Gradient Descent & Neural Networks
A Categorical Approach

Following Fong, Spivak, Tuyéras, Capucci, Gavranović, Hedges, Rischel …

I: Gradient Descent

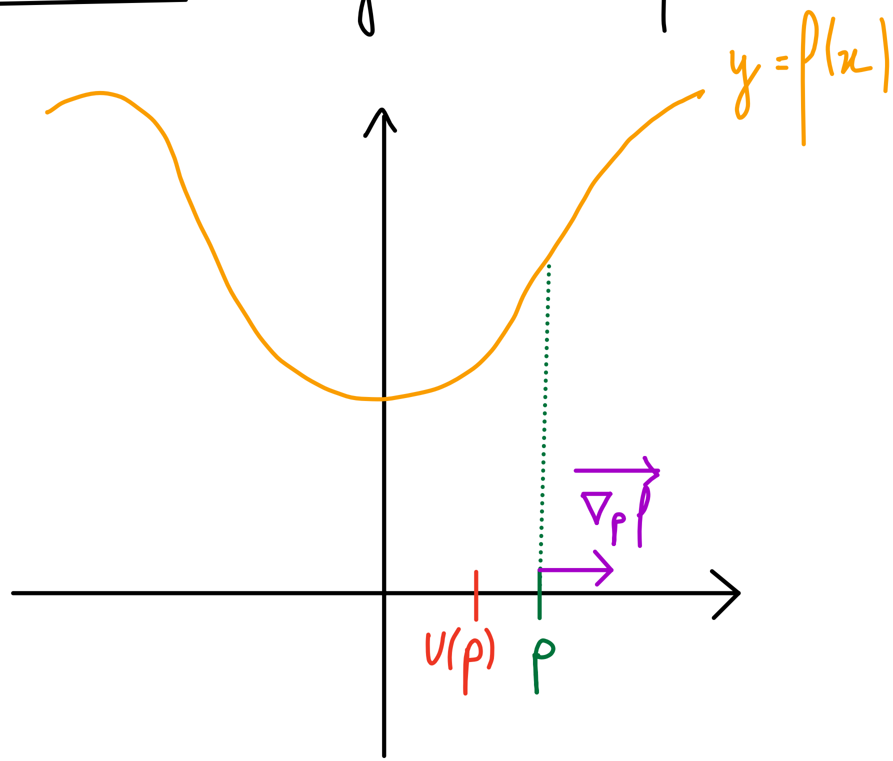
II: Categorical model: Para, Optic, …

III: Naïve Neural Networks

IV: Categorical Neural Networks

V: Training Your Network

I Gradient descent: algorithm to find local minima



$$U(p) = p - \varepsilon \overrightarrow{\nabla_p f}$$

f : total error/cost function

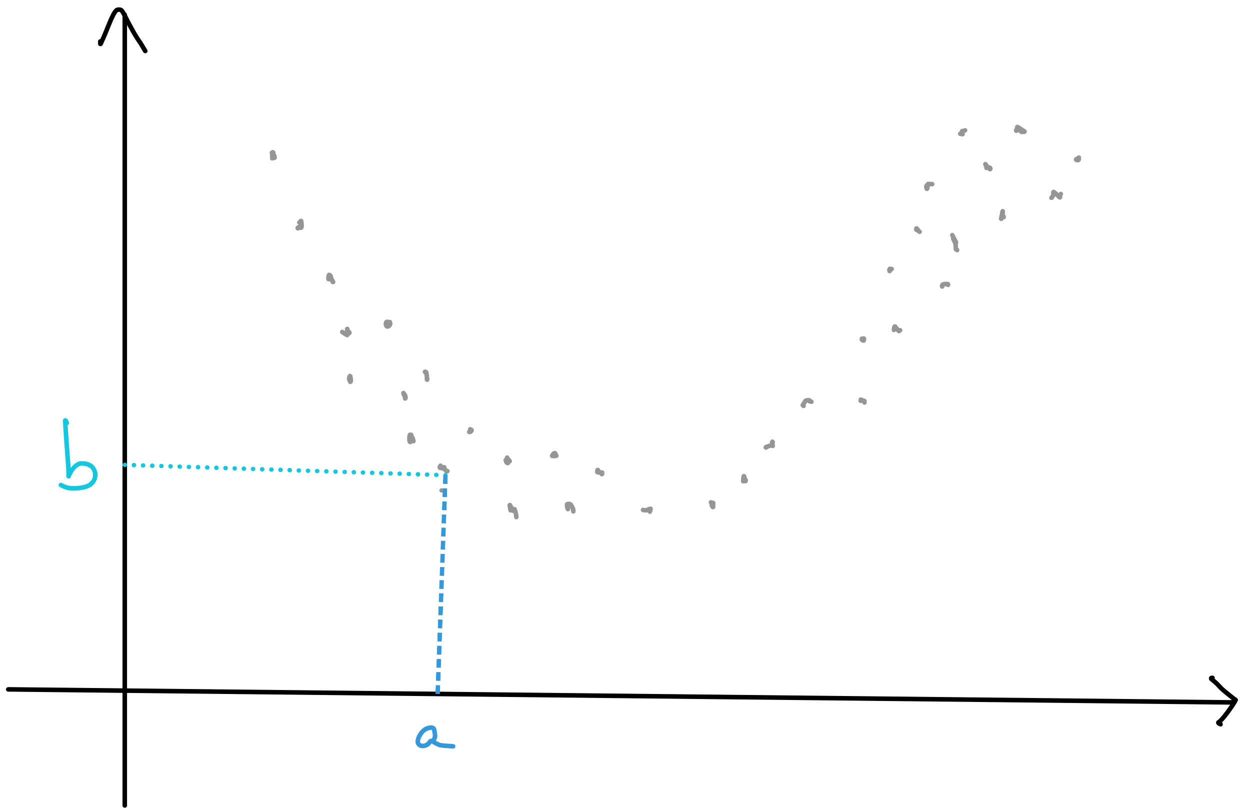
p : "First guess" parameter

$\overrightarrow{\nabla_p f}$: gradient at p.

ε : Step / learning rate

U : update function.

GD for Function Approximation



(a, b) : sample data

1) A shape of function: 2^{nd} degree polynomial

$$I(p_1, p_2, p_3, x) = p_1 + p_2 x + p_3 x^2$$

2) An error (distance) function:

$$e(x, y) = \frac{1}{2}(x - y)^2$$

Total error: $E(p, a, b) = \frac{1}{2}\left(I(p, a) - b\right)^2$

3) A step: $\varepsilon > 0$.

