Lecture 2 - Neural Networks as Universal Approximators

Contents

3 Neural Networks as Universal Approximators

3.1 Recap

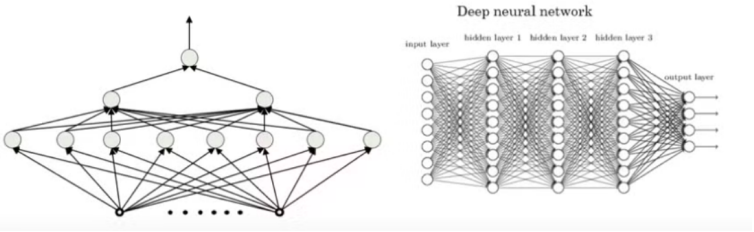
The function is like a box which takes in an input and gives an output. In a human these functions are computed by the human brain. Neural Networks mimic the structure of the human brain. The Brain is composed of a network of Neurons. The neuron's computational process can be approximated through a perceptron. Each perceptron obtains many inputs and computes a weighted sum of the inputs, compares that weighted sum to a threshold. If it exceeds a threshold, It outputs a 1, else it outputs a 0. We can emulate the large structure of the brain by creating a network of perceptrons.

3.2 Multi-Layered Perceptrons

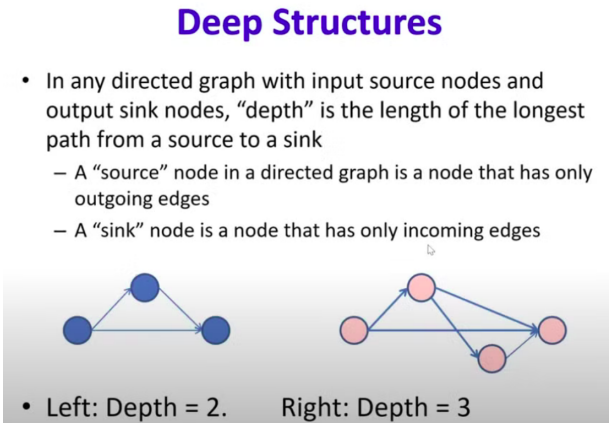
MLP is a network of perceptrons where the perceptrons are arranged in a layer wise manner, where we have perceptrons whose outputs are fed to other perceptrons whose outputs are fed to other

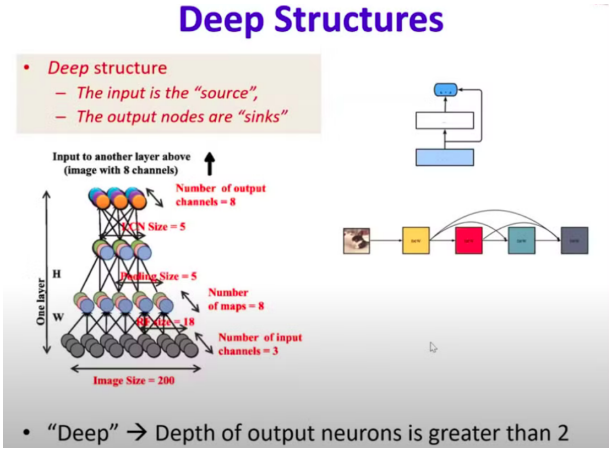
- Mark all true statements
  - 3x + 7y is a linear combination of x and y
  - 3x + 7y + 4 is a linear combination of x and y
  - 3x + 7y is an affine function of x and y
  - 3x + 7y + 4 is an affine function of x and y

perceptrons. **A Deep network Is a network whose depth of outer neurons is greater than 2**

