BITS Pilani

Pilani | Dubai | Goa | Hyderabad

Deep Neural Networks

Module # 1 : MLP as Universal Approximators

DL Team, BITS Pilani

The author of this deck, Prof. Seetha Parameswaran, is gratefully acknowledging the authors who made their course materials freely available online.

Table of Contents

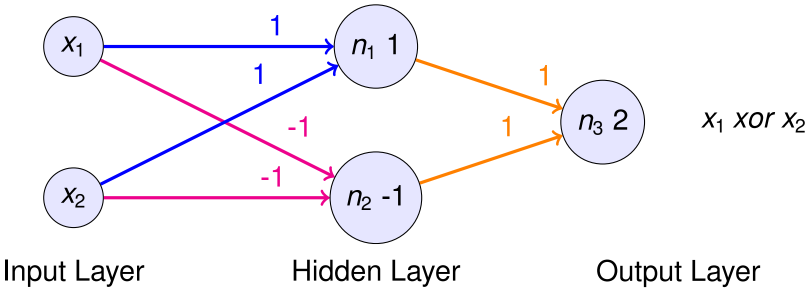
Power of MLP

MLP can be used for

- classification
  - binary
  - multiclass
- regression
  - real output
- representing complex decision boundaries
- representing boolean functions

MLP is called as Universal Approximator for the above reasons.

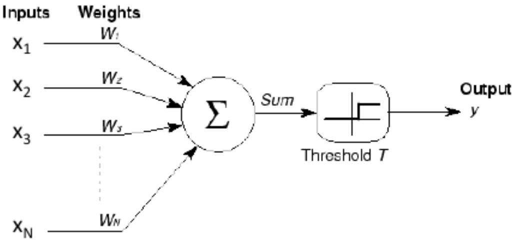
MLP for XOR gate

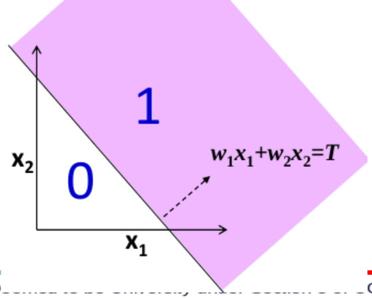


Perceptron with real inputs

- $x_1, \ldots, x_N$ are real valued.
- $W_1, \ldots, W_N$ are real valued.
- Unit ”fires” if weighted input exceeds a threshold.

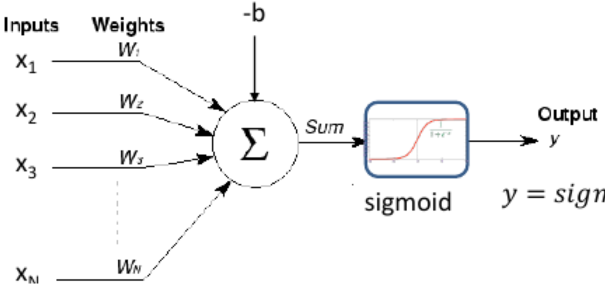
$$\hat{y} = \begin{cases} 1 & \textit{if } \sum_i w_i x_i \geq T \\ 0 & \textit{otherwise} \end{cases}$$





Perceptron with real outputs

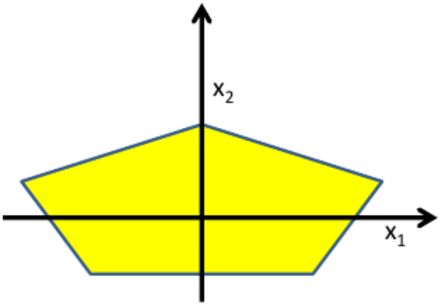
- $x_1, \ldots, x_N$ are real valued.
- $W_1, \ldots, W_N$ are real valued.
- The output can also be real valued. – Sometimes viewed as the "probability" of firing.



$$y = sigmoid(\sum_i w_i x_i - b)$$

MLP for Complicated Decision Boundaries

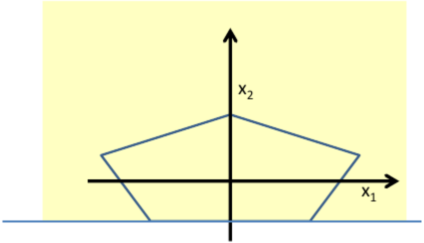
- Build a network of units with a single output that fires if the input is in the coloured area.