Get:

1) An update function:

$$U(p, a, b) = p - \varepsilon \nabla_p E(p, a, b)$$

2) A request function:

$$r(p, a, b) = \left(\frac{\partial e}{\partial x}\right)^{-1} \left(\nabla_a E(p, a, b)\right)$$

Instead of a map $\mathbb{R} \longrightarrow \mathbb{R}$, we have:

$$\mathrm{I} : \mathbb{R}^3 \times \mathbb{R} \longrightarrow \mathbb{R} \quad ,$$

$$U : \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}^3 \quad ,$$

$$\rho : \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R} \quad .$$

II: Categorical Model

<u>Definition</u>: Let $\mathcal{C}$ be a category with products and terminal $\top$.

The category **Para($\mathcal{C}$)** has:

- Objects: same as $\mathcal{C}$
- Morphisms $A \xrightarrow{P, f} B$:
  pairs $(P \in ob(\mathcal{C}),\ P \times A \xrightarrow{f} B)$

identities: $A \xrightarrow{\top, 1_A} B$

Composition:

$$\begin{tikzcd} A \arrow[r, "{P, f}"] \arrow[rr, bend right, "{Q \times P,\ g \circ Q \times f}"'] & B \arrow[r, "{Q, g}"] & C \end{tikzcd}$$

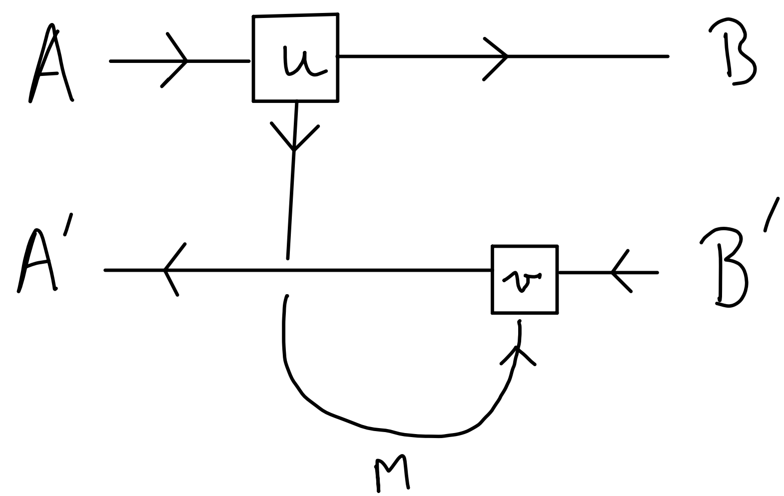
$$Q \times P \times A \xrightarrow{Q \times f} Q \times B \xrightarrow{g} C$$

Definition: Let $\mathcal{C}$ be a category with ...

The category $\mathrm{Optic}(\mathcal{C})$ has:

- Objects: pairs $\begin{pmatrix} A \\ A' \end{pmatrix} \in \mathrm{ob}(\mathcal{C})^2$
- Morphisms $\begin{pmatrix} A \\ A' \end{pmatrix} \overset{f}{\underset{g}{\rightleftarrows}} \begin{pmatrix} B \\ B' \end{pmatrix}$ are classes of triples

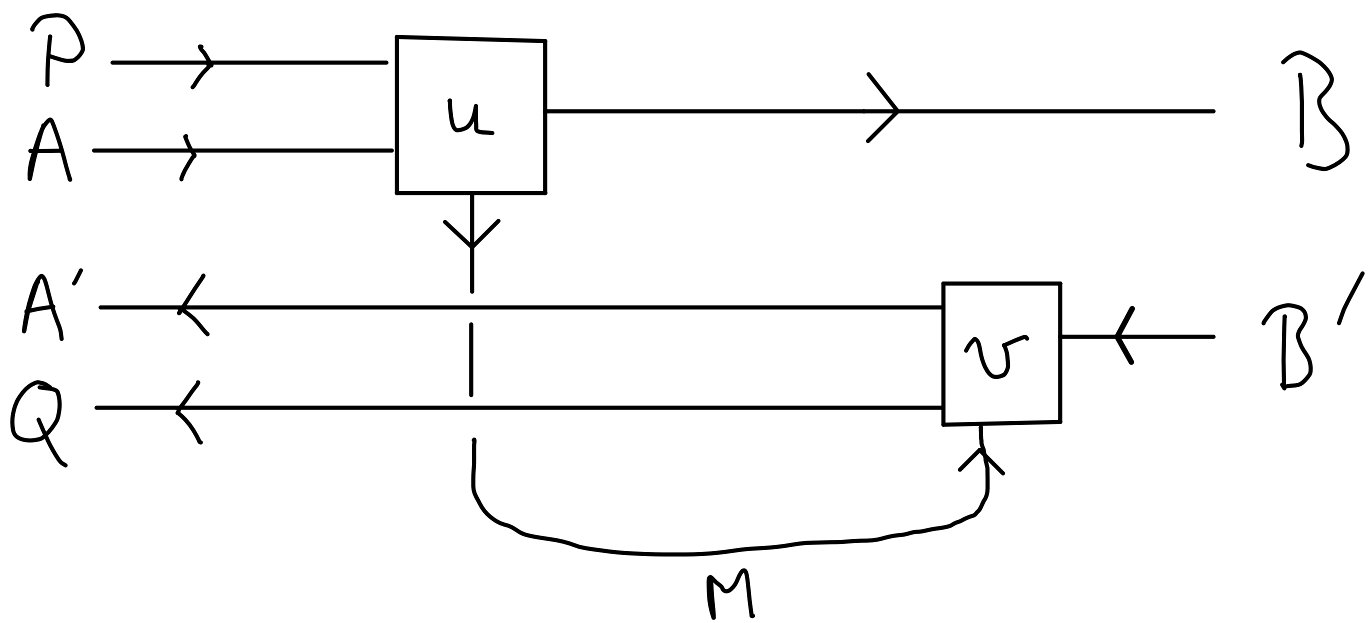
$$\big(M \in \mathrm{ob}(\mathcal{C}),\ f : A \longrightarrow B \times M,\ g : B' \times M \longrightarrow A' \big) \; \sim$$



$$\boxed{\mathrm{Optic}(\mathcal{C})\left(\begin{pmatrix} A \\ A' \end{pmatrix}, \begin{pmatrix} B \\ B' \end{pmatrix} \right) := \int^{M \in \mathrm{ob}(\mathcal{C})} \mathcal{C}(A, M \times B) \times \mathcal{C}(M \times B', A')}$$

Parametrised Optics: $\mathbf{Para}(\mathbf{Optic}(\mathcal{C}))$ has

- objects: $\mathrm{ob}(\mathcal{C})^{\times 2}$,
- morphism: $\begin{pmatrix} A \\ A' \end{pmatrix} \overset{P}{\underset{Q}{\rightleftarrows}} \begin{pmatrix} B \\ B' \end{pmatrix}$:



Def: **Smooth** has: $\mathrm{ob}(\mathrm{Smooth}) = \mathbb{N}$,

$\mathrm{Smooth}(n, m) = C^\infty(\mathbb{R}^n, \mathbb{R}^m)$

$n \times m := n + m$

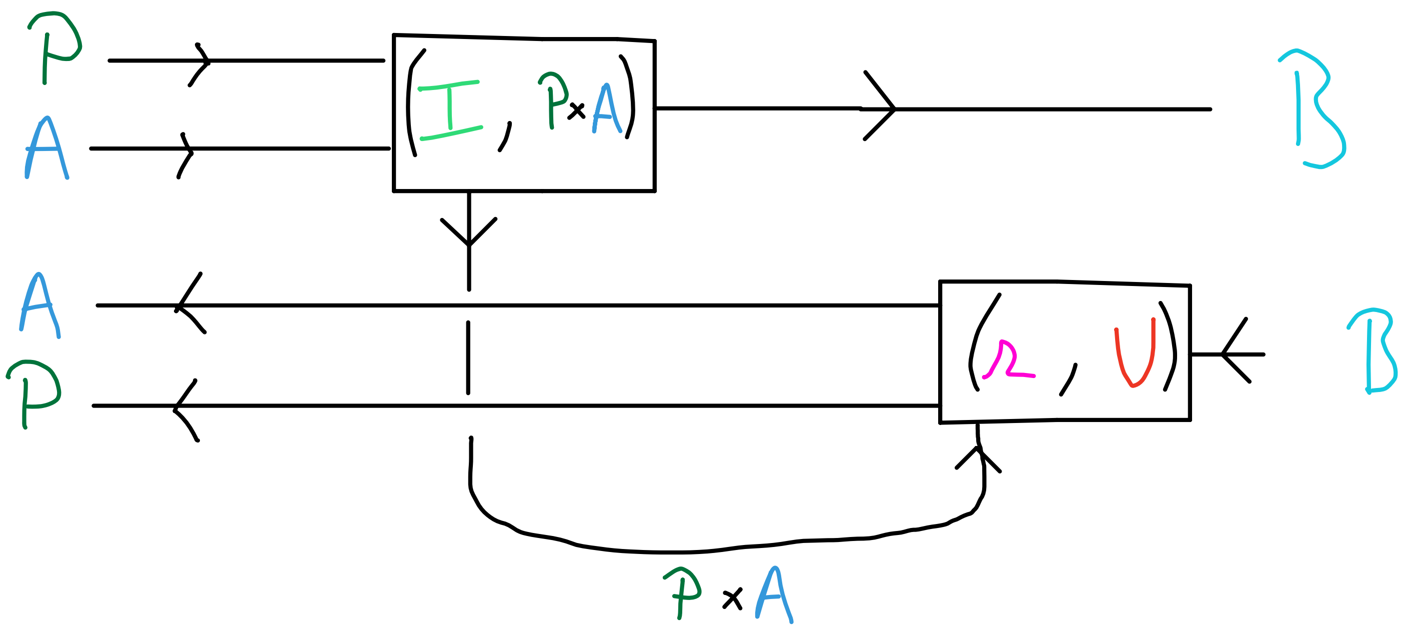
Theorem (Fong - Spivak - Tuyéras): Given a step ε and an error function e, there is a Gradient Descent functor:

$$GD_{\varepsilon, e} : \mathrm{Para}(\mathrm{Smooth}) \longrightarrow \mathrm{Para}(\mathrm{Optic}(\mathrm{Smooth}))$$

such that:

$$GD(A) = \begin{pmatrix} A \\ A \end{pmatrix}$$

$$GD\left(A \xrightarrow{P,\, I} B\right) = \begin{pmatrix} A \\ A \end{pmatrix} \underset{P}{\overset{P}{\rightleftarrows}} \begin{pmatrix} B \\ B \end{pmatrix}$$ is represented by:



$U(p,a,b) = p - \varepsilon \nabla_p E(p,a,b)$ and $r(p,a,b) = \left(\frac{\partial e}{\partial x}\right)^{-1} \left(\nabla_a E(p,a,b)\right)$

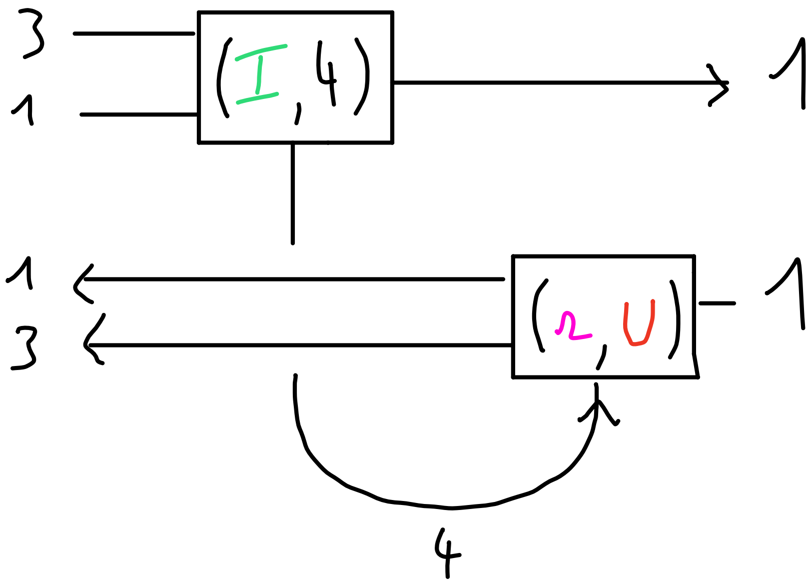
In our Example:

