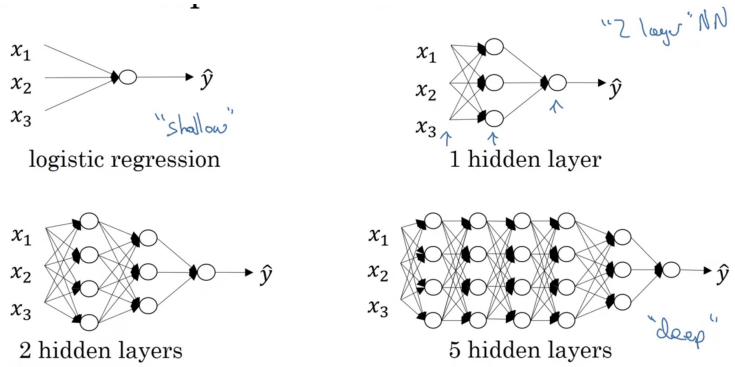
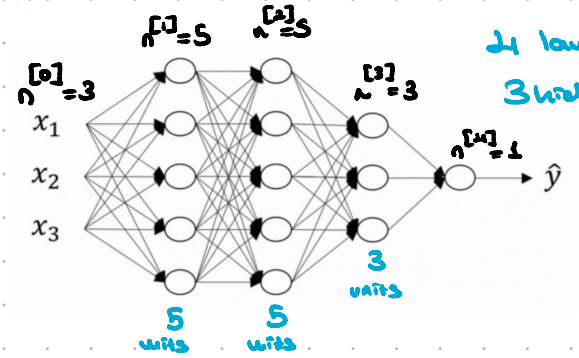
Deep L-Layer NN



there are functions that deep NN can learn but shallow models cannot.

Notations



L = # layers (number of layers)

$n^{[l]}$: # units in layer l

$a^{[l]}$: activations in layer l

$= g^{[l]}(z^{[l]})$

$w^{[l]}$: weights for $z^{[l]}$

$b^{[l]}$: bias for $z^{[l]}$

Forward Propagation in DNN

within a for loop:

$$Z^{[l]} = W^{[l]} A^{[l-1]} + b^{[l]}$$

$$A^{[l]} = g^{[l]}(Z^{[l]})$$

} This is a location where we cannot avoid having an explicit for-loop

Why Deep Representations?

Earlier layers learn simple information, then later layers use those simpler information, and compose more complex information

great for compositional information

audio → detect low level audio info → detect basic units of sound → detect words → detect sentences

earlier levels ←——→ deeper layers

images → detect edges → detect eye, mouth, nose → detect faces

Circuit Theory

informally:

there ∃ functions that you can compute with "small" (low # hidden units) L-layer deep NN

THAT

shallower NNs require exponentially more hidden units to compute

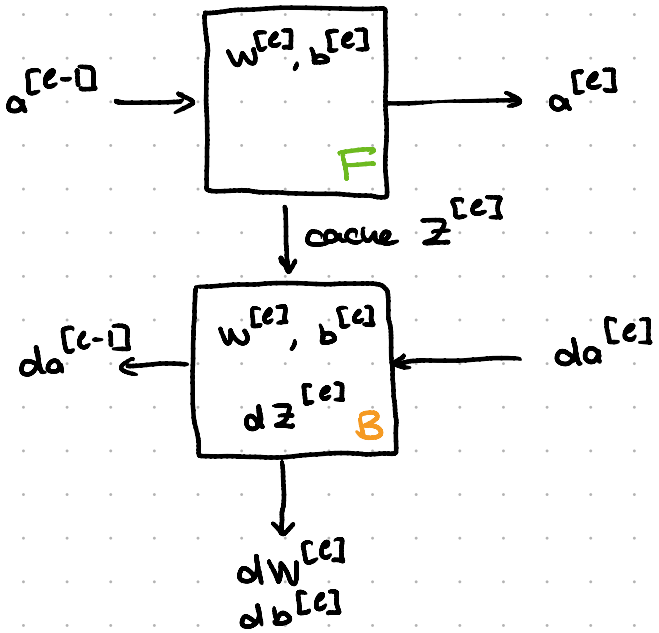
i.e. there are mathematical functions that are much easier to compute with deep networks compared to shallow networks

