Neural Networks:
What can a network represent

MLSP, Fall 2016
Projects

• Teams of two
  – Unless you’re sufficiently macho to go it alone
    • I don’t recommend it

• Format:
  – Each team selects its own project
  – Give me a written proposal
    • Problem statement
    • General approach (vague is ok)
  – Final presentation:
    • A poster and, if possible a demo +
    • a written report

• Worth 40% of your marks, so this is important
  – Grading:
    • 50% on presentation (crowd-sourced)
    • 50% on report
Project suggestions

- HBKU wants to have a Pong championship
Project suggestions

- Quantum tic-tac-toe
Project suggestions

- Image painting
Project suggestions

- Image painting
  - Images from MIT’s nightmare machine
Project suggestion: Image recognition

- Deep convolutional neural networks for image analysis
  - Compare Resnets to Conventional convolutional networks
Project suggestion: Object detection

- Tougher than image recognition
Project Suggestion

\[
W(\text{"woman"}) - W(\text{"man"}) \approx W(\text{"aunt"}) - W(\text{"uncle"}) \\
W(\text{"woman"}) - W(\text{"man"}) \approx W(\text{"queen"}) - W(\text{"king"})
\]


• Modelling language: word embeddings
And others

- Will be put up on the website.
- Commit to project by end of Jan if possible
Recap: Neural networks have taken over AI

- Tasks that are made possible by NNs, aka deep learning
Recap: NNets and the brain

- In their basic form, NNets mimic the networked structure in the brain
Recap: The brain

- The Brain is composed of networks of neurons
Recap: Nnets and the brain

- Neural nets are composed of networks of computational models of neurons called perceptrons
• Onward and upward....
Remember the perceptron

- A threshold unit
  - “Fires” if the weighted sum of inputs exceeds a threshold

\[
y = \begin{cases} 
1 & \text{if } \sum_i w_i x_i \geq T \\
0 & \text{else}
\end{cases}
\]
The “soft” perceptron

- A “squashing” function instead of a threshold at the output
  - The **sigmoid** “activation” replaces the threshold

**Activation**: The function that acts on the weighted combination of inputs (and threshold)

\[
y = \frac{1}{1 + \exp(\sum_i w_i x_i - T)}
\]
Other “activations”

- Does not always have to be a squashing function
  - We will learn more about activations later
Remember the multi-layer perceptron

- A network of perceptrons
  - Generally “layered”
What is deep learning: human perspective

- **Learning: The human perspective:**
  - Acquisition of knowledge through experience
    - Underlying causes/influences/patterns
      - for data/phenomena
    - Not the same as memory

- **What is deep learning**
  - Comprehending the *inner structure* of observed data
  - *Cross-linking* new and known concepts to make non-obvious inferences
  - As opposed to *surface* learning..
    - Learning about the immediately observed data..
What is Learning: computational perspective

• The computational perspective:

• Acquisition of knowledge through experience
  – Exposure to data

• What is deep learning
  – Learning multi-level representations from data
  – Learning layered models of inputs.
Deep Structures

• In any directed network of computational elements with input source nodes and output sink nodes, “depth” is the length of the longest path from a source to a sink

Left: Depth = 2. Right: Depth = 3
Deep Structures

- **Layered** deep structure

- “Deep” → Depth > 2
MLPs approximate functions

- MLPs can compose Boolean functions
- MLPs can compose real-valued functions
- What are the limitations?
The MLP as a Boolean function

- How well do MLPs model Boolean functions?
The perceptron as a Boolean gate

• A perceptron can model any simple binary Boolean gate
Perceptron as a Boolean gate

- The universal AND gate
  - AND any number of inputs
    - Any subset of inputs may be negated

\[ (\bigwedge_{i=1}^{L} X_i) \land (\bigwedge_{i=L+1}^{N} \overline{X}_i) \]

Will fire only if \(X_1 \ldots X_L\) are all 1 and \(X_{L+1} \ldots X_N\) are all 0
**Perceptron as a Boolean gate**

- **The universal OR gate**
  - OR any number of inputs
  - Any subset of who may be negated

\[ X_1 \lor X_2 \lor \cdots \lor X_{L-N+1} \lor \overline{X_{L+1}} \lor \cdots \lor \overline{X_N} \]