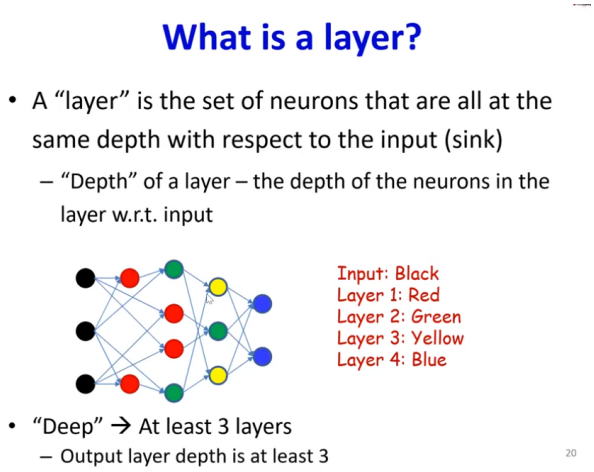


3.2.1 Depth





3.2.2 Layers



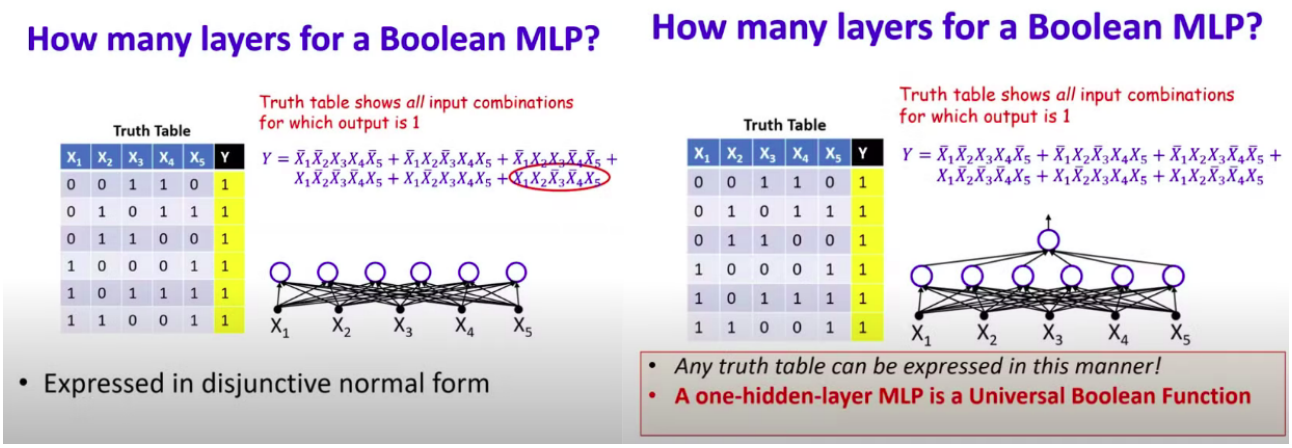
3.3 How Many Layers do we need?

We already know:

- The perceptron can model any simple binary Boolean gate.
- Even arbitrarily complex Boolean operations can be computed using a perceptron.
- While this may require long and complex circuits in traditional logic, a single perceptron can perform these computations.
- However, a single perceptron cannot compute an XOR gate. This requires 2 or 3 perceptrons.

How Many Layers do we need?

- For this we make a Truth Table and write it as a as a disjunctive normal form (DNF) formula. Skipping the details.
- Conclusion - Since any boolean function is a truth table, If i need to compute an arbitrary boolean function the minimum number of layers **I can get away with is 2 (Only 1 hidden layer)**.
- So meaning using only 1 hidden layer I can compute any Boolean Function.
- Input layer is not a layer as there is no computation done over there

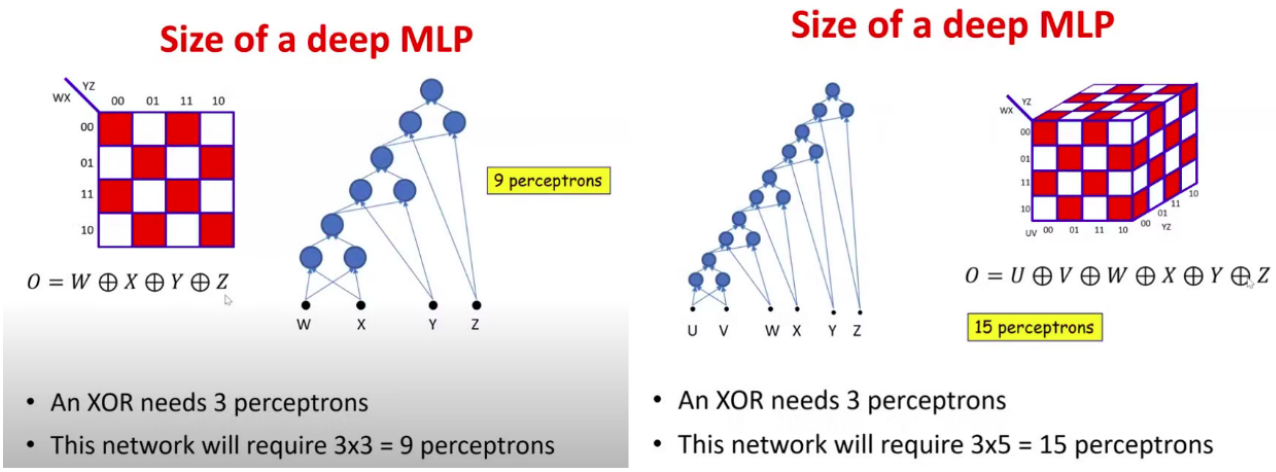


3.4 How Large can a Hidden Layer Get?

`Answer:` `Karnaugh Map` - Skipping it.
Conclusion: for N inputs, I'll need 2^{N-1} perceptrons in the hidden layer (minimum)

