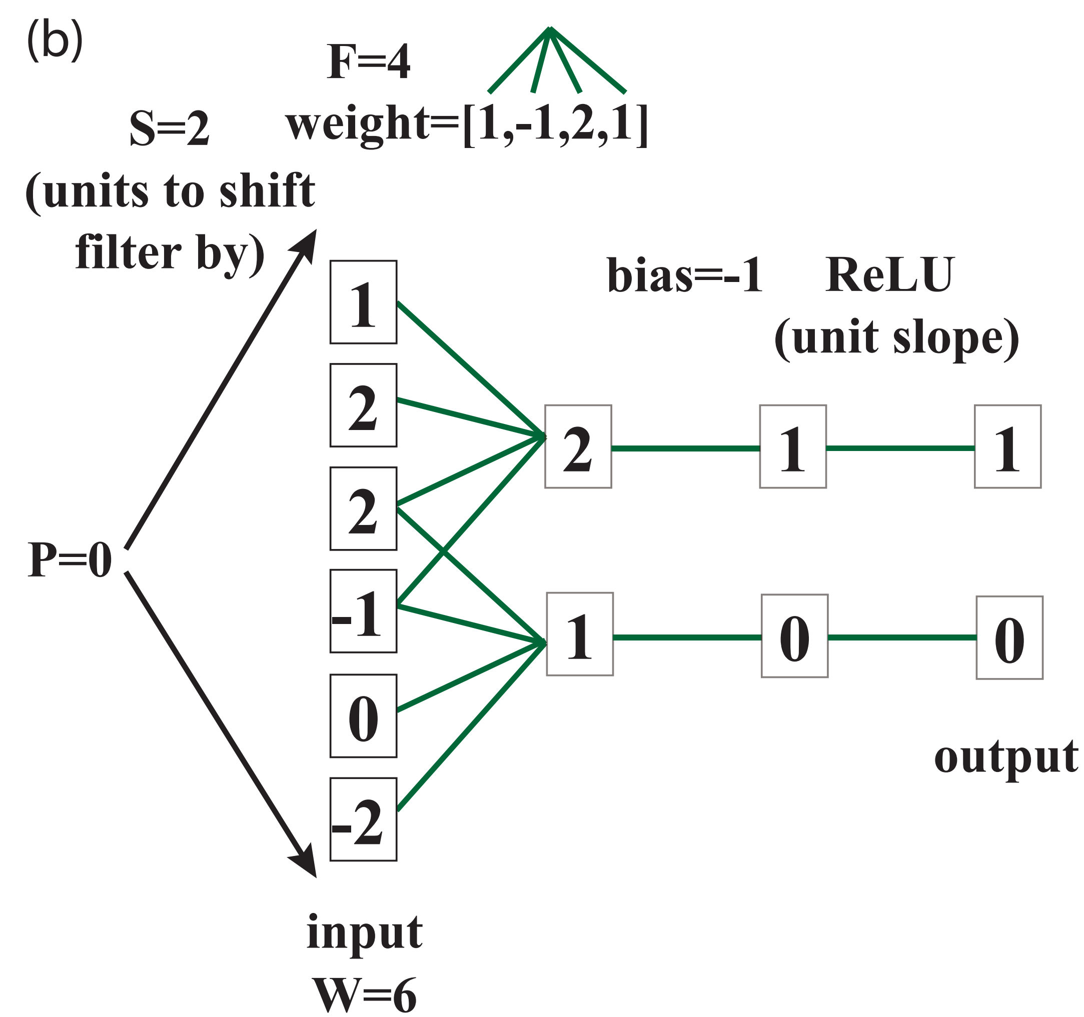
F=receptive field size of the Conv Layer neurons