$$I(p_1, p_2, p_3, x) = p_1 + p_2 x + p_3 x^2$$

$$1 \xrightarrow{3,\, I} 1 \in \mathrm{Para}(\mathrm{Smooth})$$

$$GD_{\varepsilon, e}\left(1 \xrightarrow{3,\, I} 1\right) =$$

$$e(x, y) = \frac{1}{2}(x - y)^2$$



Reparametrisation

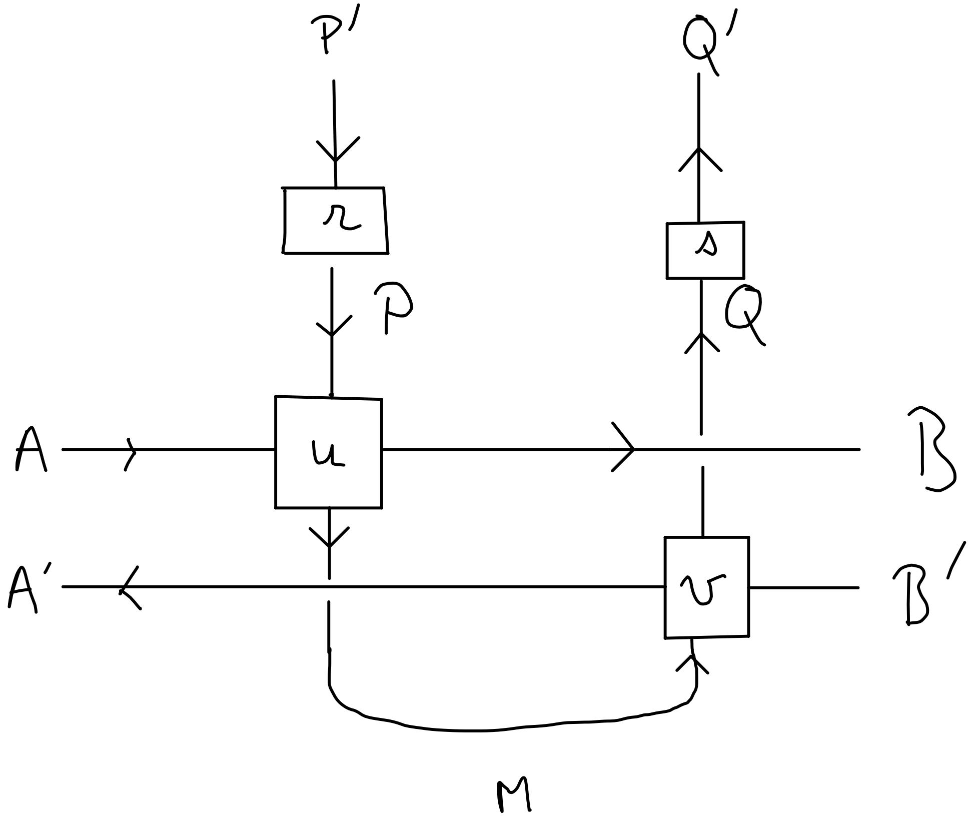
$\mathrm{Para}(\mathcal{C})$ is (in fact) a 2-category with:

2-cells:

$$
\begin{tikzcd}
A \arrow[r, "{P, f}"] \arrow[d, phantom, "\Downarrow r"] & B \\
A \arrow[r, "{P', f'}"'] & B
\end{tikzcd}
\quad = \quad
\begin{tikzcd}
P \times A \arrow[r, "f"] \arrow[dr, phantom, "\circlearrowleft"] & B \\
P' \times A \arrow[u, "r \times A"] \arrow[ur, "f'"'] &
\end{tikzcd}
$$

In $\mathrm{Para}(\mathrm{Optic})$,

$$
\begin{tikzcd}
\begin{pmatrix} A \\ B \end{pmatrix} \arrow[r, shift left, "P"] \arrow[d, phantom, "\Downarrow (r,s)"] & \begin{pmatrix} A' \\ B' \end{pmatrix} \arrow[l, shift left, "Q"] \\
\begin{pmatrix} A \\ B \end{pmatrix} \arrow[r, shift left, "P'"] & \begin{pmatrix} A' \\ B' \end{pmatrix} \arrow[l, shift left, "Q'"]
\end{tikzcd}
\quad =
$$



Generalisation of GD using Differential Categories

Def: A **Cartesian Differential Category** is a cartesian (left additive) Category $\mathscr{C}$ equipped with a **differential combinator**:

$$\frac{A \xrightarrow{f} B}{A \times A \xrightarrow{Df} B} D \qquad \text{satisfying} \ldots$$

Example: SMOOTH, with

$$\frac{n \xrightarrow{f} m}{\begin{array}{c} n+n \xrightarrow{Df} m \\ (x, y) \longmapsto D_x f(y) \end{array}} D$$

Remember: $r(p, a, b) = \left(\frac{\partial e}{\partial x}\right)^{-1} (\nabla_a E(p, a, b))$

Generalisation of GD using Differential Categories

Def: A **Reverse Differential Category** is a cartesian (left additive) category $\mathscr{C}$ equipped with a **differential combinator**:

$$\frac{A \xrightarrow{f} B}{A \times B \xrightarrow{Rf} B}\ R$$

satisfying ...

Example: SMOOTH, with

$$\frac{n \xrightarrow{f} m}{n + m \xrightarrow{Rf} n}\ R$$

$$(x, y) \longmapsto (D_x f)^t (y)$$

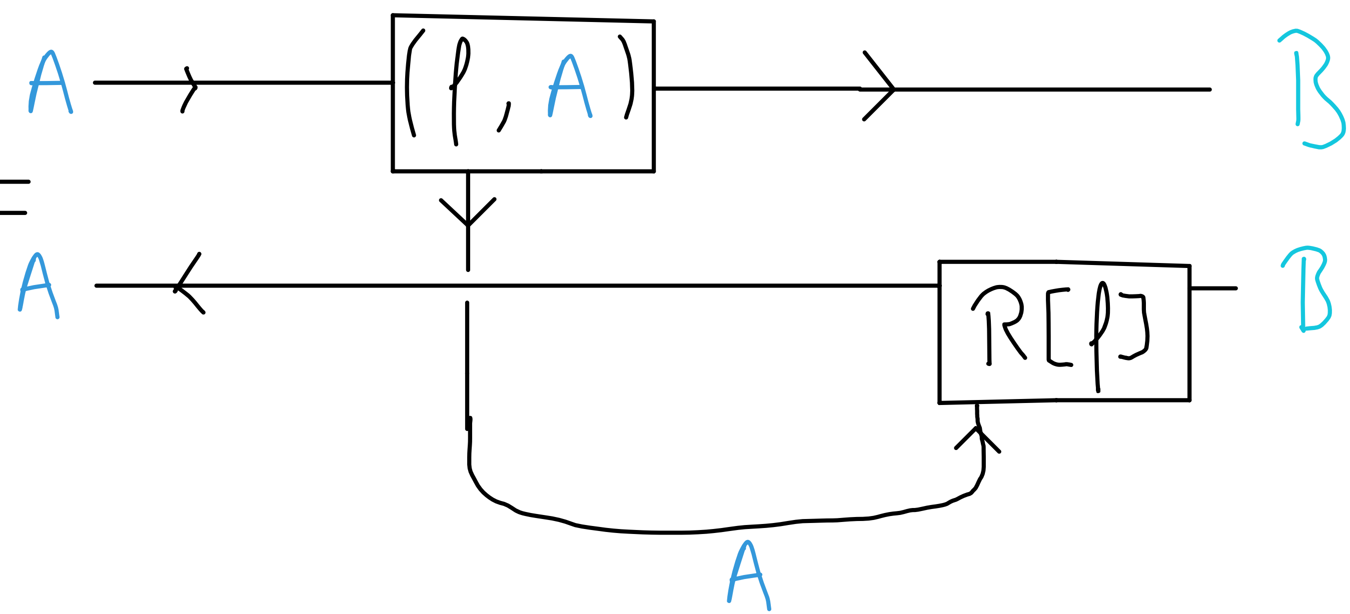
Theorem (Cockett-Cruttwell-Gallagher-Lemay-MacAdam-Plotkin-Pronk)