3.5 Single Hidden Layer vs Multiple Hidden layers

Below are 2 versions of the XOR Gate:



- A single hidden layer network will require $2^{N-1}+1$ perceptrons in all (+1 was for output unit) (which is exponential - parameter growth will also be exponentiol)
- The same function in a deep network would require $3(N-1)$ perceptrons (which is linear)

So is we have more layers we need less number of neurons. We are breaking a complex problem into simpler problems. **Less number of neurons mean less number of parameters required**

3.6 How deep a neural network can get?

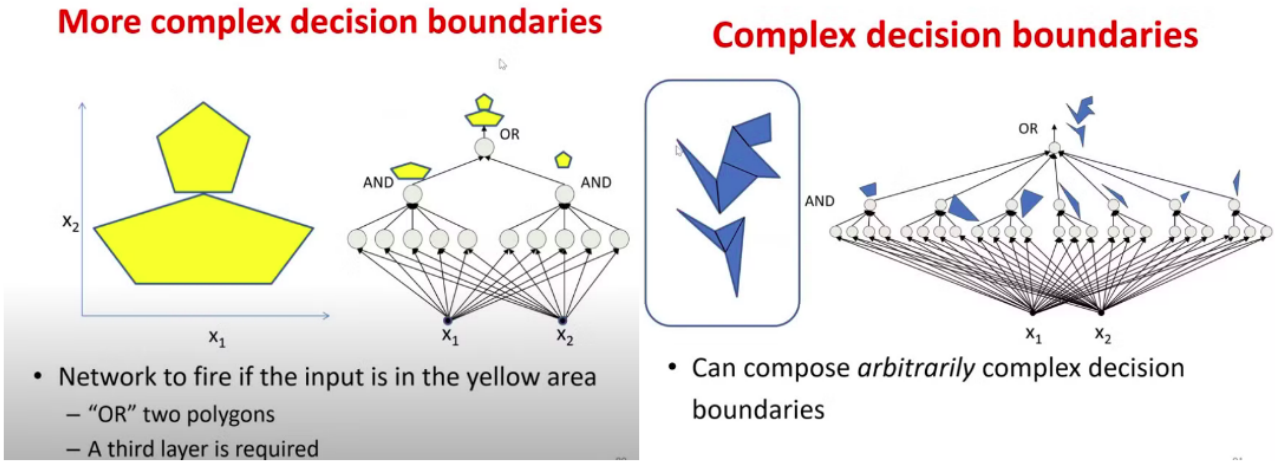
$$2 * log_2 N \tag{1}$$

Summary

- MLPs are universal Boolean machines. But given that Sufficiently wide or deep
- Even a network with a single hidden layer is a universal boolean machine
- But a single layered network may require an exponentiolly large number of perceptrons.
- Deeper networks may require far fewer neurons to express the same function and could be exponentiolly smaller.

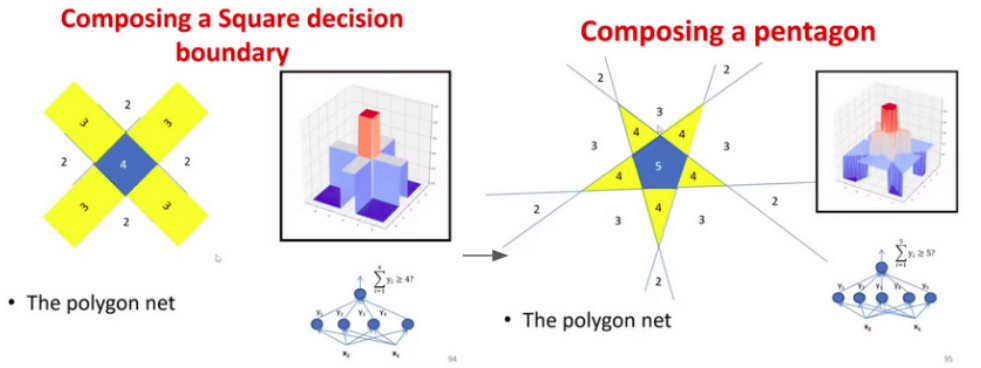
3.7 A single perceptron as a linear classifier