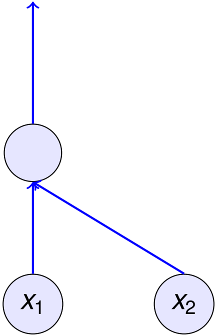


Can now be composed into "networks" to compute arbitrary classification "boundaries"

MLP for Complicated Decision Boundaries

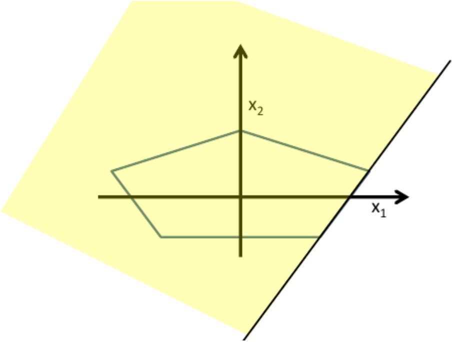
- The network must fire if the input is in the coloured area.

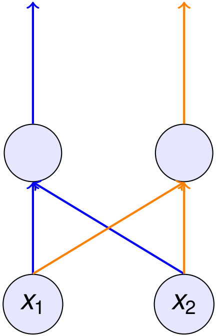




MLP for Complicated Decision Boundaries

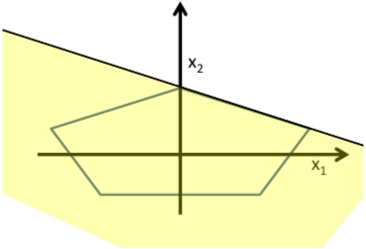
- The network must fire if the input is in the coloured area.

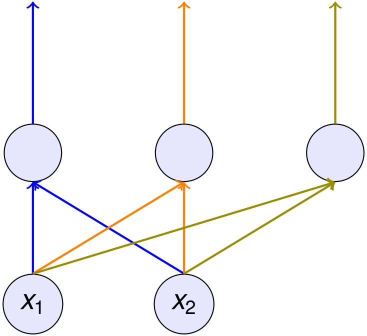




MLP for Complicated Decision Boundaries

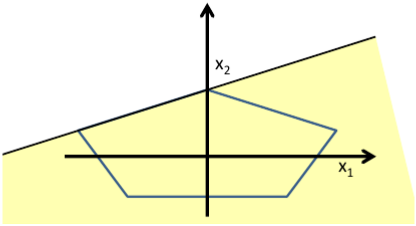
- The network must fire if the input is in the coloured area.

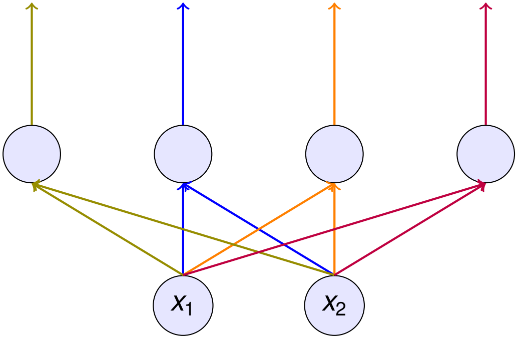




MLP for Complicated Decision Boundaries

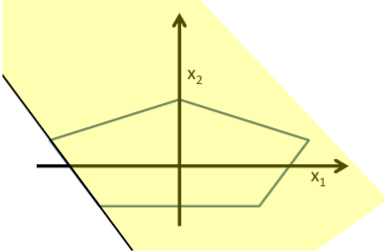
- The network must fire if the input is in the coloured area.