For any R.D.C $\mathcal{C}$, there is a functor:

$$L : \mathcal{C} \longrightarrow \mathrm{Optic}(\mathcal{C}) \qquad \text{such that:}$$

$$L(A) = \begin{pmatrix} A \\ A \end{pmatrix}, \text{ and}$$

$$L(A \xrightarrow{f} B) =$$



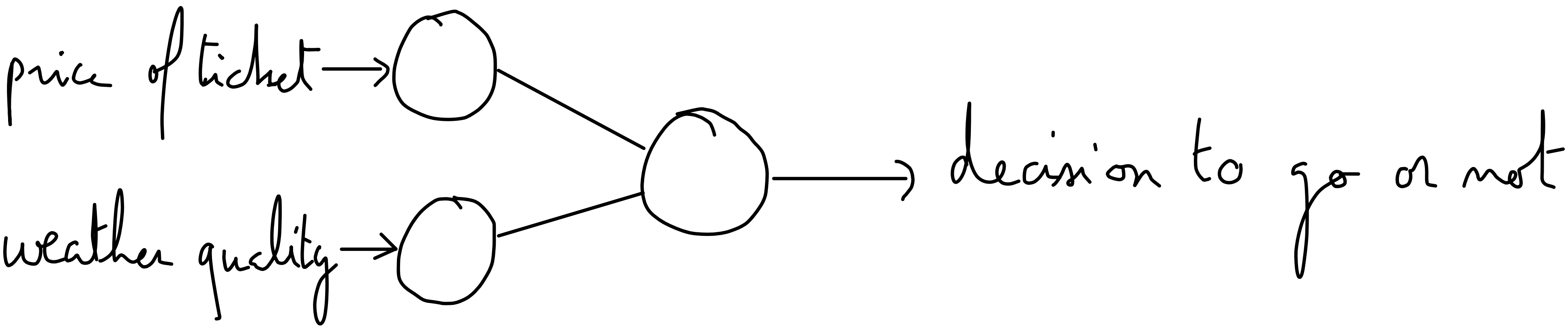
$GD = \mathrm{Para}(L)$

III (Naive) Neural Networks

<u>Ex</u>: I want to go to London if

a) The plane ticket is cheap enough

b) The weather is good enough.

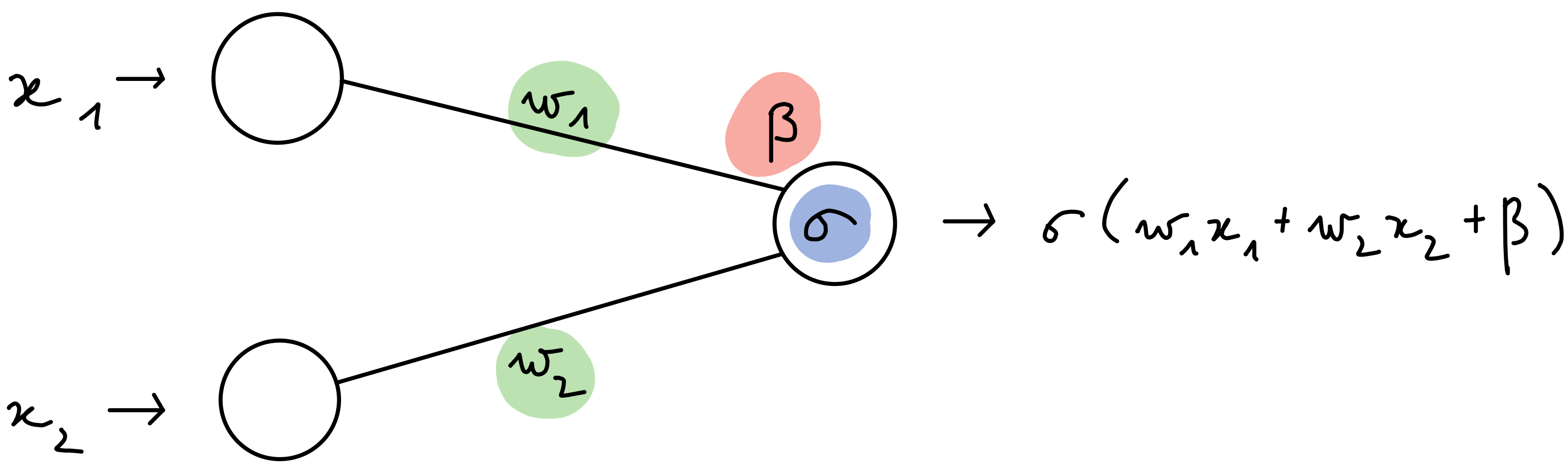


1) Set weights: w_1 = importance of cheapness

w_2 = importance of weather.

2) Set a bias: I'll go if $w_1 x_1 + w_2 x_2 > -\beta$

3) Choose an activation function σ (e.g $\sigma(z) = \frac{1}{1+e^{-z}}$)