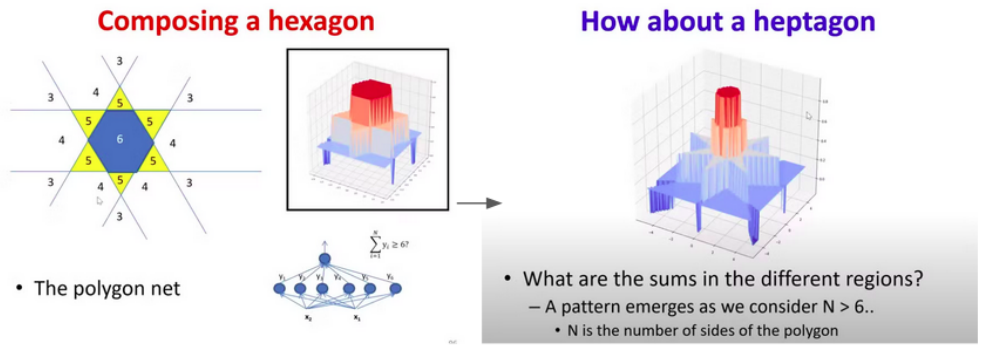
A Single perceptron is function which is a straight line - linear classifier. - linear classifier gives standard boolean gates. A Single perceptron cannot model XOR Gate - because a perceptron can go up but cannot go down.

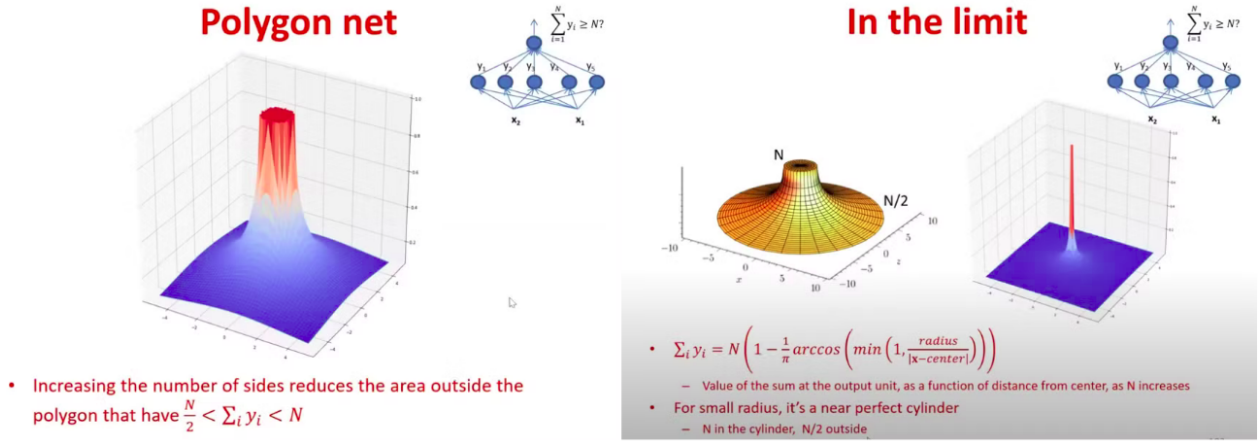


3.8 More Complex Shapes

Image 1 explains: Square decision boundary: Shows a 2D layout with a central square (value 4) surrounded by 4 squares (value 3) and 4 corner squares (value 2). A 3D representation shows how this creates a square "pillar" rising from a flat surface. Pentagon decision boundary: Similar layout but with a pentagon shape, values ranging from 2 to 5. 3D representation shows a more circular-like pillar. Hexagon decision boundary: Further refinement of the shape with a hexagon, values from 3 to 6. 3D representation shows an even more circular pillar. Heptagon (7-sided polygon): Only 3D representation is shown, looking even more circular.







