Convolutional Neural Nets

Exploiting stationarity, locality, and compositionality of natural data

Input layer / samples

$$\mathcal{X} = \{x^{(i)} \in \mathbb{R}^n \mid x^{(i)} \text{ is a data sample}\}_{i=1}^{m}$$ input samples

We don't want pixels in images defined as points in an NxN dimensional space

(eg: 1 Mega Pixel = 1 millon dimensional space)

because all data points are going to be really close in this huge dimensional space, and to separate them, we'll want to make this space bigger, which is computationally very expensive.

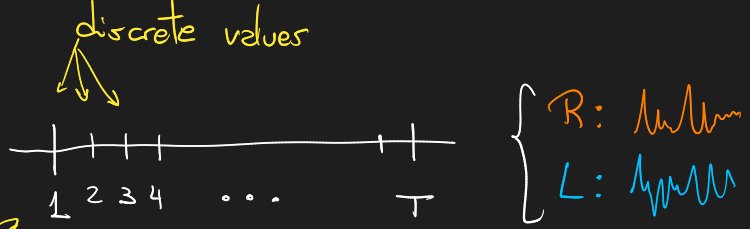
Instead, we can define the data points as functions

$$\mathcal{X} = \{x^{(i)} : \underset{\text{domain}}{\Omega} \to \underset{\text{channels}}{\mathbb{R}^c}, \omega \mapsto x^{(i)}(\omega)\}_{i=1}^{m}$$

Some examples

- Audio

$$\Omega = \{1, 2, \cdots, \underset{\text{sampling interval}}{T/\Delta t}\} \subset \mathbb{N}, \quad c \in \{\underset{\text{mono}}{1}, \underset{\text{Dolby 5.1}}{2, 5+1}, \cdots\}$$ total time, stereo



1 dimensional signal

2 channels (stereo)

- Images

$$\Omega = \{1, \cdots, \underset{\text{height}}{h}\} \times \{1, \cdots, \underset{\text{width}}{w}\} \subset \mathbb{N}^2, \quad c \in \{\underset{\text{colour}}{1}, 3, \underset{\text{hyperspectral}}{20}, \cdots\}$$ grey scale

ordered!

$$x(\omega_1, \omega_2) = \begin{pmatrix} r(\omega_1, \omega_2) \\ g(\omega_1, \omega_2) \\ b(\omega_1, \omega_2) \end{pmatrix}$$

• Physics

four-momentum

$$\Omega = \mathbb{R}^4 \times \mathbb{R}^4, \quad c = 1$$

space-time — Hamiltonian

1D signal

$$x[k] = \begin{pmatrix} x[1] & x[2] & \cdots \end{pmatrix}$$

Signals can be represented as vectors

$\boldsymbol{x} = [x_1 \;\; x_2 \;\; x_3 \;\; \ldots \;\; x_t \;\; \ldots]^\top$

x_t are waveform heights

$\boldsymbol{x} = [x_{11} \;\; x_{12} \;\; \ldots \;\; x_{1n} \;\; x_{21} \;\; x_{22} \;\; \ldots]^\top$

x_{ij} are pixel values

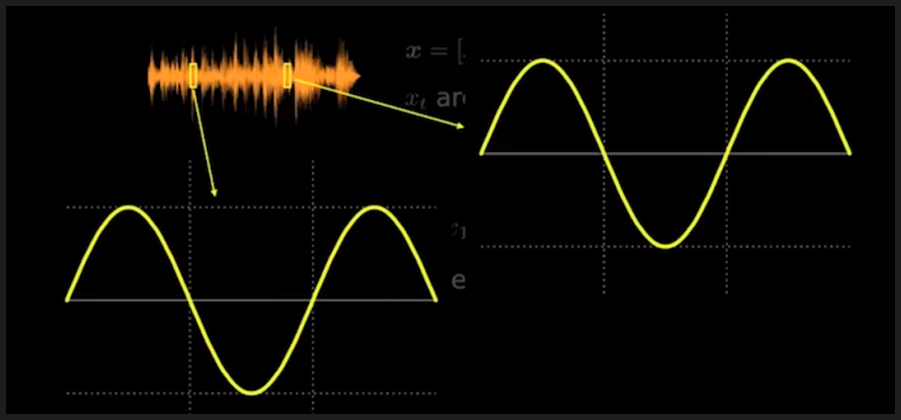
"John picked up the apple"

