→ Getting your matrix dimension right

$$z^{[1]} = w^{[1]}x + b^{[1]}$$

$(3,1)$ $\quad$ $(2,1)$

$(n^{[1]},1)$ $\quad$ $(n^{[0]},1)$

$(3,n^{[0]})$

$z = (n^{[1]},1)$

$w = (n^{[1]},n^{[0]})$

$x = (n^{[0]},1)$

$w^{[l]} = (n^{[l]},n^{[l-1]})$

$b^{[l]} = (n^{[l]},1)$

$$\boxed{z^{[1]} = w^{[1]}x + b^{[1]}}$$

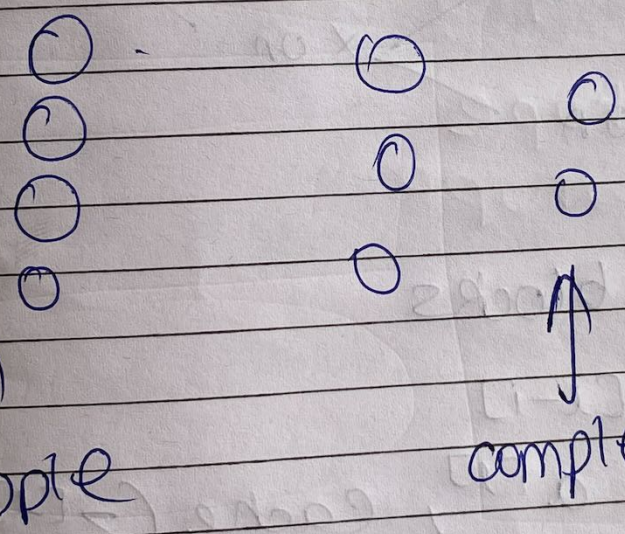
$(n^{[1]},m)$ $\quad$ $(n^{[1]},n^{[0]})$ $\quad$ $(n^{[0]},m)$ $\quad$ $(n^{[1]},m)$

$z^{[l]}, a^{[l]} : (n^{[l]}, 1)$

$Z^{[l]}, A^{[l]} : (n^{[l]}, m)$

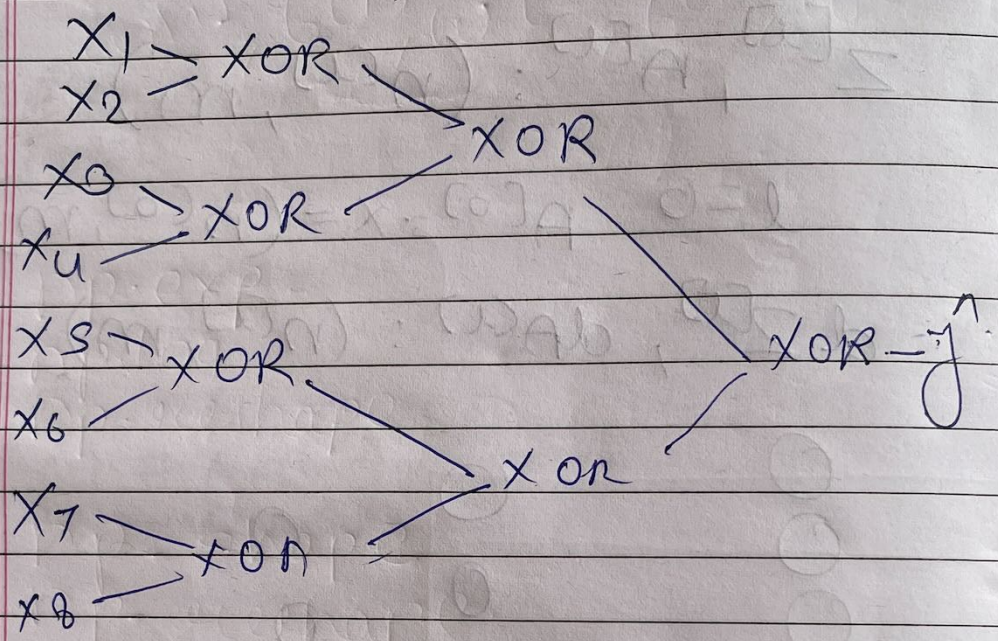
$l = 0 \quad A^{[0]} = X = (n^{[0]}, m)$

$dZ^{[l]}, dA^{[l]} : (n^{[l]}, m)$




relatively small L-layer deep neural network then we require exponentially more hidden units to compute.