S=stride

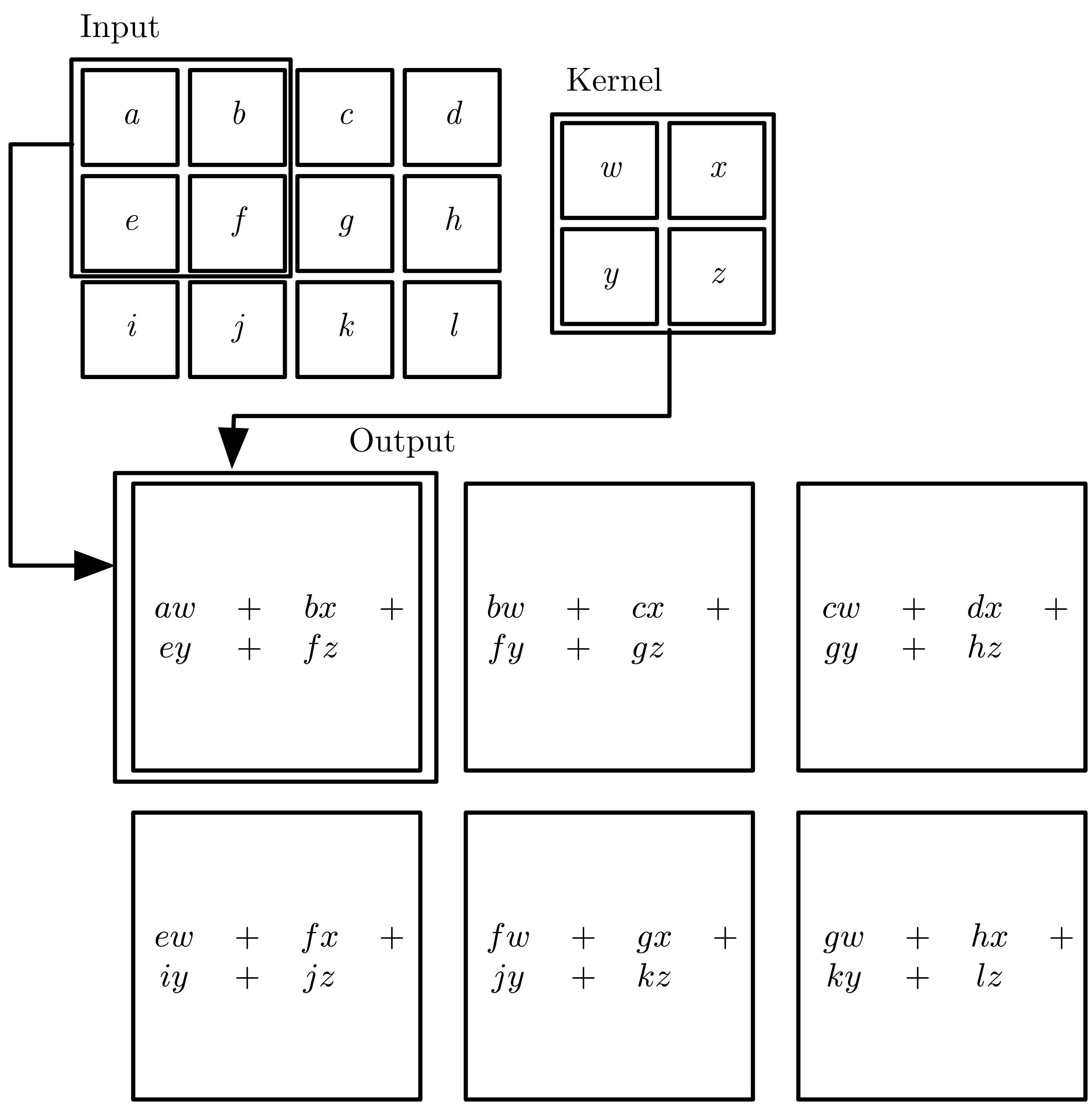
P=amount of zero padding on the border

Number of neurons (outputs) in the layer:

(W-F+2P)/S+1



2d Convolution



The Convolution Operation

For example, estimate the position of a spaceship from several measurements:

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

This operation known as **convolution** is denoted by an asterisk:

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

output
(feature map)
input
kernel

In a data set, “time” is discretized:

