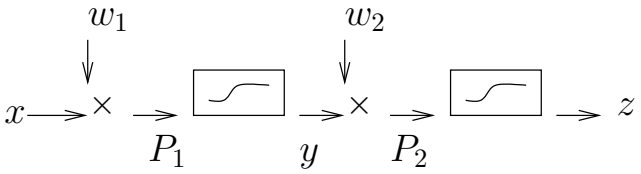
Training two neurons

- 1 neuron isn't a net
- 2 neurons is a net

This is a simplest possible neural network.



(Note that P_1 and P_2 are products.)

We talk about performance, loss, cost function:

$$L = \frac{1}{2}(d - z)^2$$

So we want to do gradient descent, this means computing partial derivatives.

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial w_2} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial P_2}\frac{\partial P_2}{\partial w_2}$$

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial P_2}\frac{\partial P_2}{\partial y}\frac{\partial y}{\partial P_1}\frac{\partial P_1}{\partial w_1}$$

How do we compute these things?

$$\frac{\partial P_2}{\partial w_2} = y$$

$$\frac{\partial z}{\partial P_2} = z(1 - z) \quad \text{(come back to this)}$$

$$\frac{\partial L}{\partial z} = d - z$$

How do we compute the derivative of the sigmoid function?

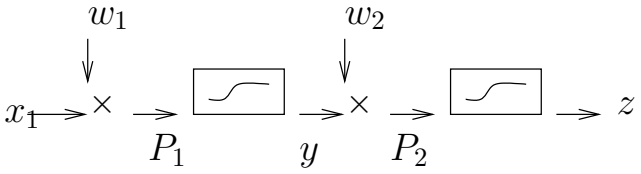
$$
\begin{aligned}
\frac{\partial \beta}{\partial \alpha} &= \frac{\partial}{\partial \alpha}\left(\frac{1}{1+e^{-\alpha}}\right) \\
&= \frac{\partial}{\partial \alpha}\left(1+e^{-\alpha}\right)^{-1} \\
&= -\left(1+e^{-\alpha}\right)^{-2} e^{-\alpha}(-1) \\
&= \frac{e^{-\alpha}}{1+e^{-\alpha}} \cdot \frac{1}{1+e^{-\alpha}} \\
&= \frac{\mathbf{1}+e^{-\alpha}-\mathbf{1}}{1+e^{-\alpha}} \cdot \frac{1}{1+e^{-\alpha}} \\
&= \left(\frac{1+e^{-\alpha}}{1+e^{-\alpha}} - \frac{1}{1+e^{-\alpha}}\right) \cdot \frac{1}{1+e^{-\alpha}} \\
&= (1-\beta)\beta
\end{aligned}
$$

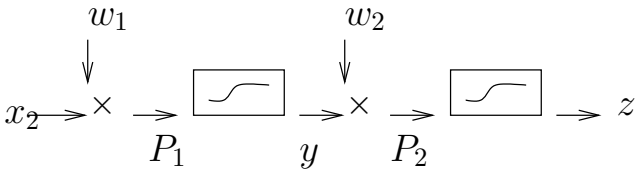
This is pretty cool: the derivative of the sigmoid function is independent of the input!

Note : at each phase, we only need to compute the last three factors.

Towards real life...

In real life, we have more than one of these going on at once. Sort of like this:





And then there are cross-overs with more weights, more multipliers. So we are again at risk of exponential blow-up: there are an exponential number of paths through the network.

But note that the dependencies are only by column, so lots of things are already computed.

In other words,

- Linear in depth
- Quadratic in width

> **Exercise:** Compute all of the partial derivatives in the 2×2 network above, assuming the interconnections we've shown.

MLP

This is our first step from 2 neurons to millions.

This is a real neural network. It's not a perceptron. Geoffrey Hinton apologised.

Back to computing hand-written digits.

8×8, then 16×16. Maybe 32×32, but let's do 28×28 for fun. We'll assume gray scale with activation in $[0, 1]$.

We'll use sigmoids to make intermediate activates also in $(0, 1)$.

Show hidden layers. Make 16 in each (somewhat arbitrary).

Then $784 \times 16 + 16 \times 16 + 16 \times 10 = 12960$ weights. Plus biases, so let's say 13K.

We might hope intermediate layers capture abstraction: say edges then loops. This time around, they don't.

Walk through backpropagation here. Note that ∇C tells us which changes matter the most.

Talk about inverse images of neuron outputs.

Talk about strange ways ANNs get things wrong (i.e., they don't recognise what we do, they can be confident about garbage).

Uber pedestrian accident

In March 2018 in Phoenix, an Uber self-driving car struck and killed a woman crossing the street with her bicycle.

- Radar detected woman 5.6 seconds before impact. SUV in far right lane, pedestrian crossing from the left. System classified as vehicle but didn't recognise that she was moving.
- Lidar pinged her several times over next few seconds, but classification repeatedly changed, so no continuity of recognition. It was a different object each time.

- At 1.5 seconds before impact, woman was partially in SUV lane, so system plotted course to steer around woman.
- Milliseconds later, identification as a bicycle on collision course. Abandons plan, which hadn't taken into account that bicycles move faster than pedestrians.
- Uber had disabled the emergency steering and braking systems because erratic. So SUV began gradual deceleration. At 1.2 seconds before impact, SUV was moving at 65 kph (40 mph).
- One second later (0.2 seconds before impact), system alerted human safety driver that it had initated a controlled slowdown. Safety driver intervened, but 0.2 seconds is inadequate for a human. The SUV struck the woman, and then the safety driver engaged the brakes.
- She was not in a crosswalk, and the system had learned to identify pedestrians in crosswalks (a correlated feature).

Deep networks

We want to go from 2 parameters to millions.

- convolution
- pooling — max pooling
- kernel *(taking some neurons and running them across the image)*
- multiple kernels
- softmax
- dropout

We don't really know why any of this works.

Talk about auto-encoders.