$x_1$ XOR $x_2$ XOR $x_3$ XOR ... XOR $x_n$




* Building blocks

1) Input $a^{[l-1]}$
2) Output $a^{[l]}$, cache $(z^{[l]})$

1) $z^{[l]} = w^{[l]} a^{[l-1]} + b^{[l]}$

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

PAGE No.
DATE / /

Vectorized.

$$Z^{[l]} = W^{[l]} \cdot A^{[l-1]} + b^{[l]}$$
$$A^{[l]} = g^{[l]}(Z^{[l]})$$

Backward Propagation

1) Input $da^{[l]}$

2) Output $da^{[l-1]}, dw^{[l]}, db^{[l]}$

$$dZ^{[l]} = da^{[l]} * g^{[l]\prime}(Z^{[l]})$$

$$dw^{[l]} = dZ^{[l]} \cdot a^{[l-1]T}$$

$$db^{[l]} = dZ^{[l]}$$

$$da^{[l-1]} = w^{[l]T} \cdot dZ^{[l]}$$

$$dZ^{[l]} = w^{[l+1]T} dZ^{[l+1]} * g^{[l]\prime}(Z^{[l]})$$

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

# Course 2 - Week 1.

PAGE No.
DATE / /

* hyperparameters

idea
code
experiment.

Train ~~dev~~ Test.

60/20/20.
70/30
98/1/1
99/0.5/0.5

② Bias/variance

underfitting (high bias).

overfitting (high variance).

| | | | | |
|---|---|---|---|---|
| TE | 1 | 15 | 15 | 0.5 |
| DE | 11 | 14 | 30 | 1 |
| | high variance | high Bias | Both | Best. |

PAGE No.
DATE / /

③ high bias → NN bigger.

④ Algorithm has high variance → i) more data
ii) Regularization
iii) Model suitable

while (Bias high & variance high)

→ Regularization

i) reduces variance

L1 :

$||w|| = \text{sum}(|w[i,j]|)$

⇒ sum of all abs w.

L2 (Frobenius)

$||w||^2 = \text{sum}(|w[i,j]|^2)$

⇒ sum of all w squared

$$J(w,b) = \frac{1}{m} \text{Sum}(L(\hat{y}(i), y(i)))$$
$$+ \frac{\lambda}{2m} \times \text{Sum}(w[i]^2)$$

L1 make lot of w values zero, which makes model size smaller (compression) of model.

$\lambda$ (Regularization parameter)

lambd ✓ lambda ✗ (fn)

↓

Py Reymond.

$$\left[ \|w^{[l]}\|^2 = \sum_{i=1}^{n^{[l-1]}} \sum_{j=1}^{n^{[l]}} (w_{ij}^{[l]})^2 \right]$$

Frobenium norm

dw (from back propagation)

$$dw^{[l]} = (\text{from backprop}) + \frac{\lambda}{m} w^{[l]}$$

→ "Weight Decay"

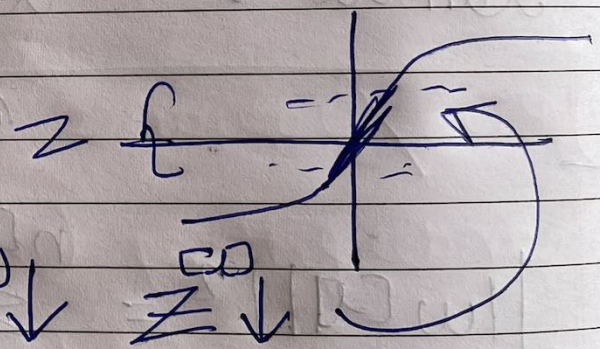
$$w^{[l]} = w^{[l]} - \alpha dw^{[l]}$$

---

$$\lambda \uparrow \longrightarrow w^{[l]} \approx 0$$

→ much simplified NN.

→ ignores complexities.




$$\lambda \uparrow \quad w^{[l]} \downarrow \quad z^{[l]} \downarrow$$

Roughly linear

linear → whole network is linear.

Tip
when implementing,

$$J(\ldots) = \sum_i \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m}\sum \|w^{(l)}\|_F^2$$

---

## Dropout Regularization Technique

Eliminating nodes

O.S change → keeping
→ eliminating

diminished network — reduced network

{on each example}