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

Convolutional Neural Networks

CNNs are composed by **two** kinds of layers

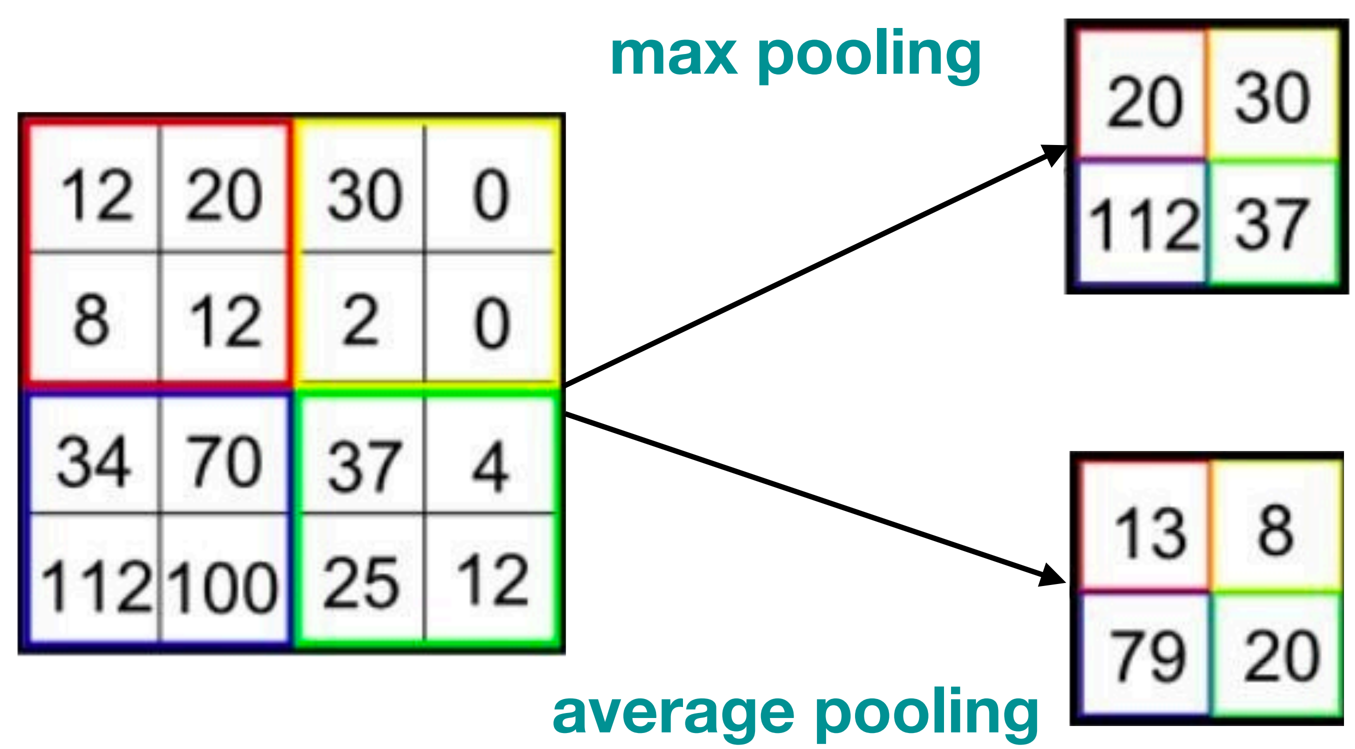
Convolution of input with filters

Pooling layer that coarse-grains the input while maintaining locality and spatial structure

~ decimation in RG

reduces the dim. of outputs (depth is kept fixed)

In this example, by pooling over 2x2 blocks, H and W are reduced by half.