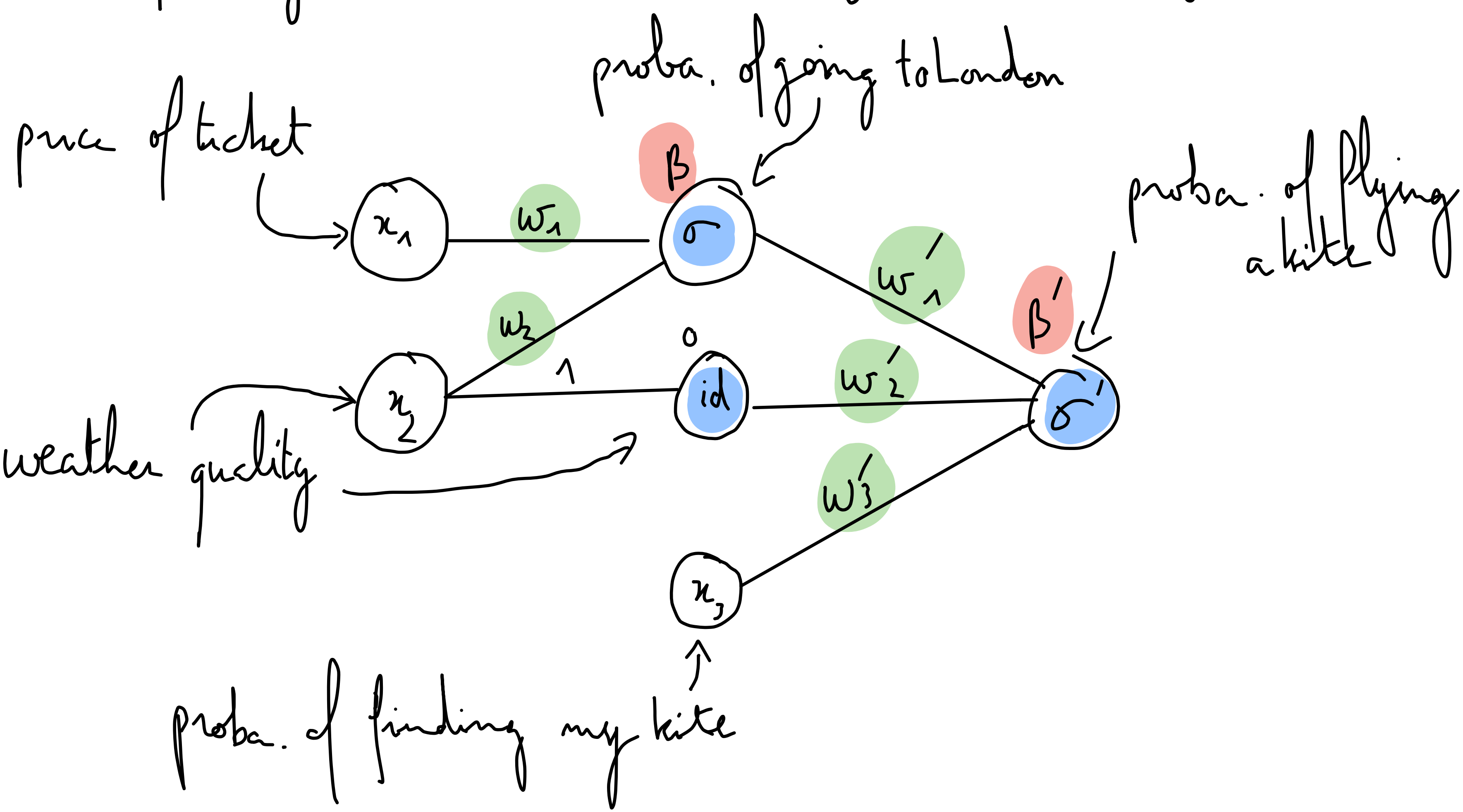


1-Layered Neural Network

Composing Neural Networks

I want to fly my kite a) If I can find it,
b) If the weather is good,

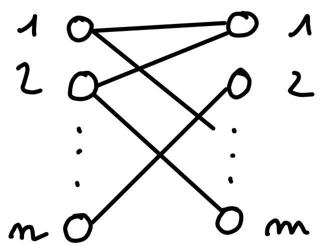
but if I go to London, I may be too busy.



IV Categorical Neural Networks

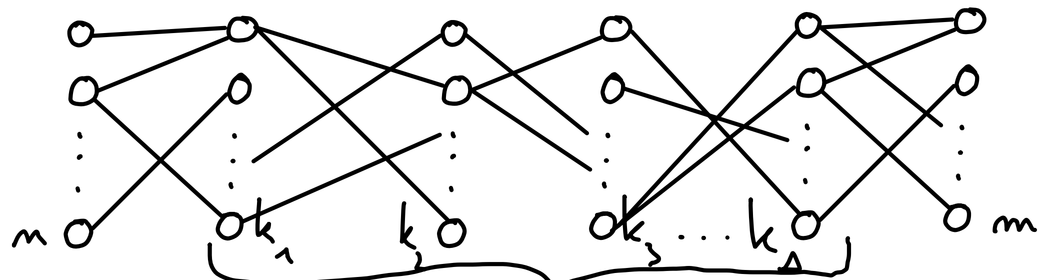
Def: The category (PROP) NNET has:

- objects : $\mathbb{N}$
- morphism $m \longrightarrow n$: freely generated by $\mathcal{P}([n] \times [m])$



1-layered morphism

$n \longrightarrow m$



s-layered morphism

$n \longrightarrow m$

"hidden layers"

Prop (Fong - Spivak - Tuyéras): Given $\sigma : \mathbb{R} \to \mathbb{R}$ there is a functor

$$I_\sigma : \mathrm{NNET} \longrightarrow \mathrm{Para}(\mathrm{Smooth})$$ such that

$I_\sigma(n) = n$, and for $C \subseteq [n] \times [m]$,

$$I_\sigma(C) : n \xrightarrow{|C|+m,\, I} m$$ with:

$$I : \mathbb{R}^{|C|+m} \times \mathbb{R}^n \longrightarrow \mathbb{R}^m$$

$$\left((w_{ji})_{(j,i) \in C}, (\beta_j)_{j \in [m]}, (x_i)_{i \in [n]} \right) \longmapsto \left(\sigma \left(\left(\sum_{(j,i) \in C} w_{ji} x_i \right) + \beta_j \right) \right)_{j \in [m]}$$

<u>Terminology</u> (Fong - Spivak - Tuyéras): Given σ, $\mathcal{E}$, e

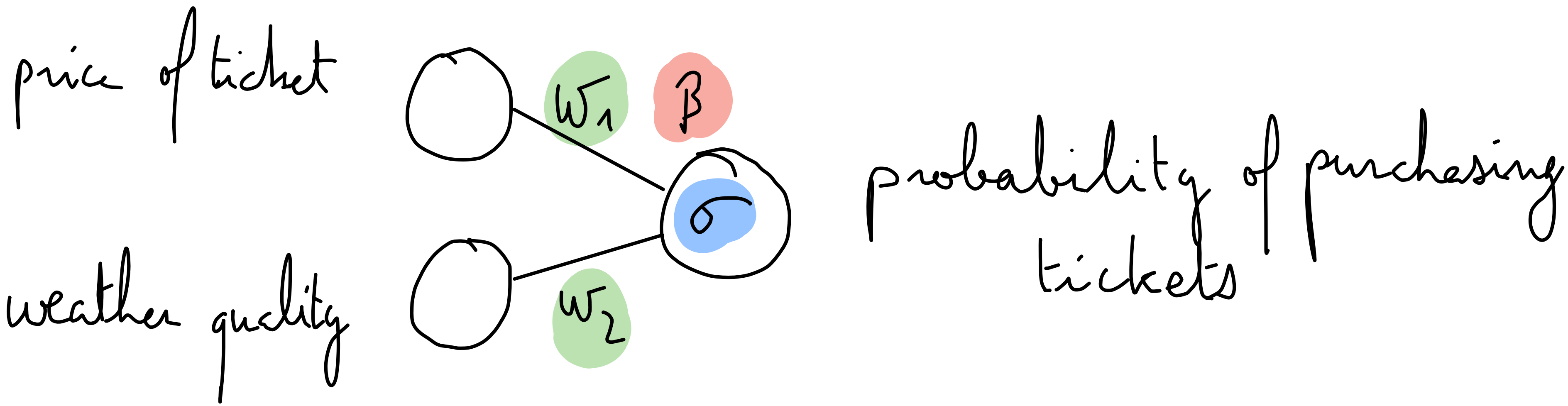
$$G_{\mathcal{E}, e} \circ I_{\sigma} : \mathrm{NNET} \longrightarrow \mathrm{Para}(\mathrm{Optic}(\mathrm{Smooth}))$$

is <u>deep learning</u>. The resulting request function

is <u>deep dreaming</u>.

