# Explanatory Notes for 6.390

Shaunticlair Ruiz (Current TA)

Spring 2023

# Choices of activation function

Our linear model is entirely **defined** by its input: the number of **weights** in a neuron is just the number of **inputs** $m$.

But our **activation** function is up to us to decide: what works best?

## Trying out linear activation

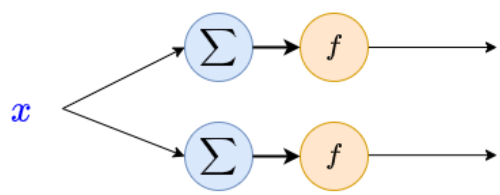
The simplest assumption would be to just use the **identity** function

$$f(z) = z \tag{1}$$

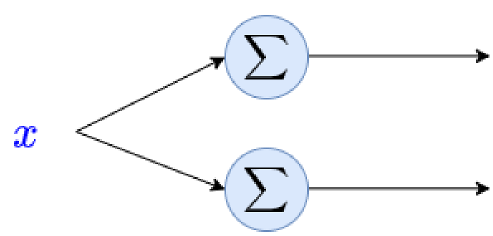
We might hope that we can combine a bunch of simple, **linear** models, and get a more sophisticated model. Why bother having a **nonlinear** activation at all?

Well, it turns out, combining **multiple** linear layers doesn't make our model any stronger. Let's try an example: we'll take a network with 2 layers, two neurons each.

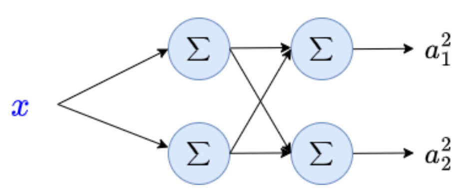
Let's look at layer 1:




Since the activation function has **no effect** on our result, we can **omit** it:

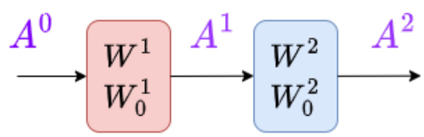



And now, we can show our **full** network:



## Linear Layers: An example

We'll assume **two** inputs $A_0 = [x_1, x_2]^T$. For our sanity, we'll lump all of the weights in each **layer**:




We'll leave out $W_0$ terms to make it more readable, but the same will apply.

Layer 1:

$$A^1 = (W^1)^T A_0 \tag{2}$$

Layer 2:

$$A^2 = \overbrace{(W^2)^T (W^1)^T}^{\text{Weight matrices}} A_0 \tag{3}$$

The full function for this equation is two matrices, **multiplied** by our input vector.

Let's take an arbitrary example:

$$W^1 = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad W^2 = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \tag{4}$$

Our equation becomes:

$$A^2 = \overbrace{\begin{bmatrix} 5 & 7 \\ 6 & 8 \end{bmatrix} \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}}^{\text{Transposed matrices}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{5}$$

We created this function by applying two matrices separately. But, can't we **combine** them?

$$A^2 = \begin{bmatrix} 19 & 43 \\ 22 & 50 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{6}$$

Wait, but this looks like a **one-layer** network with those weights! The second layer is **pointless**, we could have represented it with a single layer...

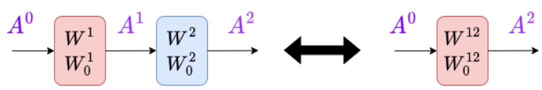
$$(W^{12})^T = \begin{bmatrix} 19 & 43 \\ 22 & 50 \end{bmatrix} \tag{7}$$

## The problem with linear networks

In fact, this is true in general: we can always take our **two** linear layers and combine them into **one**.

$$(W^2)^\top (W^1)^\top = W^{12} \tag{8}$$

Our network is **equivalent** to the supposedly "simpler" one-layer network.




What if we have more layers? Well, we can just combine them one-by-one. At the end, we're just left with one layer:

$$(W^L)^\top (W^{L-1})^\top \cdots (W^2)^\top (W^1)^\top = W \tag{9}$$

And so, we can't just use linear layers: we **need** a **nonlinear** activation function.

**Concept 1**

Having multiple consecutive **linear layers** (i.e. layers with linear **activation** functions) is **equivalent** to having one linear layer in its place.

This means that we do not expand our **hypothesis** class by using more linear layers: we have to use **nonlinear** activation functions.

This problem is even worse than it seems: let's see why. Since we can **combine** $n$ linear layers together into one, what happens if we only have **one** linear layer?

Suppose layer $\ell$ is linear. The next layer contains a **linear** component and a non-linear **activation** component. We'll focus on just the linear part.

Activation comes after this step, so we would just use $f(z^{\ell+1})$.

$$z^{\ell+1} = (W^{\ell+1})^\top x^{\ell+1} = (W^{\ell+1})^\top (W^\ell)^\top x^\ell \tag{10}$$

Wait: we have **two** consecutive **linear** components. We can combine layer $\ell$ with the linear component of the next layer!

$$(W^{\ell+1})^\top (W^\ell)^\top x^\ell = W x^\ell \tag{11}$$

Now, we've removed layer $\ell$ entirely: it makes no difference to have even just one **hidden** linear layer!

**Concept 2**

Even having one hidden **linear layer** is **redundant**: it's **equivalent** to not having that layer at all.

Since this requires **more computation** for no benefit, we **almost never** make linear hidden layers.

So, linear models are out. What if we use something **nonlinear**?

$$A^2 = f\left((W^2)^T A^1\right) \tag{12}$$

We get something that doesn't seem to **simplify**:

This is ugly, but we don't have to worry about the details.

$$A^2 = f\left((W^2)^T \overbrace{\boxed{f\left((W^1)^T x\right)}}^{A^1}\right) \tag{13}$$

If we choose our function right (and avoid linearity), this cannot be simplified to a single layer! That means, this function is **different** (and likely more **complex**) than a one-layer model.

And this kind of **expressiveness** is exactly what we're looking for.