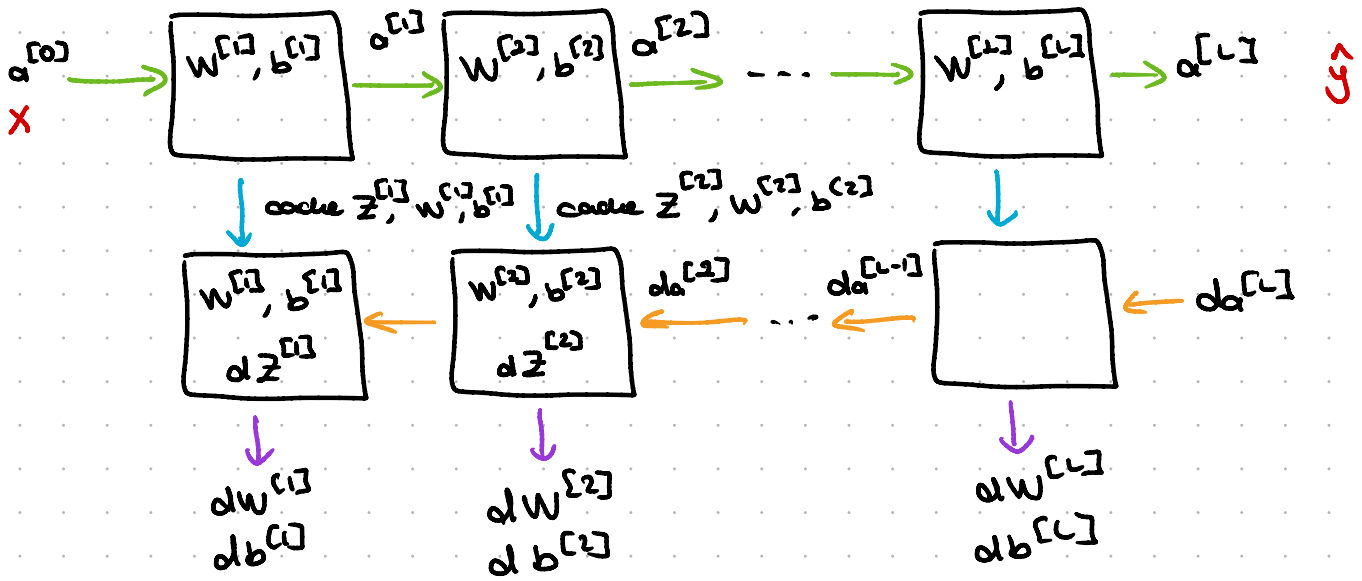
Building Blocks of DNNs

layer l: $W^{[l]}$, $b^{[l]}$

Forward: input $A^{[l-1]}$, output $A^{[l]}$, cache $Z^{[l]}$

Backward: input $dA^{[l]}$, $Z^{[l]}$ (cached), output $dA^{[l-1]}$, $dW^{[l]}$, $db^{[l]}$





BACKWARD PROPAGATION

For layer ℓ
- → input $da^{[\ell]}$
- → output $da^{[\ell-1]}, dW^{[\ell]}, db^{[\ell]}$

Main Equations:

(1) $dz^{[\ell]} = da^{[\ell]} * g^{[\ell]'}(z^{[\ell]})$

$dw^{[\ell]} = dz^{[\ell]} \, a^{[\ell-1]T}$

$db^{[\ell]} = dz^{[\ell]}$

(2) $da^{[\ell-1]} = w^{[\ell]T} dz^{[\ell]}$

by (1)(2) $\Rightarrow$ $dz^{[\ell]} = w^{[\ell+1]T} dz^{[\ell+1]} * g^{[\ell]'}(z^{[\ell]})$

Vectorized:

$$dZ^{[\ell]} = dA^{[\ell]} * g^{[\ell]'}(Z^{[\ell]})$$

$$dW^{[\ell]} = \frac{1}{m} dZ^{[\ell]} A^{[\ell-1]T}$$

$$db^{[\ell]} = \frac{1}{m} np.sum(dZ^{[\ell]}, axis=1, keepdims=True)$$

$$dA^{[\ell-1]} = W^{[\ell]T} dZ^{[\ell]}$$

Initialized As:

$$dA^{[L]} = -\frac{Y}{A} + \frac{(1-Y)}{(1-A)}$$ (we had proved before)

Parameters vs Hyperparameters

Parameters: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, \ldots$

Hyperparameters:
- Learning rate α
- # iterations
- # hidden layers L
- # hidden units
- choice of activation fn

} these are the parameters that control our parameters