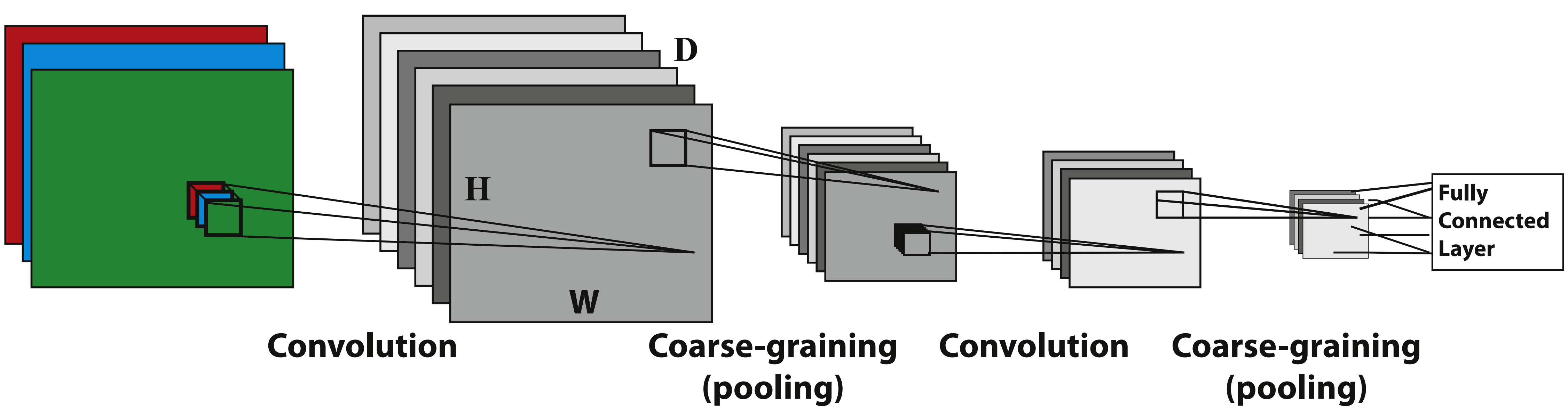
Convolutional Neural Networks

CNNs are composed by **two** kinds of layers

Convolution of input with filters

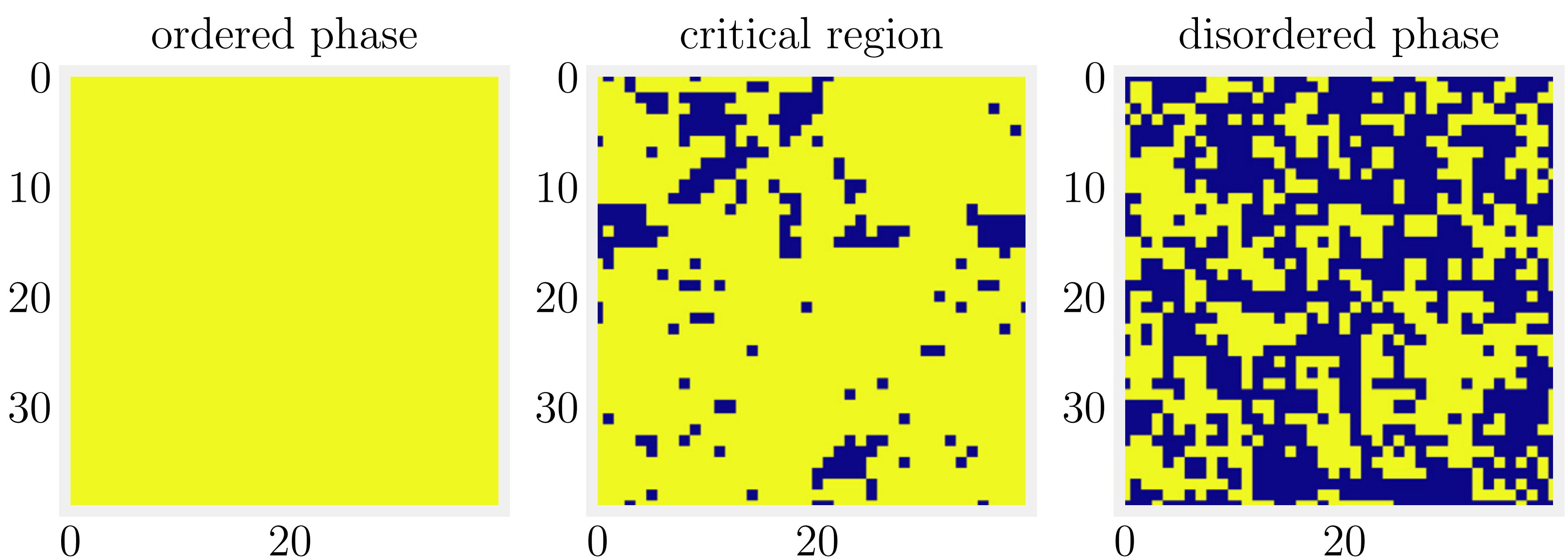
Pooling layer that coarse-grains the input while maintaining locality and spatial structure

These layers are followed by an **all-to-all connected layer and a high-level classifier**, so that one can train CNNs using the standard backpropagation algorithm:



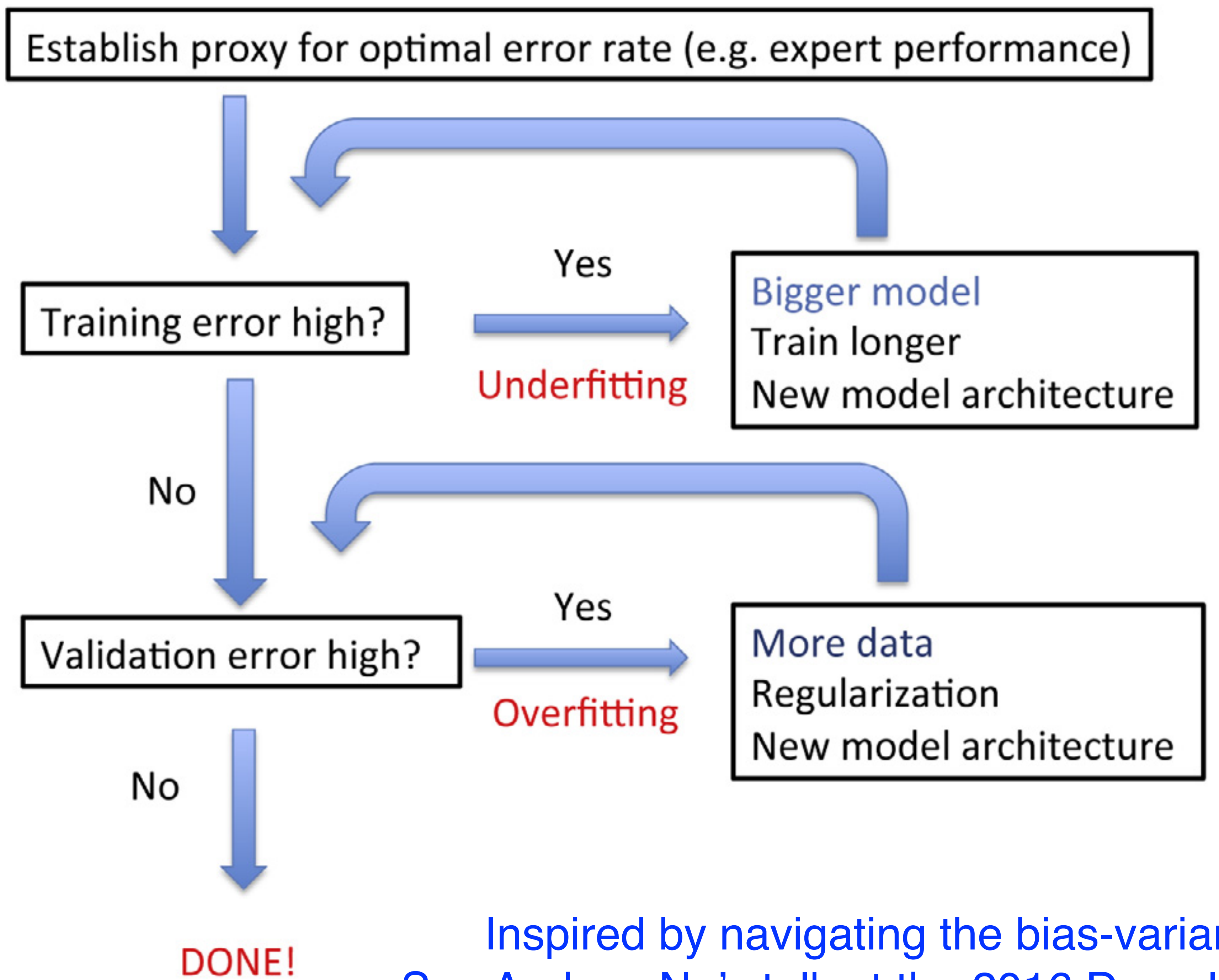
Convolutional Neural Networks

- Significantly reduce the number of parameters: determined by the number and size of the filters. Further reduced by pooling.
- Only problems characterized by spatial locality are amenable to CNNs, e.g., Ising model and MNIST but not SUSY datasets.



- See Notebook 14: Pytorch CNN (Ising); MNIST example (later).
  http://physics.bu.edu/~pankajm/ML-Notebooks/HTML/NB14_CX-CNN-ising-Pytorch.html
- Can you think of the types of datasets in particle physics and cosmology that are amendable to CNNs?

Workflow for Deep Learning



Inspired by navigating the bias-variance tradeoff.
See Andrew Ng's talk at the 2016 Deep Learning School:
https://www.youtube.com/watch?v=F1ka6a13S9I