Will fire only if any of  \(X_1 \) .. \(X_L\) are 1
or any of \(X_{L+1} \) .. \(X_N\) are 0
Perceptron as a Boolean Gate

• Universal OR:
  – Fire if any K-subset of inputs is “ON”

Will fire only if the total number of of $X_1 \ldots X_L$ that are 1 or $X_{L+1} \ldots X_N$ that are 0 is at least K
The perceptron is not enough

- Cannot compute an XOR

\[ X \oplus Y \]
Multi-layer perceptron

• MLPs can compute the XOR
Multi-layer perceptron

- MLPs can compute more complex Boolean functions
- MLPs can compute any Boolean function
  - Since they can emulate individual gates
- MLPs are universal Boolean functions
MLP as Boolean Functions

• MLPs are universal Boolean functions
  – Any function over any number of inputs and any number of outputs
• But how many “layers” will they need?
How many layers for a Boolean MLP?

- Expressed in disjunctive normal form

Truth Table

Truth table shows all input combinations for which output is 1

<table>
<thead>
<tr>
<th>X₁</th>
<th>X₂</th>
<th>X₃</th>
<th>X₄</th>
<th>X₅</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
How many layers for a Boolean MLP?

- Expressed in disjunctive normal form

Truth Table

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth table shows all input combinations for which output is 1

$$Y = \bar{X}_1 \bar{X}_2 X_3 \bar{X}_4 X_5 + \bar{X}_1 X_2 \bar{X}_3 X_4 X_5 + \bar{X}_1 X_2 X_3 \bar{X}_4 X_5 + X_1 \bar{X}_2 \bar{X}_3 \bar{X}_4 X_5 + X_1 \bar{X}_2 X_3 X_4 X_5 + X_1 X_2 \bar{X}_3 X_4 X_5 + X_1 X_2 \bar{X}_3 \bar{X}_4 X_5$$
How many layers for a Boolean MLP?

<table>
<thead>
<tr>
<th>X_1</th>
<th>X_2</th>
<th>X_3</th>
<th>X_4</th>
<th>X_5</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth table shows all input combinations for which output is 1

\[
Y = X_1 \bar{X}_2 X_3 X_4 \bar{X}_5 + \bar{X}_1 X_2 \bar{X}_3 X_4 X_5 + \bar{X}_1 X_2 X_3 \bar{X}_4 \bar{X}_5 + X_1 X_2 X_3 X_4 X_5 + X_1 \bar{X}_2 X_3 X_4 X_5 + X_1 X_2 \bar{X}_3 \bar{X}_4 X_5
\]

- Expressed in disjunctive normal form
How many layers for a Boolean MLP?

Truth Table

<table>
<thead>
<tr>
<th>X₁</th>
<th>X₂</th>
<th>X₃</th>
<th>X₄</th>
<th>X₅</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth table shows all input combinations for which output is 1

\[
Y = \bar{X}_1 \bar{X}_2 X_3 X_4 \bar{X}_5 + X_1 X_2 \bar{X}_3 X_4 X_5 + \bar{X}_1 X_2 X_3 \bar{X}_4 \bar{X}_5 \\
+ X_1 \bar{X}_2 \bar{X}_3 \bar{X}_4 X_5 + X_1 X_2 X_3 X_4 X_5 + X_1 X_2 \bar{X}_3 \bar{X}_4 X_5
\]

• Expressed in disjunctive normal form
How many layers for a Boolean MLP?

Truth Table

<table>
<thead>
<tr>
<th>X_1</th>
<th>X_2</th>
<th>X_3</th>
<th>X_4</th>
<th>X_5</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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

\[
Y = \overline{X_1} \overline{X_2} X_3 X_4 \overline{X_5} + \overline{X_1} X_2 \overline{X_3} X_4 X_5 + \overline{X_1} X_2 \overline{X_3} X_4 X_5 + X_1 \overline{X_2} X_3 \overline{X_4} X_5 + X_1 \overline{X_2} X_3 X_4 X_5 + X_1 X_2 X_3 X_4 X_5
\]

- Expressed in disjunctive normal form
How many layers for a Boolean MLP?

