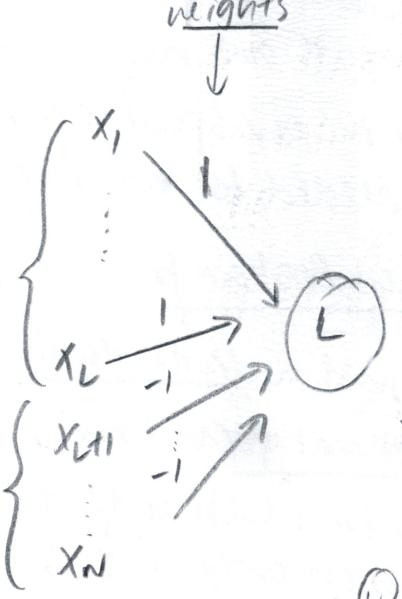
11-785 L2

- Review

(A1) - universal AND -



$$\left(\bigwedge_{i=1}^{L} x_i\right) \wedge \left(\bigwedge_{i=L+1}^{N} \bar{x}_i\right)$$

A B

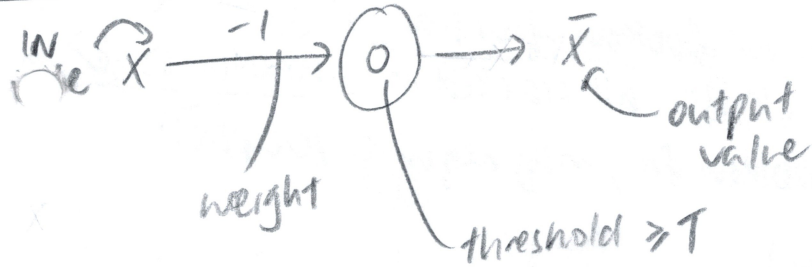
$Y=1 \Rightarrow (x_1,\ldots,x_L) = 1 \quad 1,2,\ldots,L$

$(x_{L+1},\ldots,x_{N+1}) = 0 \quad L,\ldots,N+1$

(W): Lectures unclear here

- (W): negative on weight $\Rightarrow \bar{x}$

- BR: input weights output val.



$x = 0 \quad x = 1$

$\bar{x} = 1 \quad \bar{x} = 0$ 'NOT'

(A2) - universal OR

(W): Okay, I vaguely understood, no need to get too forensic

(*) - Perceptron can model :- (single perceptron)

i) AND.

ii) NOT

iii) OR

iv) universal AND

v) universal OR

vi) generalised majority.

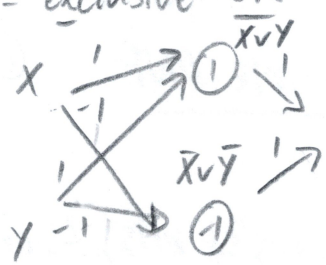
- But not vii) XOR

(*) XOR require an MLP i.e. 3 perceptrons (2 hidden)

- XOR

X	Y	X⊕Y
0	0	0
0	1	1
1	0	1
1	1	0

- exclusive OR (either X or Y but 'not both')



(*) MLP → universal Boolean fns:-
- i.e. any function over any no of inputs and any no. of outputs.

- This is very loose; can we do better in our specification.

Q1) How many layers (depth) are needed for an MLP to be universal Boolean fn.

A) 1 hidden layer MLP is a universal Boolean fn

why: Any Boolean function is just a truth table
- Express MLP in disjunctive normal form (DNF)
- Show all input combinations for which output Y is 1.
- There are 6 conditions for which output Y is 1 (representing a row of truth table, and particular realisations of X_i)
  6 clauses + 1 → 7 units
- 6 rows ⇒ 6 disjunctive clauses for Boolean function. (output layer)
- 1 perceptron for each disjunctive clause, all stacked in 1 hidden layer
- No matter how complex the Boolean fn; only require perceptron with 1-hidden layer (width)

Q2) How many perceptrons are required, if 1-hidden layer is used?

Karnaugh maps → topological representation of bit patterns
- 2 adjacent squares differ in 1-bit
- each square represents 4 bit input comb.
(*) - Highlighted blocks correspond to no. of DNF clauses/rows in table (7 in this case)

- Group cells/clauses in Karnaugh map, giving reduced DNF form.
- reduced DNF form reduces DNF clauses (7) → reduced DNF (3) clauses
∴ MLP for the example → only 3 neurons required in hidden layer (for this example)
(i.e. algo → find reduced DNF no. of clauses → no. of neurons in hidden)

(*) - No need for 1 percep. for every DNF clause (row).
(input combo)

(?)

- via consideration of largest irreducible DNF

(W): relation between irreducible DNF & XOR

- 3D Karnaugh map → 6 variables
- Largest irreducible DNF - 64 combos of bits, 32 are 1 → 32 neurons.

(A): for N-inputs, 1 hidden layer, require 2^{N-1} neurons/perceptrons in 1 hidden layer.

- Exponential

(W): depth - no. of layers
width - no. of units in layer

use Karnaugh map to state how many perceptrons required to rep XOR for 1 hidden layer.

XOR - 3 perceptrons, 1 hidden layer; (or 2 percep and alt. config (see slides))

(Q3) How many units/neurons/perceptrons will we need (width) if we use multiple layers.

(A1) - An XOR needs 3 perceptrons; XOR of N variables requires 3(N-1) perceptrons in a deep network, arranged in $2\log_2 N$ layers (width) (depth)

- using only K hidden layers requires $O(2^{CN})$ neurons in K^{th} layer, where $C = 2^{-(K-1)/2}$

- Because output is XOR of outputs of $(K-1)^{th}$ layer

(*) Reducing no. of layers below minimum i.e. ($2\log_2 N$ layers) will result in exponential growth of no. neurons required to compute function

(*) Network with fewer than minimum no. of required neurons cannot compute the function (at a given depth?)

(*) BR: Optimal depth depends on size of input.

(W): I am going to be sceptical about these claims until I see the results in papers (i.e. treat it as heresay)

(A6): Deep Boolean MLPs that scale linearly in no. inputs can become exponentially large if using 1 layer (leading to superexponential no. of parameters to estimate)

(*) BR: Any function you are trying to compute can have prelim seq. of operations after which you get a set of XORs; at that point if you limit no. of layers from there on → expon. large network

(*) Few extra layers (depth) can greatly reduce network size (width)

(A8) - A XOR parity problem

Lecture 2

- We are using threshold gates, not Boolean circuits
  → more powerful.

- composition of  using 1 hidden layer

(?) - Poor explanation (this header will)

(w): Limit argument:

For square d.b. ; yellow 3 → infinite area strips
pentagon -11-; yellow 4 → finite area
hexagon d.b; yellow 5 → finite area (smaller)

N-gon d.b.; yellow (N-1) → finite area (smaller)

(*) Increasing no. of sides N reduces area outside polygon that have

$$\frac{N}{2} < \Sigma_i y_i < N$$

(*) circle net → increase no. of neurons (N) / no. of sides of modellable polygon (N) → ∞

- sum is $\frac{N}{2}$ inside circle d.b.

- 0 outside almost everywhere

- Normalise → $\sum_{i=1}^{N} y_i \quad : \quad \sum_{i=1}^{N} y_i - \frac{N}{2} \geq 0$

Intersection of neurons to get a cylinder
- Approx. pentagon with ~~a circle~~ lots of circles in diffent locations.
- one subnet for each cylinder ; 'add'

(*) can arbitrarily approximate (using analysis style reasoning)

2) (*) Hence in this fashion, we can arbitarily model a classification/decision boundary using a 1-hidden layer MLP

- (w): This is the sense in which they are universal classifiers

(*) Neural network needed nearly $N \to \infty$ no. of neurons for arbitrary precision.

- ~~In fact,~~

BR: deeper networks require fewer neurons

(*) optimal depth → formal results on class. in terms of arithmetic circ.

- Different scenario:
  - generic nets
  - threshold activation
- (*) Naively:- 1 hidden layer MLP → infinite no. neurons
  2 hidden layers MLP → 57 hidden neurons

- 1st layer - 16 neurons
  (8 lines /) ; (8 lines \)
- 2nd layer
  - 40 neurons ~~in~~ 2 for each square inside checkerboard

} 57

+ output

- Also, this is just $Y_1 \oplus \dots \oplus Y_{16}$ (XOR translation) → 61 neurons

---

More complex:- (checkerboard)

- 1 hidden layer MLP → infinite neurons
  2 — " — MLP - 609
- 12 hidden layer MLP - 253
  (XOR net)

deeper

(*) As dimensionality ↑, difference in size between optimal XOR and shallow nets ↑. with complexity, input dim.

(*)

---

3. MLP as universal approx.

- 1-hidden layer MLP → universal fn. approx (but exp. wide)
- deeper networks → fewer neurons for same approx. error

- (01): Is activation function discrete/step-like or continuous (ReLU)
- Heuristic arguments regarding activation/info propagation due to info. gating from threshold
- Eg. ReLU retains info about how far from boundary; can use fewer no. neurons
(could (i.e. above arguments based on threshold activ.)