Key points:

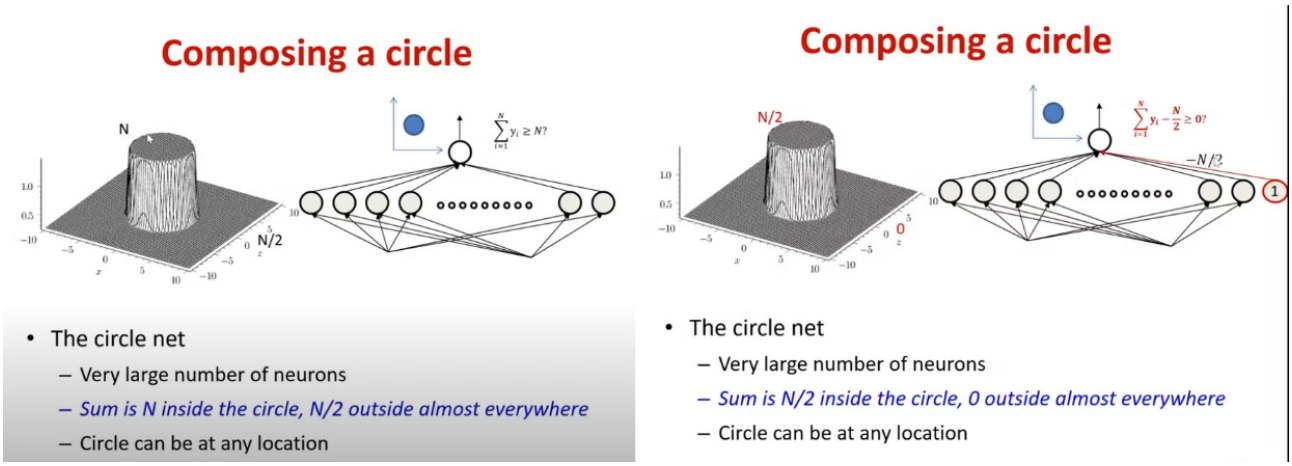
- Sum outside the boundary is consistently 3.
- 6 is the upper limit and 3 is the lower limit.
- As N (number of sides) increases, the shape gets smoother and more circular.

Image 2 elaborates on the concept: Polygon net: Shows how increasing the number of sides reduces the area outside the polygon that satisfies a certain condition. 3D representation shows a smoother, more circular pillar. In the limit: As the number of sides approaches infinity, the shape becomes a perfect cylinder. Provides a mathematical formula for the sum at the output as a function of

distance from the center. For small radius, it's nearly a perfect cylinder with N inside and N/2 outside. Left side: Shows a large number of neurons creating a circular net. Sum is N inside the circle, N/2 outside almost everywhere. Right side: Similar to left, but specifies sum is N/2 inside the circle, 0 outside almost everywhere. The final point notes that subtracting a bias term leaves an area of N/2 inside the circle and 0 outside. These slides demonstrate how neural networks can create complex decision boundaries, ultimately approximating smooth circular shapes as the number of nodes or sides increases.
The equation of the last image shows that the sum is N inside the circle and then rapidly falls of to $N/2$ outside.

3.9 Composing Circles

We have a circle with N inside and N/2 outside and if i add an extra bias term which subracts the N/2, then this becomes N/2 inside and 0 outside.

