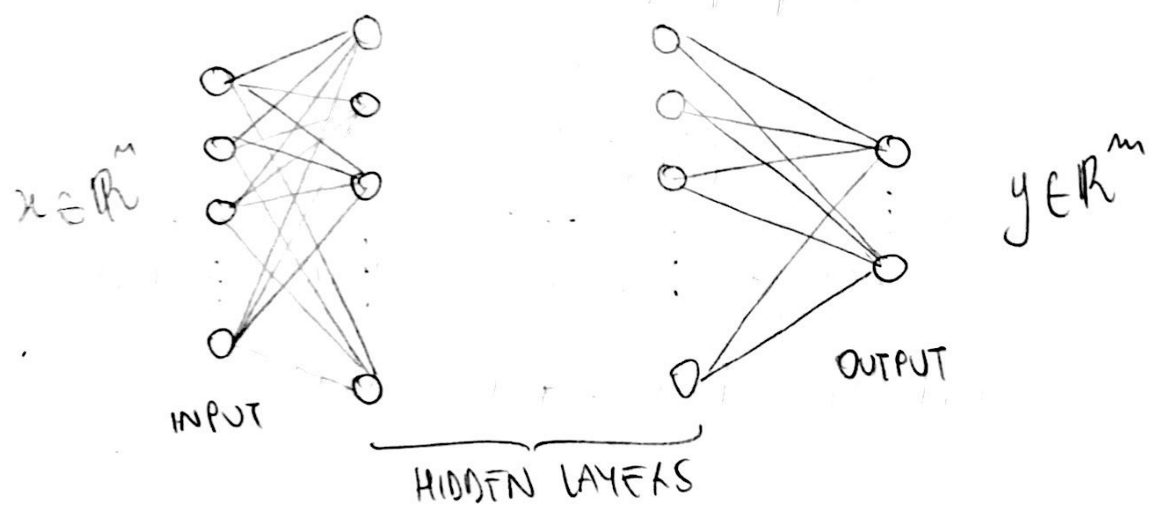
ARTIFICIAL NEURAL NETWORKS

HUBEL + WIESEL → NOBEL PRIZE.
1980 - NEOCOGNITRON : FIRST MODEL TO SIMULATE THE HIERARCHICAL STRUCTURE OF THE CAT VISUAL CORTEX

2012 - ImageNet : REVOLUTIONIZE WHAT WE THINK ABOUT OF NEURAL NETWORKS. → HUGE DATASET

THEY INVOLVE OPTIMIZATION AT VERY LARGE SCALE.



$$y = f(x, \beta)$$

COMPUTER VISION → INPUT IS AN IMAGE

WE CAN SE THE MAP BETWEEN INPUTS AND FIRST LAYER AS $x^{(1)} = A_1 x$ AND SIMILARLY FOR ALL THE NEXT LAYERS.

$$\left.\begin{aligned} x^{(1)} &= A_1 x \\ x^{(2)} &= A_2 x^{(1)} \\ &\vdots \\ y &= A_M x^{(M-1)} \end{aligned}\right\} \text{NESTED STRUCTURE}$$

OR MORE COMPACTLY:

$$y = A_M \dots A_2 A_1 x$$

IF ALL LINEAR:

$$\bar{A} = A_M \ldots A_2 A_1$$

$$y = \bar{A} x$$

THE BEST WE CAN DO IS JUST A LINEAR MAPPING. WE NEED TO INTRODUCE NONLINEARITY:

$$x^{(1)} = f_1(A_1, x)$$
$$x^{(2)} = f_2(A_2, x^{(1)})$$
$$\vdots$$
$$y = f_M(A_M, x^{(M-1)})$$

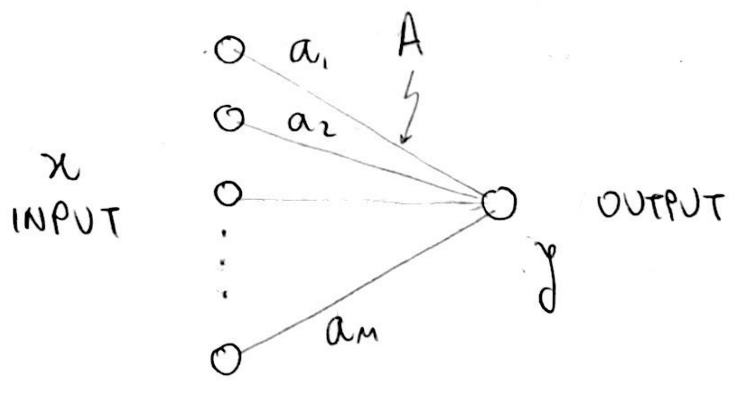
THESE OPERATIONS WILL NOT COLLAPSE TO A SINGLE MATRIX:

$$y = f_M(A_M, \ldots f_2(A_2, f_1(A_1, x)))$$

LET'S CONSIDER AN EXAMPLE

x - dog / cat picture 64×64 images

y - $\{\text{dog}, \text{cat}\} = \{+1, -1\}$



$$x = [x_1 \; x_2 \ldots x_M] \qquad y = [+1, -1, \ldots, +1]$$

$$Y = AX$$

$$\begin{bmatrix} +1 & \dots, & -1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & \dots & a_m \end{bmatrix} \begin{bmatrix} | & & | \\ x_1 & \dots & x_m \\ | & & | \end{bmatrix}$$

THE EASIEST WAY TO COMPUTE THE WAY IS WITH PSEUDOINVERSE.

THE SYSTEM IS OVERDETERMINED

IN THIS EXAMPLE THE ACTIVATION FUNCTION WAS THE IDENTITY:

$$F(x) = x$$

NONLINEAR ACTIVATION FUNCTIONS ARE MORE POWERFUL:

- $F(x) = \begin{cases} 0 & x \leq 0 \\ 1 & x > 0 \end{cases}$ BINARY STEP

CONTINUOUS VERSION OF

- $F(x) = \dfrac{1}{1 + e^{-x}}$ LOGISTIC

- $F(x) = \tanh(x)$ HYPERBOLIC TANGENT

- $f(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$ RELU

RELU HELP TO PROPAGATE GRADIENT WHEN $x \to \pm\infty$ SINCE LOGISTIC AND TANH TENDS TO BE FLAT IN THESE REGIONS.

AN ANN IS A NONLINEAR MAPPING FROM INPUTS TO OUTPUTS WITH A BUNCH OF PARAMETERS AND A COMPOSITIONAL STRUCTURE.

▷ GRADIENT DESCENT & BACKPROP

LET'S START CONSIDERING THE SIMPLEST NETWORK POSSIBLE

$$x \; \circ \xrightarrow{\;a\;} \underset{z}{\circ} \xrightarrow{\;b\;} \circ \; y$$

$$x \to f(x,a) \to z \to g(z,b) \to y$$

BACKPROPAGATION IS ITERATIVELY APPLY CHAIN RULE WHICH WORKS SINCE THE NET IS COMPOSITIONAL:

$$y = g(z,b) = g\left(f(x,a);b\right) \leftarrow \text{COMPOSITION}$$

WE HAVE TRAINING DATA X THAT MAPS INTO OUTPUT DATA. OUR JOB IS TO FIGURE OUT a AND b. THERE IS NO UPPERBOUND IN THE ERROR SO WE CAN OBTAIN A MINIMUM WITH THE DERIVATIVE.

$$E = \frac{1}{2}(\underset{\text{MODEL}}{y} - \underset{\text{TRUTH}}{y_0})^2$$

$$\frac{\partial E}{\partial a} = 0 \to (y-y_0)\frac{dy}{dz}\frac{dz}{da} = 0$$

$$\frac{\partial E}{\partial b} = 0 \to (y-y_0)\frac{dy}{db} = 0$$

NOW WE UPDATE THE NET' PARAMETERS WITH GRADIENT DESCENT:

$$a_{k+1} = a_k + \delta \left.\frac{\partial E}{\partial a}\right|_k$$

$$b_{k+1} = b_k + \delta \left.\frac{\partial E}{\partial b}\right|_k$$

δ IS CALLED THE LEARNING RATE.

LET'S CONSIDER THE LINEAR CASE

$$z = ax$$
$$y = bz$$

$$\frac{\partial E}{\partial a} = -(y_0 - y)bx$$

$$\frac{\partial E}{\partial b} = -(y_0 - y)z = -(y_0 - y)ax$$

$$y = f(x, \beta) = f(x, A_1, A_2, \ldots, A_m)$$

$$\underset{A_j}{\arg\min}\, E(A_1, A_2, \ldots, A_m) = \underset{A_j}{\arg\min} \sum_{k=1}^{m} (\underbrace{f(x_k, \beta) - y_k}_{E_k})^2$$

LARGE SCALE OPTIMIZATION

$$\underline{x}_{j+1} = \underline{x}_j - \delta \nabla f(\bar{x}_j)$$

THE BEST WAY TO UPDATE WEIGHTS IS WITH STOCHASTIC GRADIENT DESCENT. INSTEAD OF USING ALL THE SAMPLES IN A TIME WE USE JUST A RANDOM BATCH