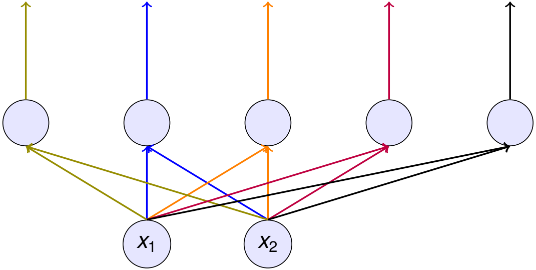




MLP for Complicated Decision Boundaries

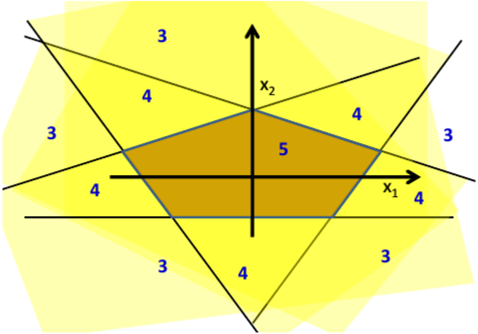
- The network must fire if the input is in the coloured area.



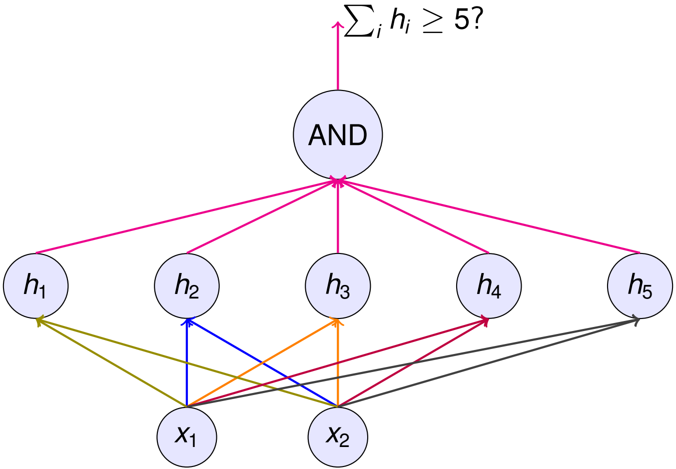


MLP for Complicated Decision Boundaries

- The network must fire if the input is in the coloured area.

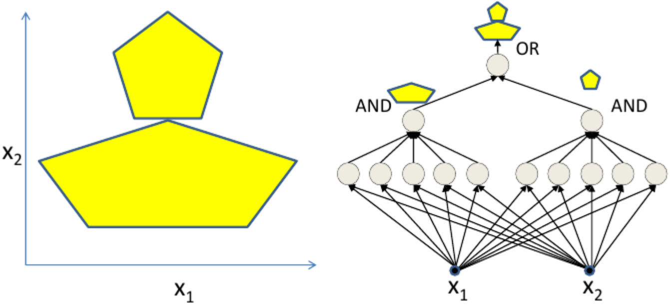


MLP for Complicated Decision Boundaries



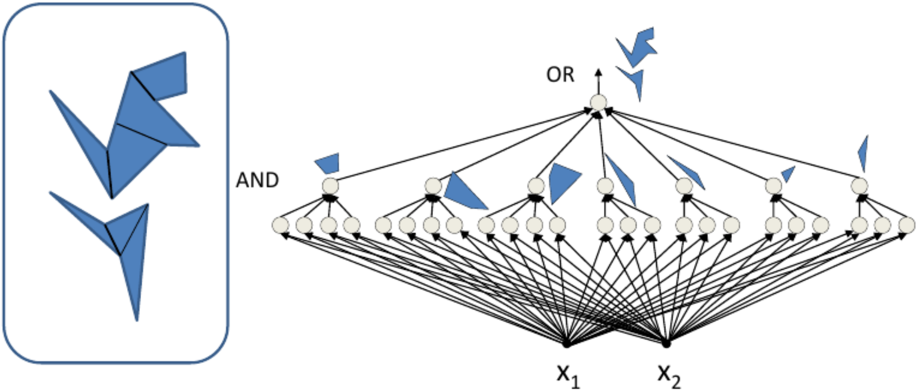
MLP for Complicated Decision Boundaries

- Network to fire if the input is in the yellow area.
  - ”OR” two polygons
  - A third layer is required