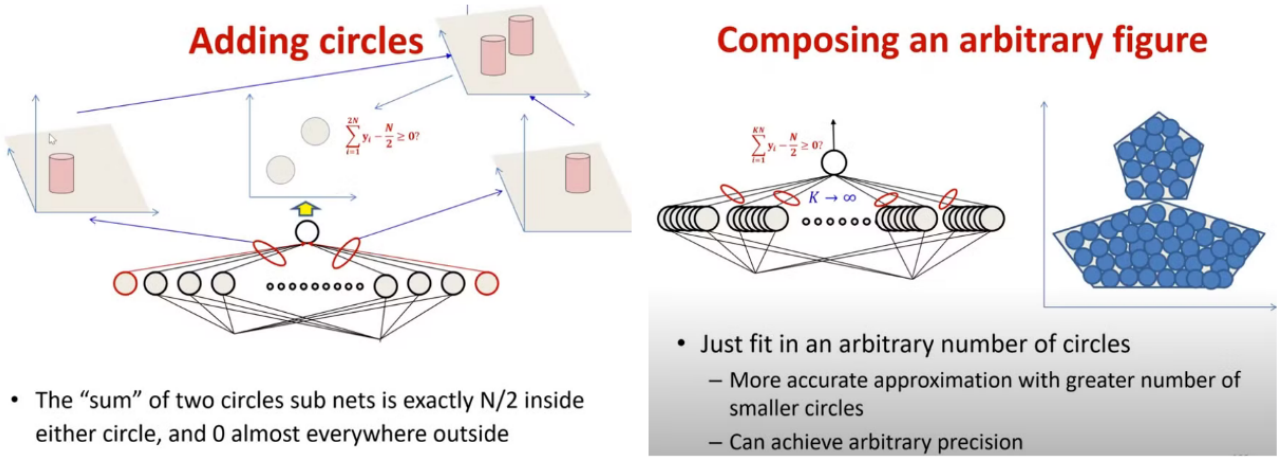


A very large number of neurons is required to capture the structure of the circle, which makes sense given the smoothness and symmetry of a circle. Inside the circle, the sum of the neuron activations is N (the total number of neurons). Outside the circle, the sum is N/2 (half the neurons are activated outside the circle). This suggests that the neurons are being used to approximate or simulate the circular shape, with their activation's varying depending on whether a point is inside or outside the circle. The circle is not fixed and can be at any location in the input space, which the neural network must adapt to. This reflects the flexibility of the network's ability to detect circles regardless of position.
In the second image The equation shown in the diagram subtracts a bias term $(\frac{N}{2})$ from the total sum of neuron outputs. This operation ensures that inside the circles, the output is exactly $\frac{N}{2}$, while outside the circles, the output is 0.

3.10 Adding Circles

The below equation is used to determine whether a point is inside the circle:

$$\sum y_i - \frac{N}{2} \geq 0 \tag{2}$$



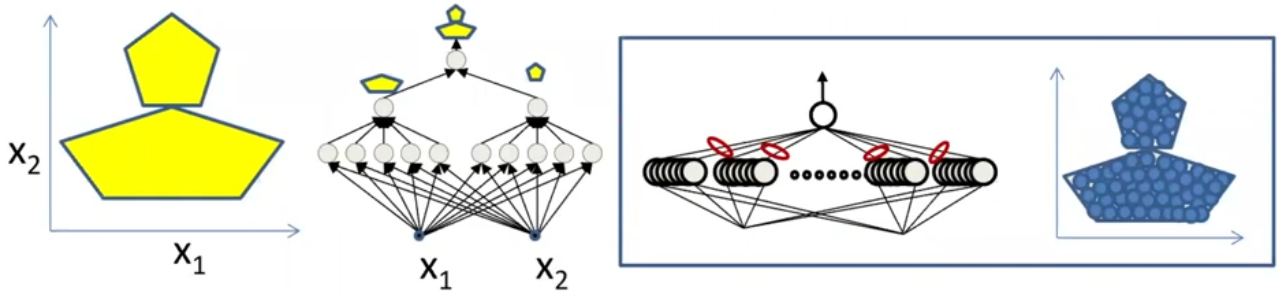
The "sum" of two circles' sub-networks is described as being exactly N/2 inside either circle, and 0 almost everywhere outside. This suggests how the network combines information to define circular boundaries. These are going to be 2 seperate cylinders as one does not interfere with the other.
The right side explains how this concept can be extended to represent more complex, arbitrary shapes. It shows a similar network architecture, but now used to approximate a more complex shape (illustrated by the pentagon and irregular shape). The key point is that an arbitrary number of circles can be "fit" into the shape. More accurate approximation is achieved with a greater number of smaller circles. This approach can achieve **arbitrary precision** in representing complex shapes.
This all was to prove that MLP is a universal classifier and even a 1 hidden layer MLP can model any classification boundary, not exactly but to arbitrary precision

Conclusion

The perceptron network of a single layer of infinitely many neurons can fit any decision surface, but a deep network requires fewer neurons to achieve its goal.

3.11 Why Depth Matters

In the below image we can see that we only needed 12 neurons in the first case and ∞ neurons in the second case.

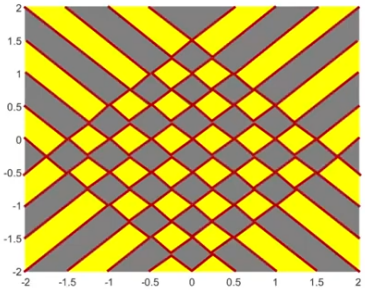


3.11.1 Optimal Depth

To classify the checkerboard pattern below, we want the output to be 1 inside the yellow regions and 0 outside. Using the circle technique we would end up using ∞ neurons if we want to do with 1 hidden layer.
We can do this with 2 hidden layers as well $\rightarrow$ 56 hidden neurons. The board has 16 lines. But since the checkerboard is essentially an XOR Gate we can solve for that technique as well.