V Training Your Network

"Go to London" example:



a) Pick a starting guess for w_1, w_2, β

b) Train your network using a data set:

$\{(\text{conditions}, \text{outcome})\}$, e.g.:

((ticket was 80€, weather was meh), did not purchase)

For each sample data, compute:

$$U\left(\omega_1, \omega_2, \beta, \text{conditions}, \text{outcome}\right) = \text{new } \omega_1, \omega_2, \beta$$

...

At the end,

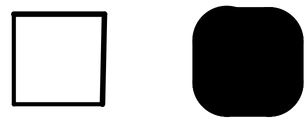
$$I\left(\omega_1, \omega_2, \beta, \text{conditions}\right) \quad \text{predicts the decision.}$$

Example : Image recognition.

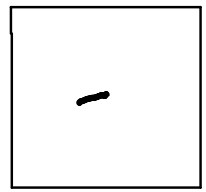
Aim : Recognise hand-drawn digits in b & w,

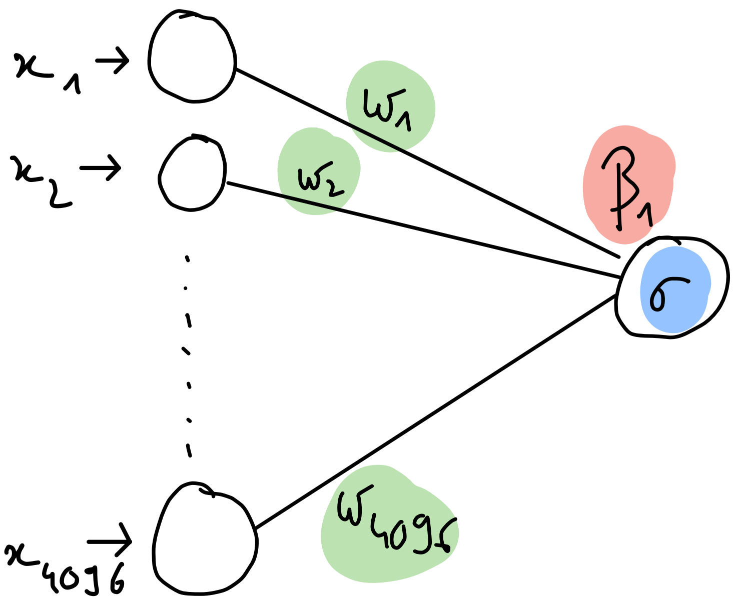
(4096 =) 64 × 64 pixels images.

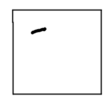
1) Encoding images: $(x_i) \in \mathbb{R}^{4096}$ $]-\infty ; +\infty[$



2) First Layers recognise small edges:



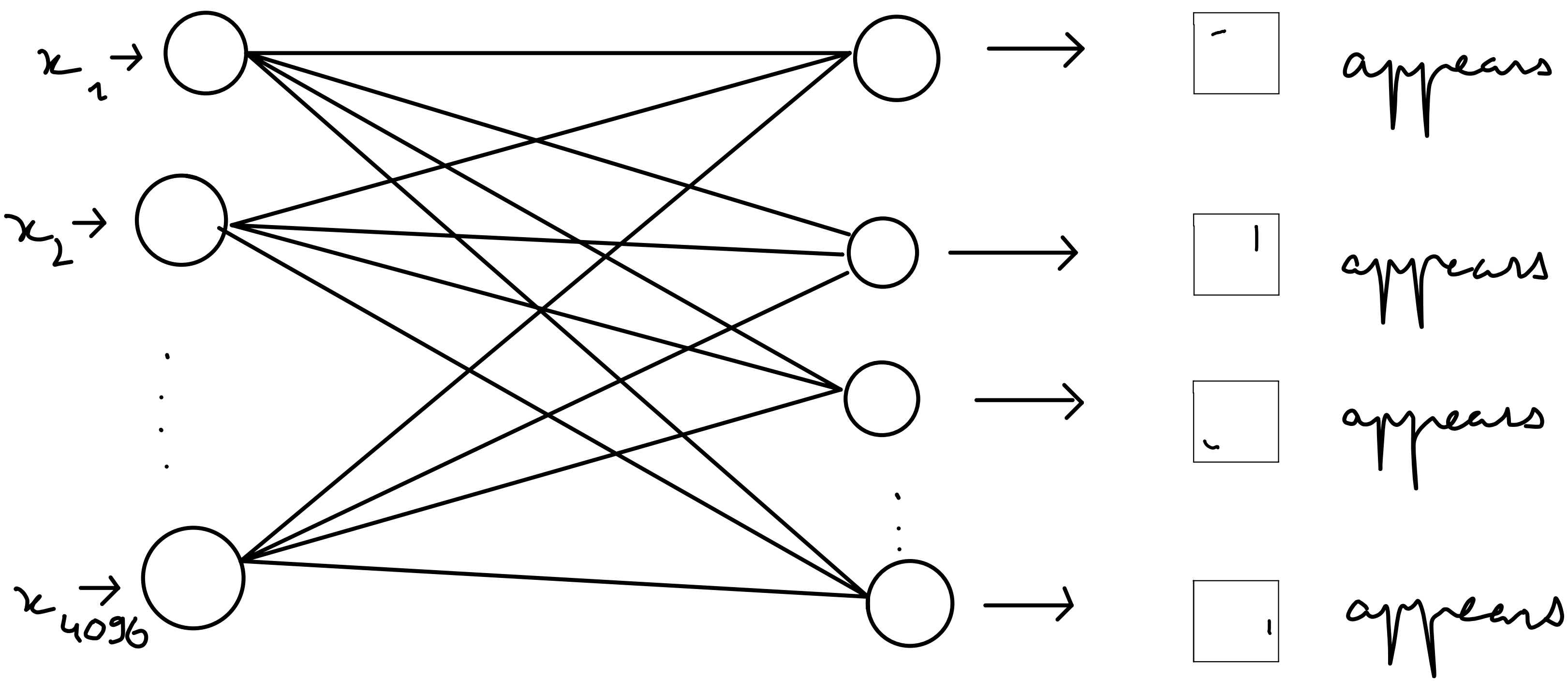


$\rightarrow$  appears

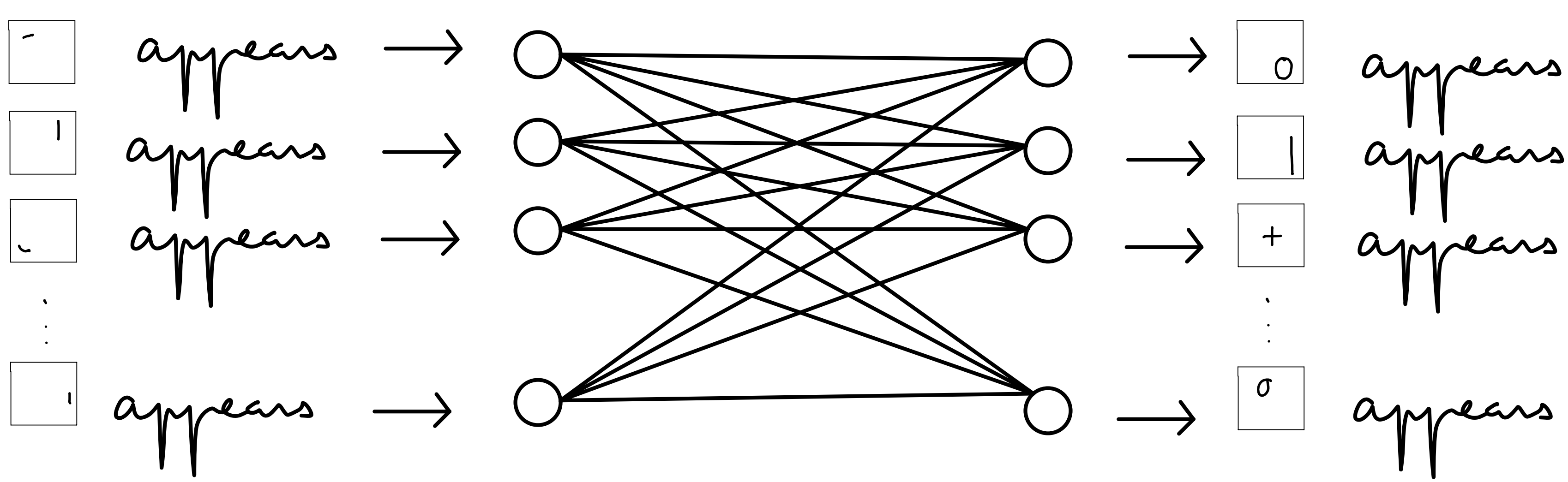
3) Train your network with sample data:

$$\left\{ \left(\boxed{5}\ , \text{mostly yes} \right) \right\}$$

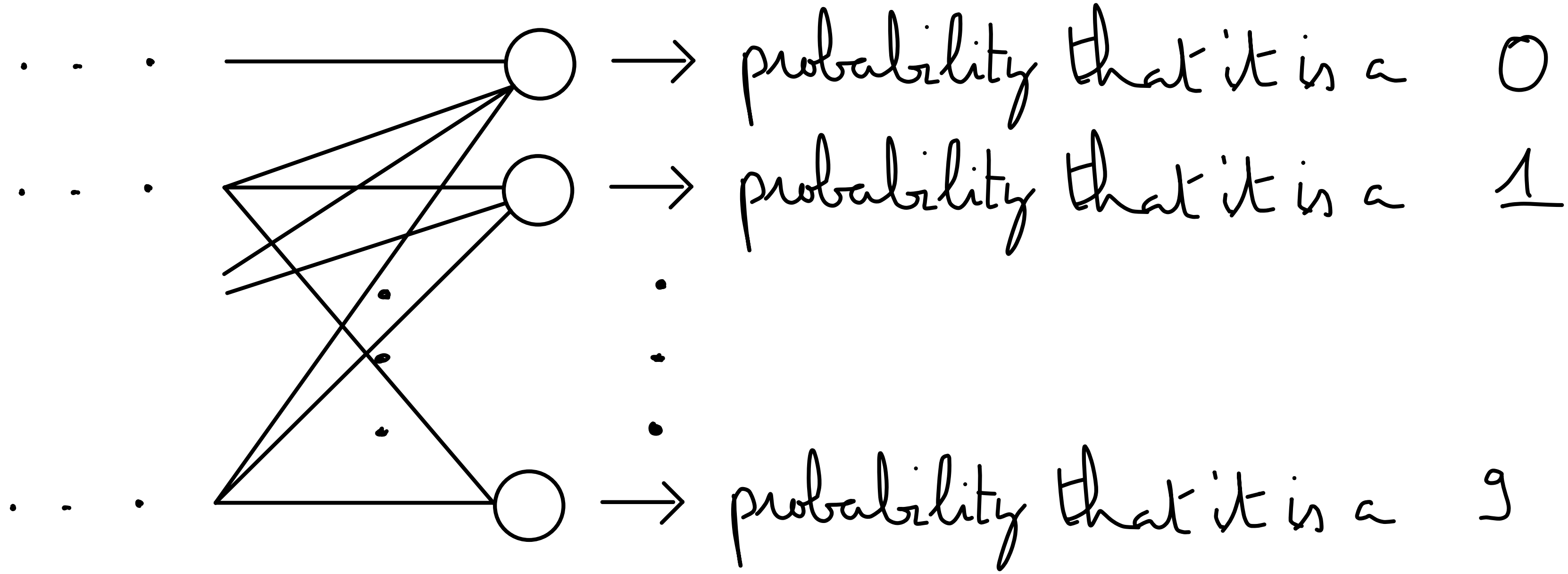
4) Repeat for <u>many</u> edges to get first layer.



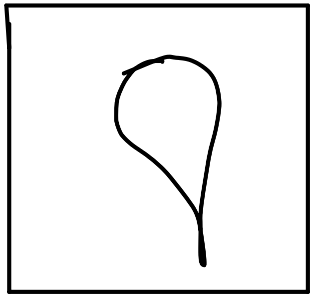
5) Second layers, third layers...: more complex shapes



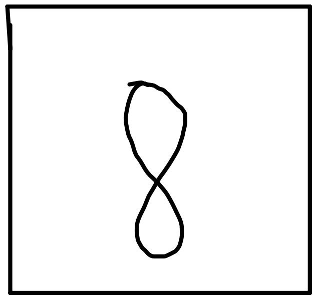
6) Final Layer



What does r do?

$r\,($ w, β,  , "This is definitely an 8" $)$

$=$



. . . Image Generation

Thank You!

References:

Fong, Spivak, Tuyéras - Backprop as Functor: A compositional perspective on supervised learning

Cruttwell, Gavranović, Ghani, Wilson, Zanasi - Categorical Foundations of Gradient-Based Learning

Cockett, Cruttwell, Gallagher, Lemay, MacAdam, Plotkin, Pronk - Reverse derivative categories

Capucci, Gavranović, Hedges, Rischel - Towards Foundations of Categorical Cybernetics