Truth Table

<table>
<thead>
<tr>
<th>X₁</th>
<th>X₂</th>
<th>X₃</th>
<th>X₄</th>
<th>X₅</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth table shows all input combinations for which output is 1

\[ Y = \bar{X}_1 \bar{X}_2 \bar{X}_3 X_4 \bar{X}_5 + \bar{X}_1 X_2 \bar{X}_3 \bar{X}_4 X_5 + \bar{X}_1 X_2 X_3 \bar{X}_4 \bar{X}_5 + \\
X_1 \bar{X}_2 \bar{X}_3 \bar{X}_4 X_5 + X_1 \bar{X}_2 X_3 X_4 X_5 + X_1 X_2 \bar{X}_3 \bar{X}_4 X_5 \]

• Expressed in disjunctive normal form
How many layers for a Boolean MLP?

Truth Table

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth table shows all input combinations for which output is 1

$$Y = \overline{X}_1 \overline{X}_2 X_3 X_4 \overline{X}_5 + \overline{X}_1 X_2 \overline{X}_3 X_4 X_5 + \overline{X}_1 X_2 X_3 \overline{X}_4 \overline{X}_5 + X_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 X_5 + \overline{X}_1 \overline{X}_2 X_3 X_4 X_5 + X_1 X_2 \overline{X}_3 \overline{X}_4 X_5$$

- Expressed in disjunctive normal form
How many layers for a Boolean MLP?

- Expressed in disjunctive normal form

Truth Table

<table>
<thead>
<tr>
<th>X₁</th>
<th>X₂</th>
<th>X₃</th>
<th>X₄</th>
<th>X₅</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth table shows all input combinations for which output is 1

\[
Y = \bar{X}_1 \bar{X}_2 X_3 X_4 \bar{X}_5 + \bar{X}_1 X_2 \bar{X}_3 X_4 X_5 + \bar{X}_1 X_2 X_3 \bar{X}_4 X_5 + X_1 X_2 \bar{X}_3 \bar{X}_4 X_5 + X_1 X_2 X_3 X_4 X_5 + X_1 X_2 \bar{X}_3 X_4 X_5
\]
How many layers for a Boolean MLP?

Truth Table

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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

$$Y = \overline{X}_1 \overline{X}_2 X_3 X_4 \overline{X}_5 + \overline{X}_1 X_2 \overline{X}_3 X_4 X_5 + \overline{X}_1 X_2 X_3 \overline{X}_4 \overline{X}_5 + X_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 X_5 + X_1 \overline{X}_2 X_3 X_4 X_5 + X_1 X_2 \overline{X}_3 \overline{X}_4 X_5$$

• Expressed in disjunctive normal form
How many layers for a Boolean MLP?

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

Truth table shows all input combinations for which output is 1

\[
Y = \bar{X}_1 \bar{X}_2 X_3 X_4 \bar{X}_5 + \bar{X}_1 X_2 \bar{X}_3 X_4 X_5 + \bar{X}_1 X_2 X_3 \bar{X}_4 \bar{X}_5 \\
+ X_1 \bar{X}_2 \bar{X}_3 \bar{X}_4 X_5 + X_1 \bar{X}_2 X_3 X_4 X_5 + X_1 X_2 \bar{X}_3 \bar{X}_4 X_5
\]

But what is the largest number of perceptrons required in the single-hidden layer for an N-input variable 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.

• DNF form:
  – Find groups
  – Express as reduced DNF
Reducing a Boolean Function

Basic DNF formula will require 7 terms
Reducing a Boolean Function

- Reduced DNF form:
  - Find groups
  - Express as reduced DNF

\[ O = \overline{Y} \overline{Z} + \overline{W} X \overline{Y} + \overline{X} Y \overline{Z} \]
Reducing a Boolean Function

Reduced DNF form:

- Find groups
- Express as reduced DNF

\[ O = \overline{Y} \overline{Z} + \overline{W} X \overline{Y} + \overline{X} Y \overline{Z} \]
• What arrangement of ones and zeros simply cannot be reduced further?
Largest irreducible DNF?

- What arrangement of ones and zeros simply cannot be reduced further?
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?
• How many neurons in a DNF (one-hidden-layer) MLP for this Boolean function of 6 variables?
How many neurons in a DNF (one-hidden-layer) MLP for this Boolean function

Can be generalized: Will require $2^{N-1}+1$ perceptrons in hidden layer

Exponential in $N$

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

Can be generalized: Will require $2^{N-1} + 1$ perceptrons in hidden layer

Exponential in $N$

How many units if we use multiple layers?