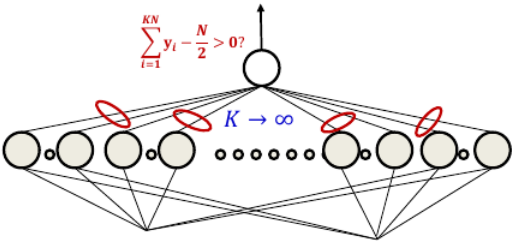
MLPs for Complex Decision Boundaries

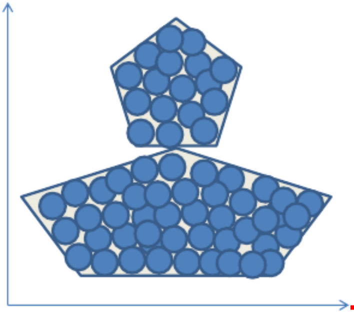
- MLPs can compose arbitrarily complex decision boundaries with only one hidden layer.



MLPs for Arbitrary Classification Boundary

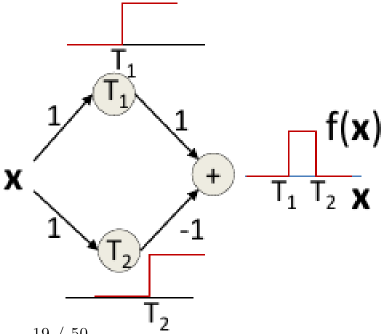
- Just fit in an arbitrary number of circles
  - More accurate approximation with greater number of smaller circles.
  - Can achieve arbitrary precision
- MLPs can capture any classification boundary.
- Deeper networks can require far fewer neurons.





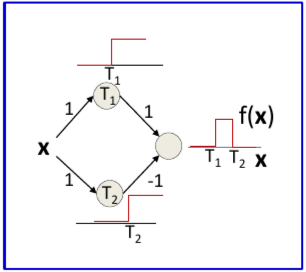
MLP AS CONTINUOUS-VALUED REGRESSION

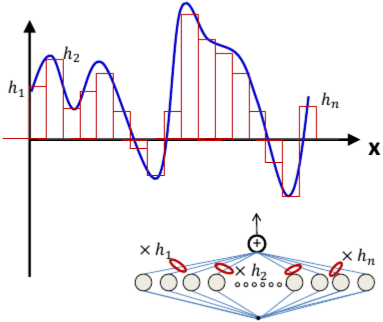
- A simple 3-unit MLP with a ”summing” output unit can generate a ”square pulse” over an input.
  - Output is 1 only if the input lies between T1 and T2.
  - T1 and T2 can be arbitrarily specified



MLP as Continuous-Valued Regression

- A simple 3-unit MLP with a "summing" output unit can generate a "square pulse" over an input.
- An MLP with many units can model an arbitrary function over an input.
- This generalizes to functions of any number of inputs.

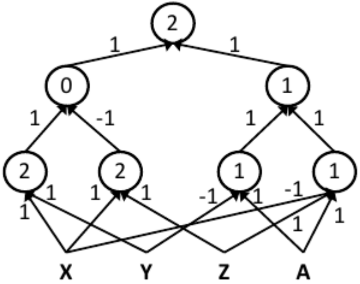




MLPs are Universal Boolean Functions

- MLPs can compute more complex Boolean functions.
- MLPs can compute any Boolean function; since they can emulate individual gates.

$$((A\&\bar{X}\&Z)|(A\&\bar{Y}))\&((X\ \&\ Y)|\overline{(X\&Z)})$$



How many layers for a Boolean MLP?

- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

How many layers for a Boolean MLP?

- Express any Boolean function in disjunctive normal form.

Truth table shows *all* input combinations for which output is 1

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$

How many layers for a Boolean MLP?

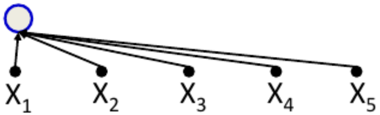
- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



How many layers for a Boolean MLP?

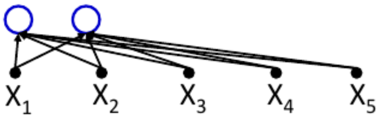
- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



How many layers for a Boolean MLP?