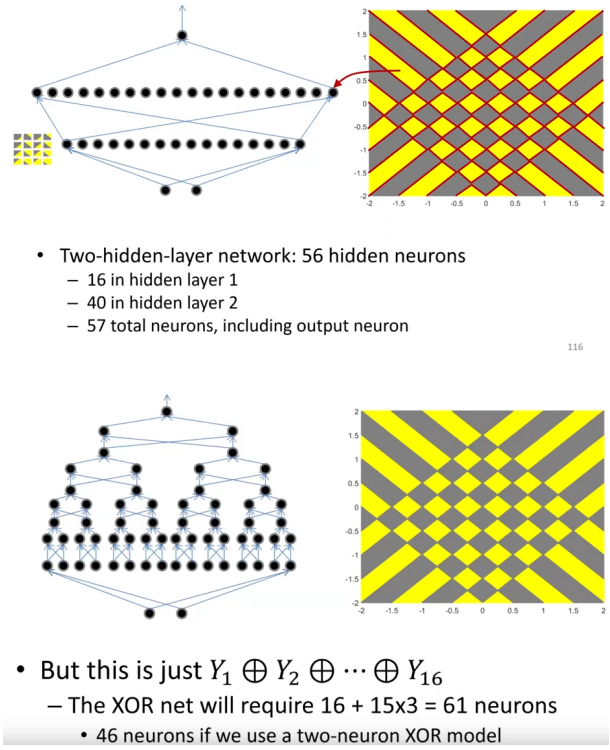
- we will need total 61 neurons if we use 3 neurons per xor
- total 46 neurons if 2 neurons per xor

(each XOR requires 3 neurons) so we could do this with 61 neurons.



3.12 MLP for Regression

A 3-unit MLP can generate a square pulse for continuous value functions. First neuron: Outputs 1 when input exceeds a threshold T_1, otherwise 0. Output is 0 before T_1, becomes 1 between T_1 and T_2, and reverts to 0 after T_2.Arbitrary functions can be built by slicing the domain into "windows" where a 2-unit network creates a pulse, scaled to the average value in that range. Increasing precision is possible by narrowing the windows (boxes). If the neuron's output is sinusoidal, no approximation is needed for functions that are inherently sinusoidal.



3.12.1 Higher Dimensional Functions

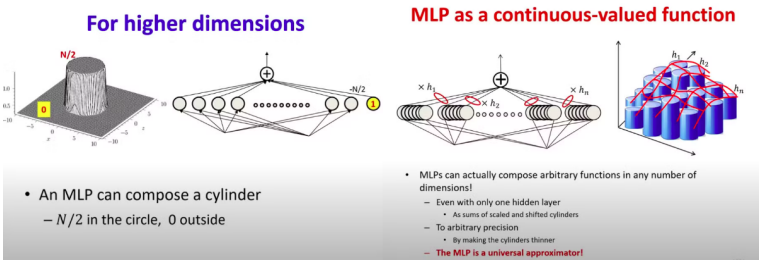
Higher dimensions use cylinders instead of pulses. Cylinders are 0 outside a specific region and $N/2$ (or normalized to 1) inside. The MLP fills the space under the continuous function with these cylinders. MLPs with only one hidden layer can compose any function in higher dimensions by summing scaled and shifted cylinders. Precision can be increased by making the cylinders thinner. So in summary:

- Even a 1-hidden layer MLP is capable of approximating any function, but the number of neurons required may be infinite.
- Deeper networks reduce the number of neurons needed for a given approximation error.

Depth: Summary

The number of neurons required in the shallow network is potentiolly exponentiol in the dimensionlity of the input.

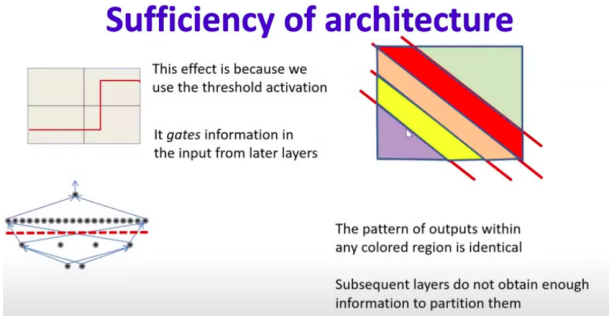
- Even a network with a single layer is a universal classifier
- But it may require exponentiolly large number of perceptrons than a deeper one
- Deeper networks require far fewer neurons than shallower ones to express the same function. They are exponentiolly smaller and are more expressive.