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

\begin{align*}
O &= W \oplus X \oplus Y \oplus Z \\
O &= U \oplus V \oplus W \oplus X \oplus Y \oplus Z
\end{align*}
An XOR takes three perceptrons.
Width of a deep MLP

\[ O = W \oplus X \oplus Y \oplus Z \]

- An XOR needs 3 perceptrons
- This network will require \(3 \times 3 = 9\) perceptrons
An XOR needs 3 perceptrons

This network will require $3 \times 5 = 15$ perceptrons
An XOR needs 3 perceptrons

This network will require $3 \times 5 = 15$ perceptrons

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

More generally, the XOR of $N$ variables will require $3(N-1)$ perceptrons!!
Width of a single-layer Boolean MLP

Single hidden layer: Will require $2^{N-1}+1$ perceptrons in hidden layer

Exponential in $N$

Will require $3(N-1)$ perceptrons in a deep network

Linear in $N$!!!
The need for depth

- Deep Boolean MLPs that scale *linearly* with the number of inputs ...
- ... can become exponentially large if recast using only one layer
- It gets worse..
The need for depth

- The wide function can happen at any layer
- Having a few extra layers can greatly reduce network size

\[ a \oplus b \oplus c \oplus d \oplus e \oplus f \]
Story so far

- Multi-layer perceptrons are *Universal Boolean Machines*

- Even a network with a *single* hidden layer is a universal Boolean machine
  - But a single-layer network may require an exponentially large number of perceptrons

- Deeper networks may require far fewer neurons than shallower networks to express the same function
  - Could be *exponentially* smaller
The MLP as a classifier

• MLP as a function over real inputs
• MLP as a function that finds a complex “decision boundary” over a space of *reals*
A Perceptron on Reals

- A perceptron operates on real-valued vectors
- This is a linear classifier
Boolean functions with a real perceptron

• Boolean perceptrons are also linear classifiers
  – Purple regions are 1
Composing complicated “decision” boundaries

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

• The network must fire if the input is in the coloured area
Booleans over the reals

- The network must fire if the input is in the coloured area
Booleans over the reals

• The network must fire if the input is in the coloured area
Booleans over the reals

- The network must fire if the input is in the coloured area
Booleans over the reals

- The network must fire if the input is in the coloured area
Booleans over the reals

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

\[ \sum_{i=1}^{N} y_i \geq 5? \]
More complex decision boundaries

• Network to fire if the input is in the yellow area
  – “OR” two polygons
  – A third layer is required
Complex decision boundaries

• Can compose *arbitrarily* complex decision boundaries
Complex decision boundaries

- Can compose *arbitrarily* complex decision boundaries
Complex decision boundaries

• Can compose arbitrarily complex decision boundaries
  – With only one hidden layer!
  – How?
Exercise: compose this with one hidden layer

- How would you compose the decision boundary to the left with only one hidden layer?
Composing a Square decision boundary

- The polygon net

\[ \sum_{i=1}^{4} y_i \geq 4? \]
Composing a pentagon

- The polygon net

\[
\sum_{i=1}^{5} y_i \geq 5?
\]
Composing a hexagon

- The polygon net

\[ \sum_{i=1}^{N} y_i \geq 6? \]
How about a heptagon

• What are the sums in the different regions?
  – A pattern emerges as we consider $N > 6$. ..
Composing a polygon

- The polygon net
- Increasing the number of sides reduces the area outside the polygon that have $N/2 < \text{Sum} < N$
Composing a circle

- The circle net
  - Very large number of neurons
  - Sum is $N$ inside the circle, $N/2$ outside everywhere
  - Circle can be of arbitrary diameter, at any location

\[ \sum_{i=1}^{N} y_i \geq N? \]
Adding circles

- The “sum” of two circles sub nets is exactly $N$ inside either circle, and $N/2$ outside.

$$\sum_{i=1}^{N} y_i \geq N?$$
Composing an arbitrary figure

- Just fit in an arbitrary number of circles
  - More accurate approximation with greater number of smaller circles
  - Can achieve arbitrary precision
MLP: Universal classifier