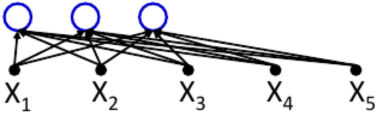
- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



How many layers for a Boolean MLP?

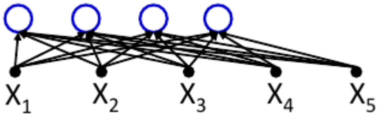
- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



How many layers for a Boolean MLP?

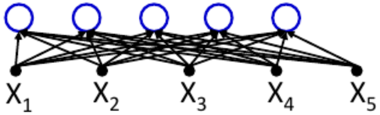
- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



How many layers for a Boolean MLP?

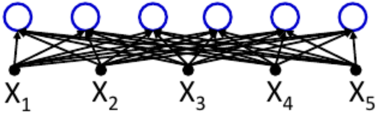
- Express any Boolean function in disjunctive normal form.

Truth Table

X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

Truth table shows *all* input combinations for which output is 1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



How many layers for a Boolean MLP?

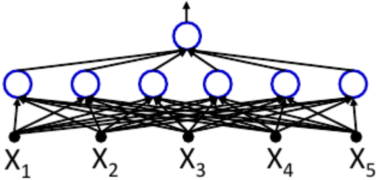
- Express any Boolean function in disjunctive normal form.

Truth table shows *all* input combinations for which output is 1

Truth Table

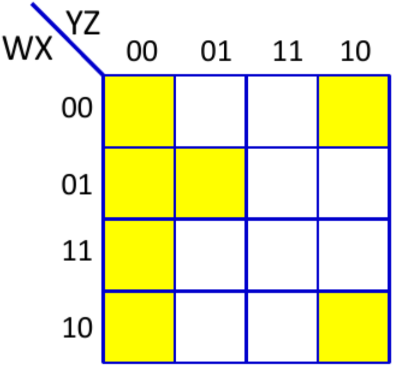
X_1	X_2	X_3	X_4	X_5	Y
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1

$$Y = \bar{X}_1\bar{X}_2X_3X_4\bar{X}_5 + \bar{X}_1X_2\bar{X}_3X_4X_5 + \bar{X}_1X_2X_3\bar{X}_4\bar{X}_5 + X_1\bar{X}_2\bar{X}_3\bar{X}_4X_5 + X_1\bar{X}_2X_3X_4X_5 + X_1X_2\bar{X}_3\bar{X}_4X_5$$



- Any truth table can be expressed in this manner.
- A one-hidden-layer MLP is a Universal Boolean Function.

Reducing a Boolean Function



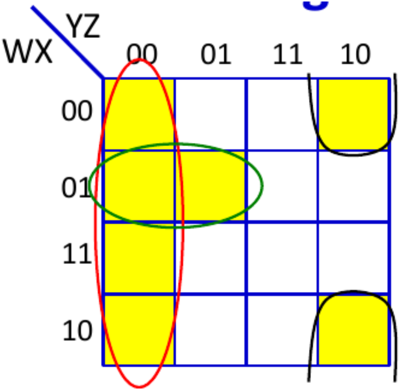
This is a "Karnaugh Map"

It represents a truth table as a grid
Filled boxes represent input combinations for which output is 1; blank boxes have output 0

Adjacent boxes can be "grouped" to reduce the complexity of the DNF formula for the table

Reducing a Boolean Function

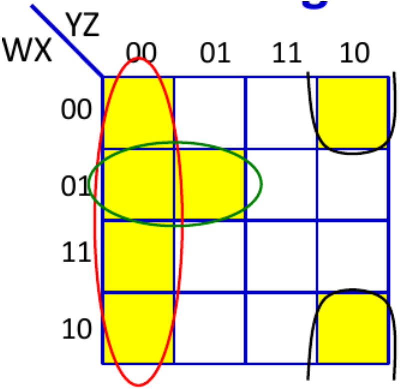
- Use K-map to reduce the DNF to find groups and express as reduced DNF.



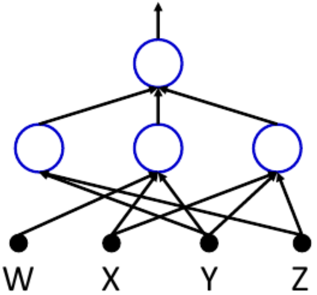
$$O = \bar{Y}\bar{Z} + \bar{W}X\bar{Y} + \bar{X}Y\bar{Z}$$

Reducing a Boolean Function

- Use K-map to reduce the DNF to find groups and express as reduced DNF.



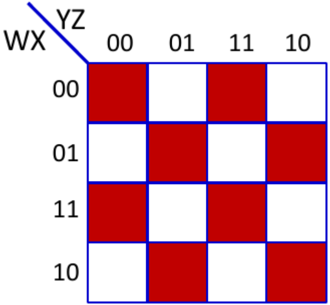
$$O = \bar{Y}\bar{Z} + \bar{W}X\bar{Y} + \bar{X}Y\bar{Z}$$



- *Reduced* DNF form:

Largest Irreducible DNF?

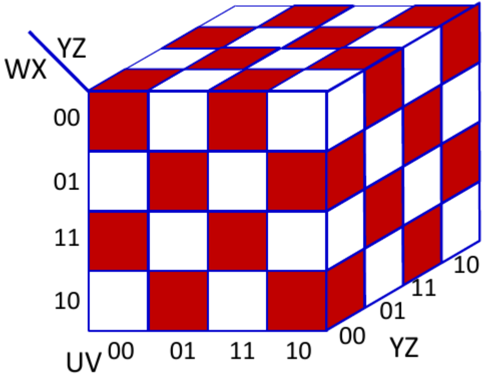
- What arrangement of ones and zeros simply cannot be reduced further?



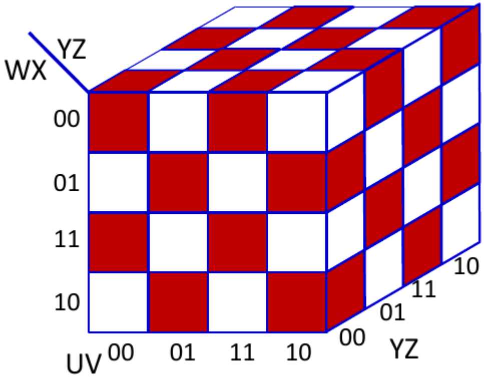
- How many neurons in a DNF (one-hidden-layer) MLP for this Boolean function?

Width of a Single-Layer Boolean MLP

- How many neurons in a DNF (one-hidden-layer) MLP for this Boolean function of 6 variables?

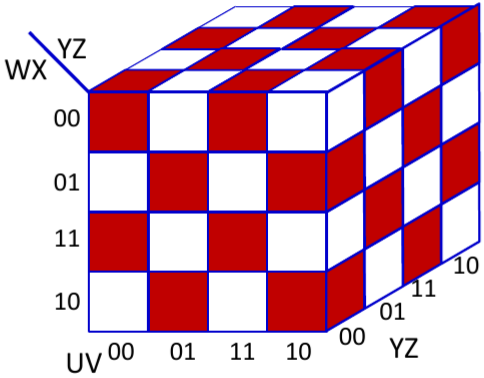


Width of a Single-Layer Boolean MLP



- In general, XOR of N variables will require 2^{N-1} perceptrons in single hidden layer. **Exponential in N**

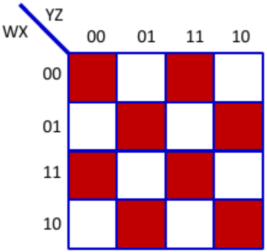
Width of a Single-Layer Boolean MLP



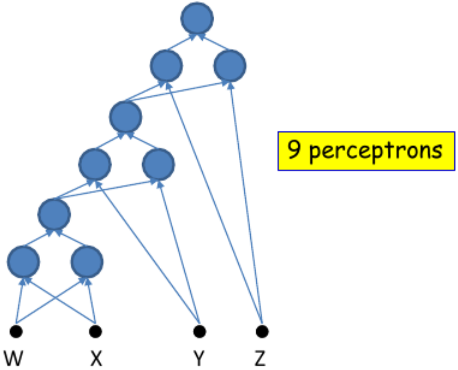
- In general, XOR of N variables will require 2^{N-1} perceptrons in single hidden layer.
- **How many units if we use multiple layers?**

Width of a Deep Boolean MLP

- An XOR needs 3 perceptrons.

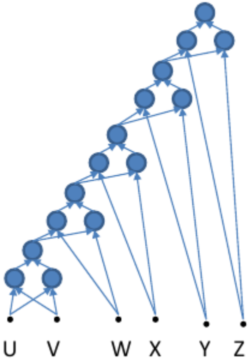


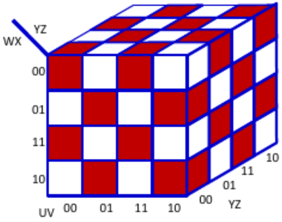
$$O = W \oplus X \oplus Y \oplus Z$$



- This network will require 3x3 = 9 perceptrons.

Width of a Deep Boolean MLP



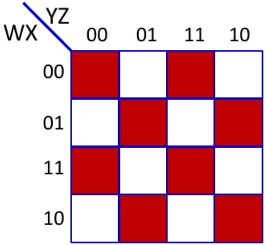


$$O = U \oplus V \oplus W \oplus X \oplus Y \oplus Z$$

15 perceptrons

- This network will require 3x5 = 9 perceptrons.
- In general, the XOR of N variables will require $3(N-1)$ perceptrons.

Width of a Deep Boolean MLP

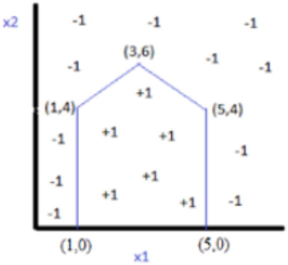


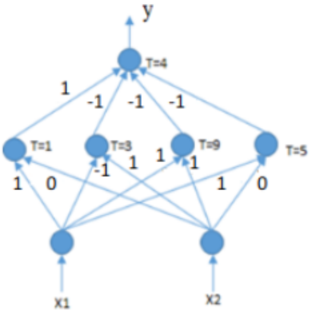
- Single hidden layer: Will require $2^{N-1} + 1$ perceptrons in all (including output unit). Exponential in N.
- Deep network: Will require $3(N-1)$ perceptrons in a deep network. Linear in N.
  This can be arranged in only $2log_2(N)$ layers.

Summary of MLP

- MLPs are connectionist computational models,
  - Individual perceptrons are computational equivalent of neurons.
  - The MLP is a layered composition of many perceptrons.
- MLP can model universal Boolean functions
  - Individual perceptrons can act as Boolean gates.
  - Networks of perceptrons are Boolean functions.
- MLPs are Boolean machines.
  - They represent Boolean functions over linear boundaries.
  - They can represent arbitrary decision boundaries.
  - They can be used to classify data.
- MLP can model continuous valued functions.
- MLPs as universal classifiers.
- MLPs as universal approximators.

Exercise





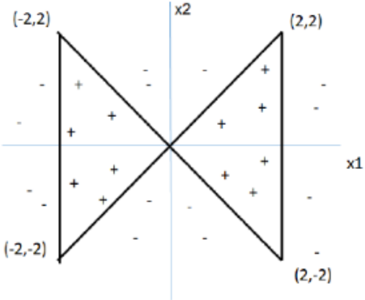
- **Note: Different Choices of weights and bias are possible.**
- Left hidden node implements x1=1 line
- Right hidden node implements x1=5 line
- 2nd node from left implements x2=x1+3 and 3rd from left implements x1+x2=9
- For + class, output of left hidden node = +1, for other nodes output = -1

Exercise

(x1, x2) are input features and target classes are either +1 or -1 as shown in the figure.

A. What is the minimum number of hidden layers and hidden nodes required to classify the following dataset with 100% accuracy using a fully connected multilayer perceptron network? Step activation functions are used at all nodes, i.e., output=+1 if total weighted input >= bias b at a node, else output = -1.

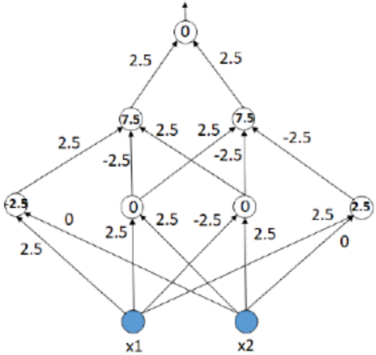
B. Show the minimal network architecture by organizing the nodes in each layer horizontally. Show the node representing x1 at the left on the input layer. Organize the hidden nodes in ascending order of bias at that node. Specify all weights and bias values at all nodes. Weights can be only -2.5, 2.5 or 0, and bias +ve/-ve multiples of 2.5.



Exercise - Solution

A. 2 hidden layers, 4 nodes in first hidden layer and 2 nodes in second hidden layer needed.

B.

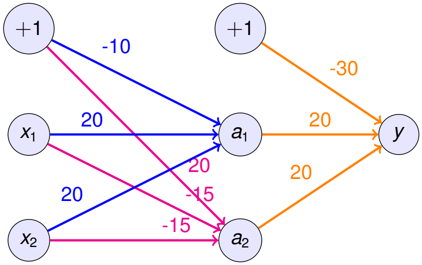


EXERCISE

An XOR cannot be represented using a single perceptron. If this statement is true, demonstrate the alternative way to represent XOR using an ANN.

Exercise - Solution

An XOR cannot be represented using a single perceptron as the XOR has non-linear decision boundary. To represent non-linear decision boundary we need a multi-layer perceptron with one hidden layers.



$$a_k = sign(w_{k1}x_1 + w_{k2}x_2 + b)$$

x_1	x_2	a_1	a_2	y
0	0	$-10 \sim 0$	$20 \sim 1$	$-10 \sim 0$
0	1	$10 \sim 1$	$5 \sim 1$	$10 \sim 1$
1	0	$10 \sim 1$	$5 \sim 1$	$10 \sim 1$
1	1	$30 \sim 1$	$-30 \sim 0$	$-10 \sim 0$

Exercise

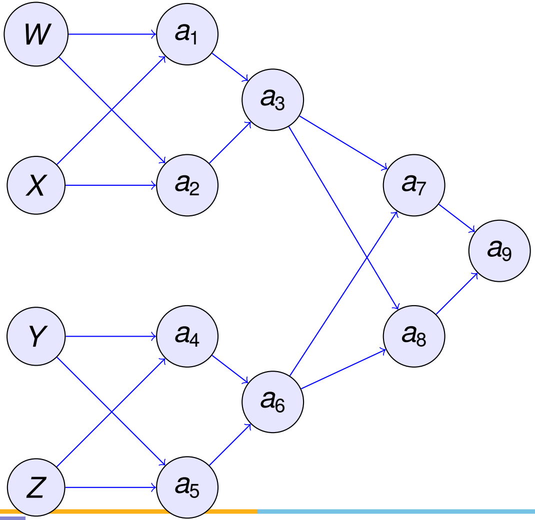
How many perceptrons are required to represent $W \oplus X \oplus Y \oplus Z$?

Exercise - Solution

An XOR cannot be represented using a single perceptron as the XOR has non-linear decision boundary. To represent non-linear decision boundary we need a multi-layer perceptron with one hidden layers.

- Traditional method:
  - Number of variables $n = 4$
  - Number of perceptrons = $2^n - 1 = 2^4 - 1 = 15$
- Alternate method:
  - Layer 1: $A = W \oplus X$
  - Layer 2: $B = Y \oplus Z$
  - Layer 3: $C = A \oplus B$
  - Number of perceptrons = $3(n-1) = 3 * (4-1) = 9$
  - Number of layers = Width = $2 \log_2 n = 2 \log_2 4 = 4$

Exercise - Solution



Further Reading

1. Dive into Deep Learning (T1)
2. `http://mlsp.cs.cmu.edu/people/rsingh/docs/Chapter1_Introduction.pdf`
3. `http://mlsp.cs.cmu.edu/people/rsingh/docs/Chapter2_UniversalApproximators.pdf`

Thank You!