Convolutional Neural Networks

- They are also made up of "neuron", and they have differentiable weights an biases.

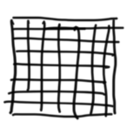
- Each neuron receives some inputs, performs a dot product, (optionally) followed by a non-linearity.

- Fwd pass and loss are fully differentiable.

CNNs: Are tailored to address the unfavorable complexity of MLPs for high-dimensional inputs/outputs, by making the explicit assumption that our data "lives" on grid (e.g. images, time-series).
This assumption allows us to encode certain structure in our NN archi.

e.g.

x $\quad\quad\quad\quad$ y

32  $\longrightarrow$ $\{0,1\}$ $\quad\quad$ $3 \times 200 \times 200 = 120{,}000$ weights

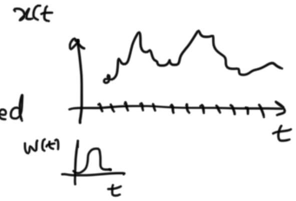
32

input

$\Longrightarrow$ ConvNets (CNNs) are neural networks that use convolution instead of matrix multiplication/projection, in at least one of their layers

Convolution: Definition in 1D

Assume a time-series $x(t)$ (noisy) that is sampled at regular intervals.



To filter out the noise we will compute some weighted average of the measurements. To do so, we will choose a weighting function $w(s)$, and obtain the smoothed signal as:

$$S(t) = \int^{\infty} x(s)\, w(t-s)\, ds = \int^{\infty} \ldots$$

$$\int_{-\infty} \ldots = \int_{-\infty} w(s)\,x(t-s)\,ds$$

or $\quad s(t) = (x * w)(t)$ $\quad$ convolution is a commutative linear operation.

Discrete convolution in 1D:

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

input

→ x : $\boxed{x_1 \mid x_2 \mid x_3 \mid x_4}$

conv. kernel

→ $\underline{w}$: $\boxed{w_1 \mid w_2}$

feature map

→ s : $\boxed{s_1 \mid s_2 \mid s_3}$

$$\left.\begin{aligned} s_1 &= w_1 x_1 + w_2 x_2 \\ s_2 &= w_1 x_2 + w_2 x_3 \\ s_3 &= w_1 x_3 + w_2 x_4 \end{aligned}\right.$$

Discrete convolution for a 2D image ($m \times n$ pixels)

- $S(i,j) = \sum_m \sum_n I(m,n)\,K(i-m,\, j-n) = \sum_m \sum_n K(m,n)\,I(i-m,\, j-\ldots$