- MLPs can capture *any* classification boundary
- A *one-layer MLP* can model any classification boundary
- *MLPs are universal classifiers*
• Deeper networks can require far fewer neurons
Story so far

• Multi-layer perceptrons are *Universal Boolean Machines*
  – Even a network with a *single* hidden layer is a universal Boolean machine

• Multi-layer perceptrons are *Universal Classification Functions*
  – Even a network with a single hidden layer is a universal classifier

• But a single-layer network may require an exponentially large number of perceptrons than a deep one

• Deeper networks may require far fewer neurons than shallower networks to express the same function
  – Could be *exponentially* smaller
MLP as a 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 $T_1$ and $T_2$
  - $T_1$ and $T_2$ can be arbitrarily specified
MLP as a continuous-valued regression

- A simple 3-unit MLP can generate a “square pulse” over an input
- An MLP with many units can model an arbitrary function over an input
  - To arbitrary precision
    - Simply make the individual pulses narrower
- A one-layer MLP can model an arbitrary function of a single input
For higher-dimensional functions

- An MLP can compose a cylinder
  - $N$ in the circle, $N/2$ outside
A “true” cylinder

• An MLP can compose a true cylinder
  – N/2 in the circle, 0 outside
  – By adding a “bias”
  – We will encounter bias terms again
    • They are standard components of perceptrons
MLP as a continuous-valued function

• MLPs can actually compose arbitrary functions
  – Even with only one layer
    • As sums of scaled and shifted cylinders
  – To arbitrary precision
    • By making the cylinders thinner
  – The MLP is a universal approximator!
Caution: MLPs with additive output units are universal approximators

- MLPs can actually compose arbitrary functions
- But explanation so far only holds if the output unit only performs summation
  - i.e. does not have an additional “activation”
“Proper” networks: Outputs with activations

• Output neuron may have actual “activation”
  – Threshold, sigmoid, tanh, softplus, rectifier, etc.
• What is the property of such networks?
The network as a function

- Output unit with activation function
  - Threshold or Sigmoid, or any other
- The network is actually a map from the set of all possible input values to all possible output values
  - All values the activation function of the output neuron

\[ f : \{0,1\}^N \rightarrow \{0,1\} \quad \text{Boolean} \]

\[ f : R^N \rightarrow \{0,1\} \quad \text{Threshold} \]

\[ f : R^N \rightarrow (0,1) \quad \text{Sigmoid} \]

\[ f : R^N \rightarrow (-1,1) \quad \text{Tanh} \]

\[ f : R^N \rightarrow (0,\infty) \quad \text{Softmax, Rectifier} \]
The network as a function

- **Output unit with activation function**
  - Threshold or Sigmoid, or any other

- The network is actually a map from the set of all possible input values to all possible output values
  - All values the activation function of the output neuron

- **The MLP is a Universal Approximator for the entire class of functions (maps) it represents!**

- **Functions**
  - $f: \{0,1\}^N \rightarrow \{0,1\}$  
    - Boolean
  - $f: \mathbb{R}^N \rightarrow \{0,1\}$  
    - Threshold
  - $f: \mathbb{R}^N \rightarrow (0,1)$  
    - Sigmoid
  - $f: \mathbb{R}^N \rightarrow (-1,1)$  
    - Tanh
  - $f: \mathbb{R}^N \rightarrow (0,\infty)$  
    - Softmax, Rectifier
The issue of depth

• Previous discussion showed that a *single-layer* MLP is a universal function approximator
  – Can approximate any function to arbitrary precision
  – But may require infinite neurons in the layer

• More generally, deeper networks will require far fewer neurons for the same approximation error
  – The network is a generic map
    • The same principles that apply for Boolean networks apply here
  – Can be exponentially fewer than the 1-layer network
Lessons today

- MLPs are universal Boolean function
- MLPs are universal classifiers
- MLPs are universal function approximators

- A *single-layer* MLP can approximate anything to arbitrary precision
  - But could be exponentially or even infinitely wide in its inputs size
- Deeper MLPs can achieve the same precision with far fewer neurons
Next up

• We *know* MLPs can emulate any function
• But how do we *make* them emulate a specific desired function
  – E.g. a function that takes an image as input and outputs the labels of all objects in it
  – E.g. a function that takes speech input and outputs the labels of all phonemes in it
  – Etc...

• *Training an MLP*