$\boldsymbol{x} = [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5]^\top$

x_t are one-hot vectors

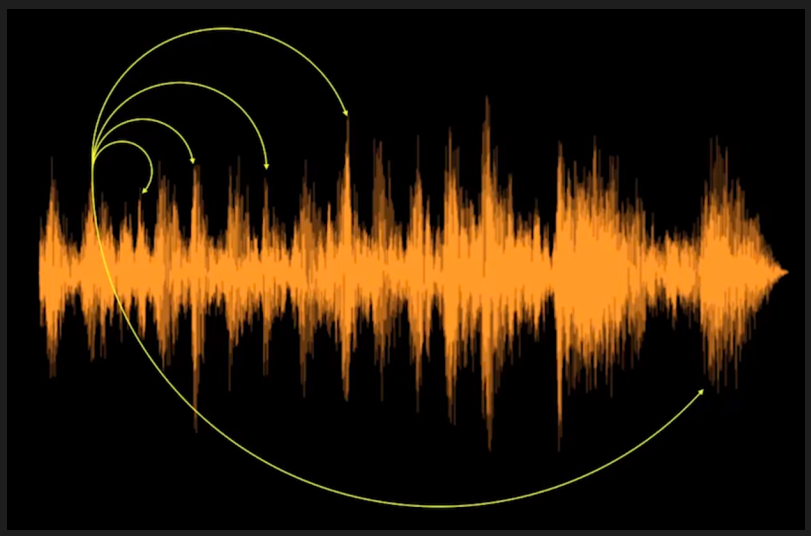
Stationarity:

• Patterns repeat



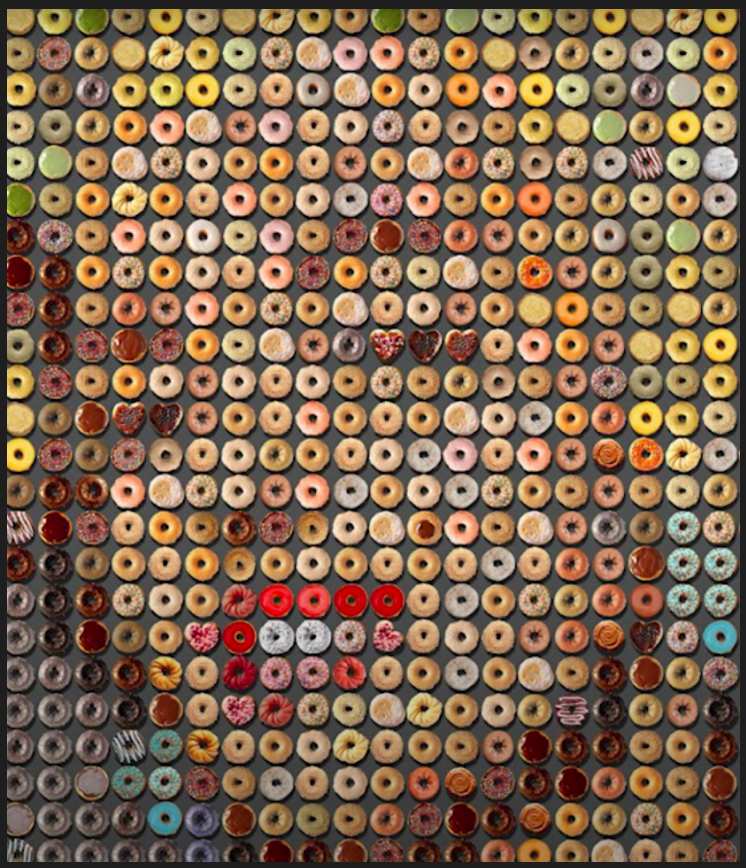
Locality:

- Strong correlation when closer.



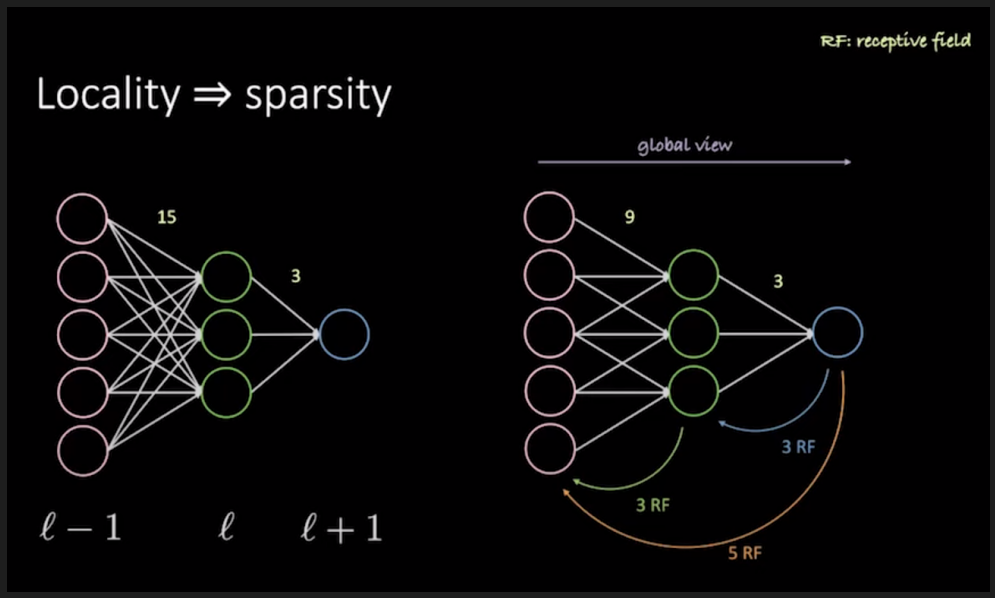
Compositionality

- Meaning comes from hierarchical structure



Review:

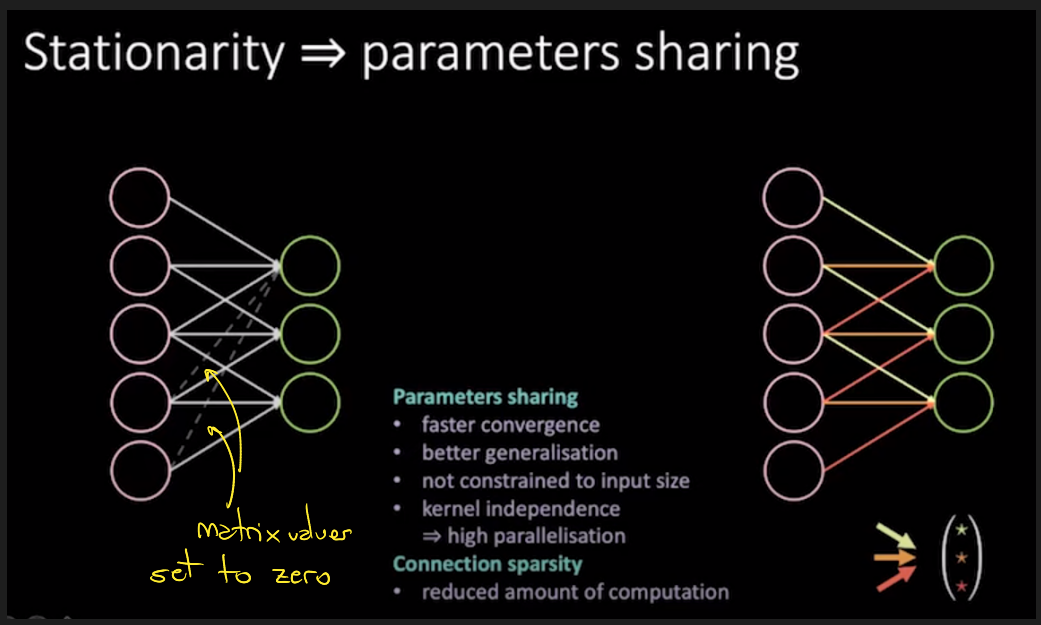
- CNNs work best when input has:
  - ↳ Locality (Close ≡ Similar)
  - ↳ Stationarity (Patterns repeat)
  - ↳ Compositionality (Big picture = Small pieces together)



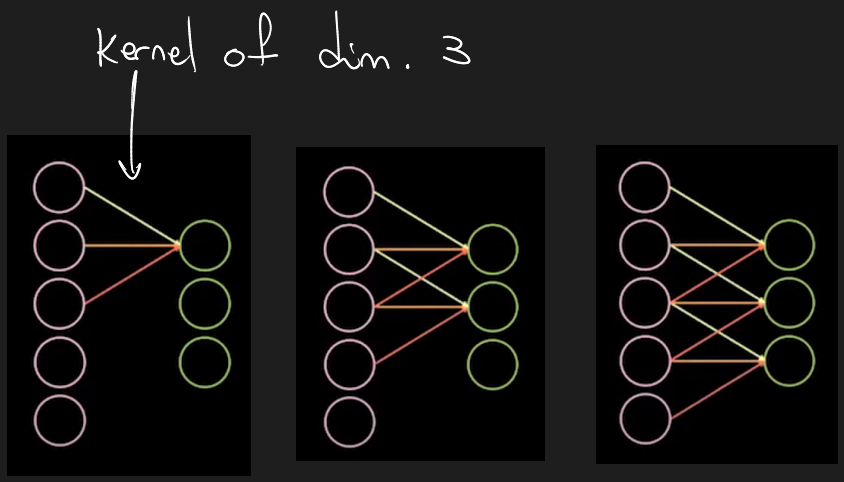
Recall

NNs : { Rotations
Squashing

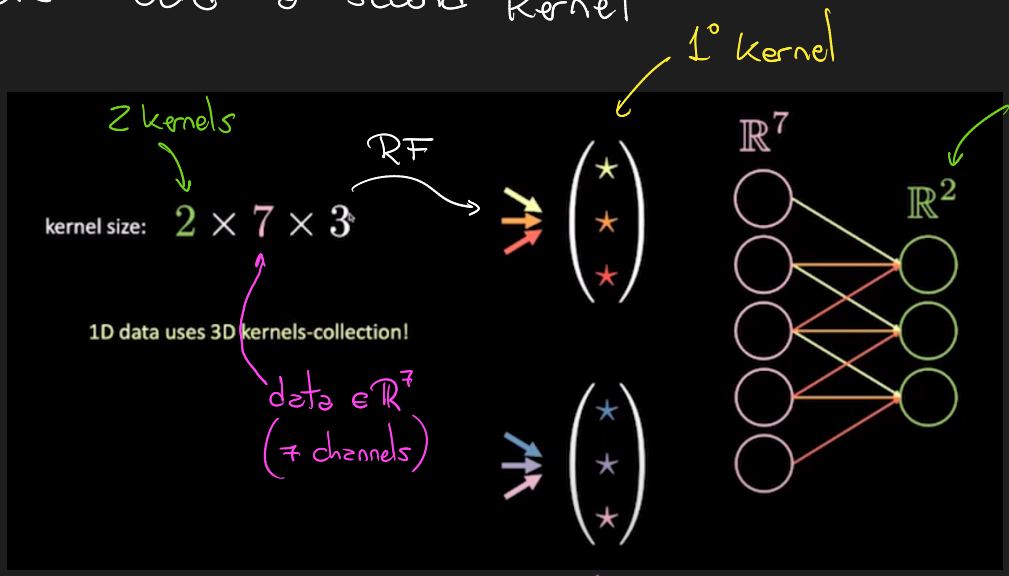
So CNN do these two only over the RF



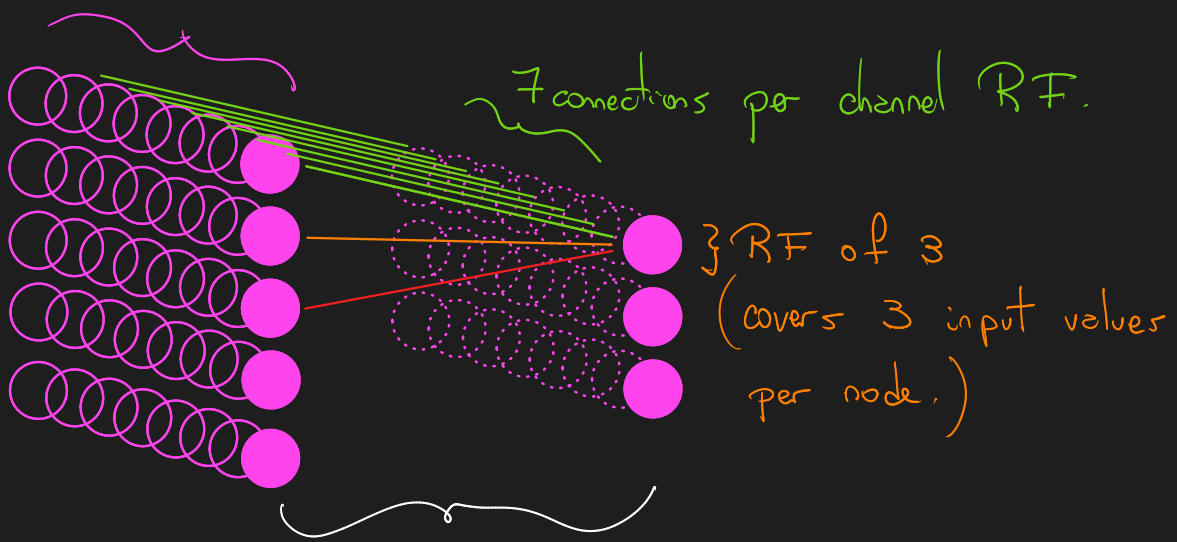
Kernels - 1D data



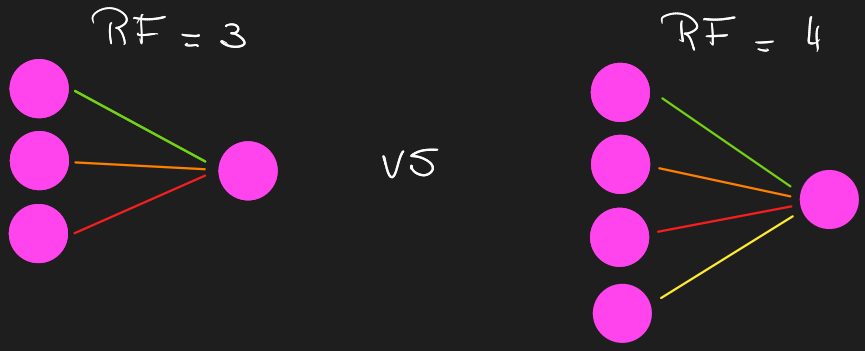
Lets add a second kernel

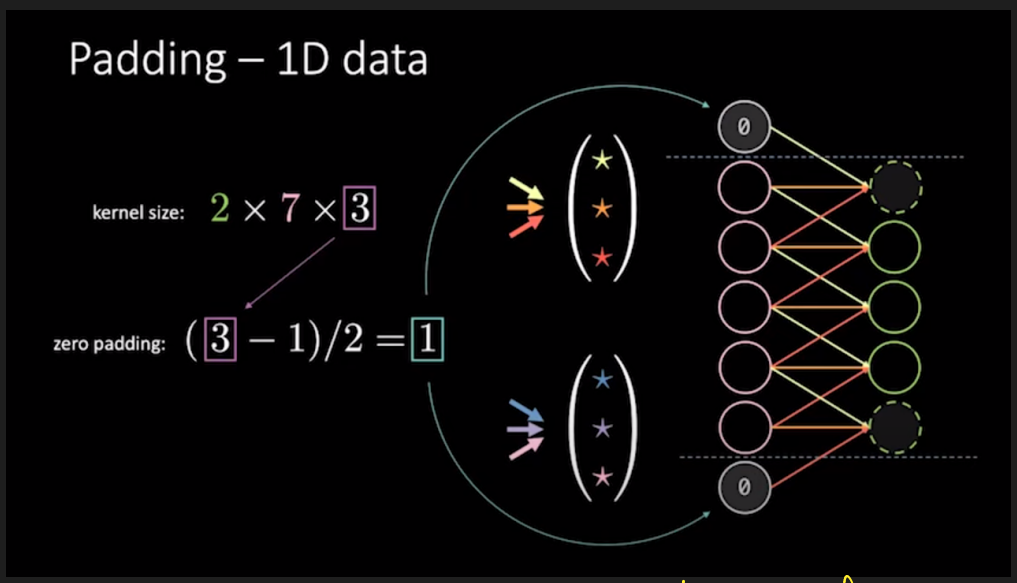


Input has 7 channels

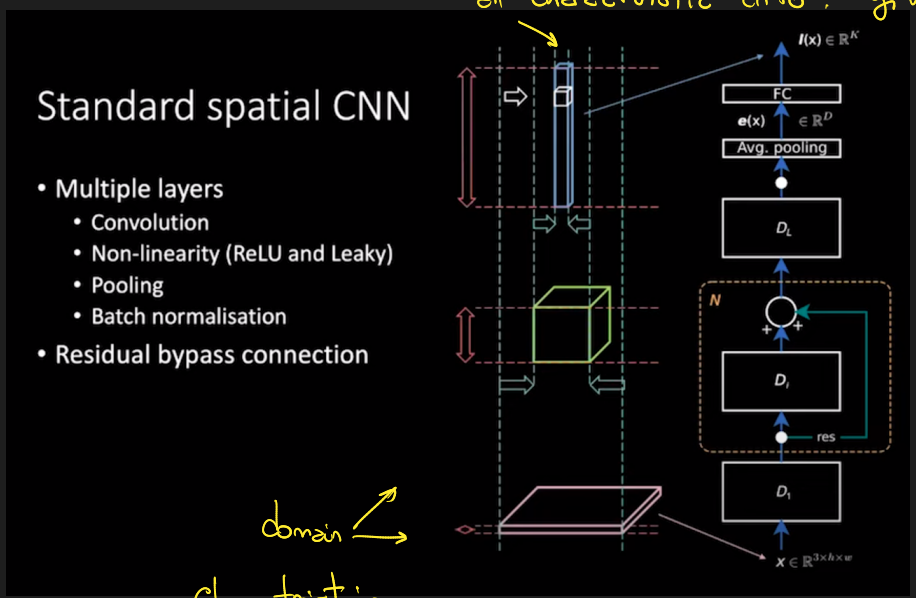


RF usually is odd, so the center correspond to a single value, and not 2





all characteristic info: given location tell us, eg, the class.

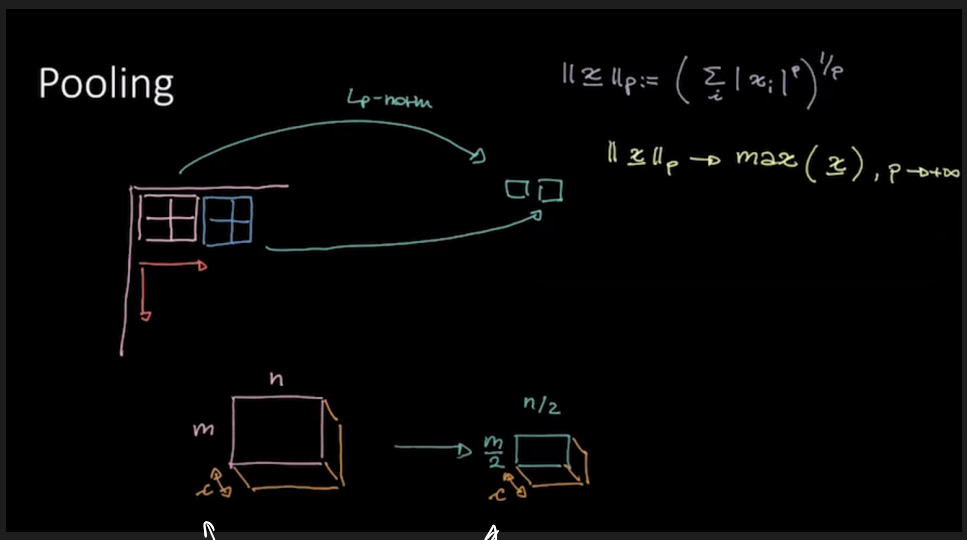


domain

Characteristic information

(in this case, the color of the pixel)

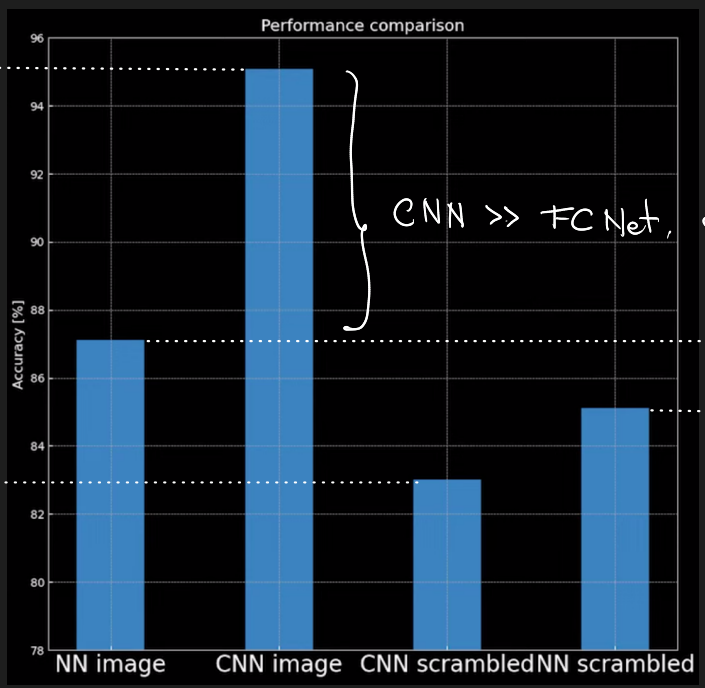
Network: Reduction in domain, increasing characteristic information,



channels after pooling stays the same.

Notebook demo:

Alf shows how when permuting the pixels of the image (in a deterministic way), we loss all advantages of the CNN priors, resulting in similar performance to a Fully Connected Network, which perform exactly the same for both input cases.



Big improvement when data has:
- locality
- stationarity
- compositionality

CNN >> FC Net, on "good" data ⊛

statistical fluctuations

(FC) (FC)

⊛ where "good" means it has the 3 properties