3.13 Issue with Depth

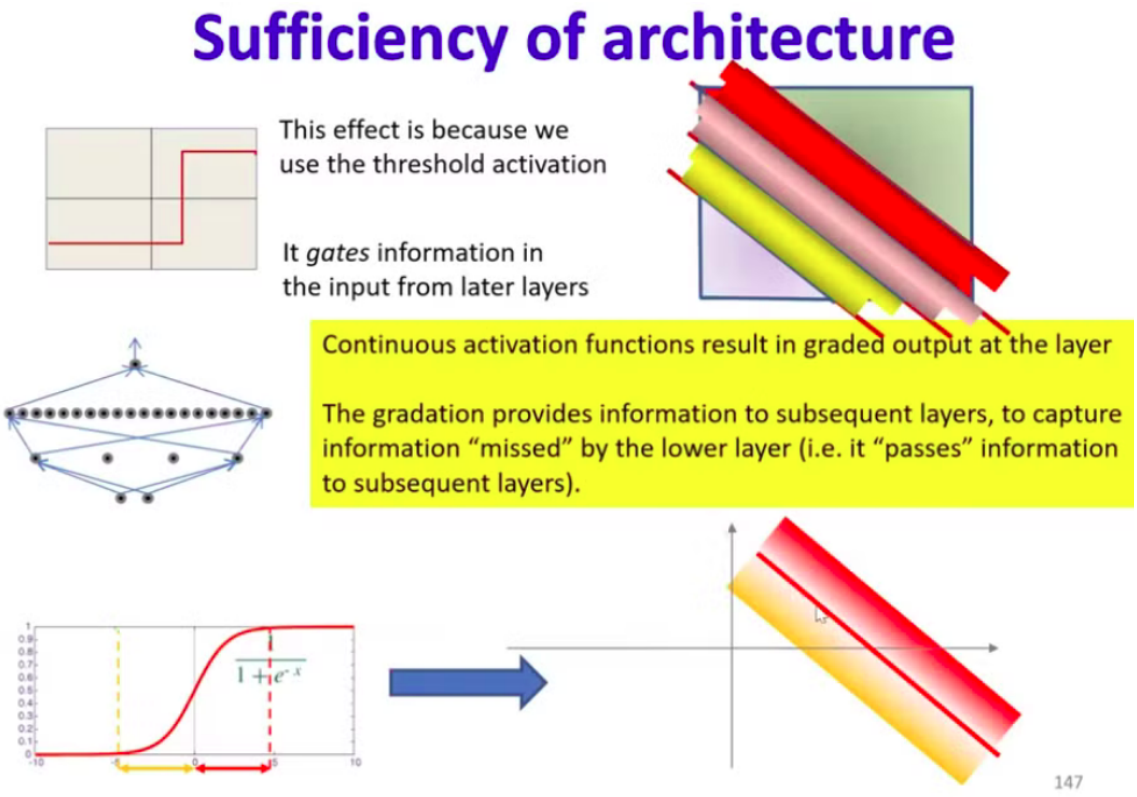
Every layer must be wide enough so that subsequent layers can construct the required pattern. Each neuron captures a line, and the next layer only has information about those lines which were identified by the previous layer. `Theshold activation gates information`

- Since sigmoid is not a step function, the value continues to change as we go away from the boundary. It tells you how far you are from the boundary
- The outputs of the neurons will continue to have some information that subsequent layers could use to learn the function properly
- But sigmoid quickly saturates, tells you for only a little while
- But RELU gives accurate distance from the boundary forever - more informative activation



You can think of the threshold activation and the architecture under the threshold activation as a boundary condition. As you decrease the width of the layers (from whatever the limitations the

threshold activation functions may have) your activation functions must continue to give information about the distance from the decision boundary. So if the activation function is sufficiently informative you can make the layers narrower and hope that subsequent layers in the network will capture the missing information But this also has limitations.



Summary

Deeper neurons can achieve the same information with far fewer neurons but must still have sufficient capacity. But activations must pass the information through. And each layer must be sufficiently wide to convey all relevant information to subsequent layers.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx