

DS 4400

Machine Learning and Data Mining I

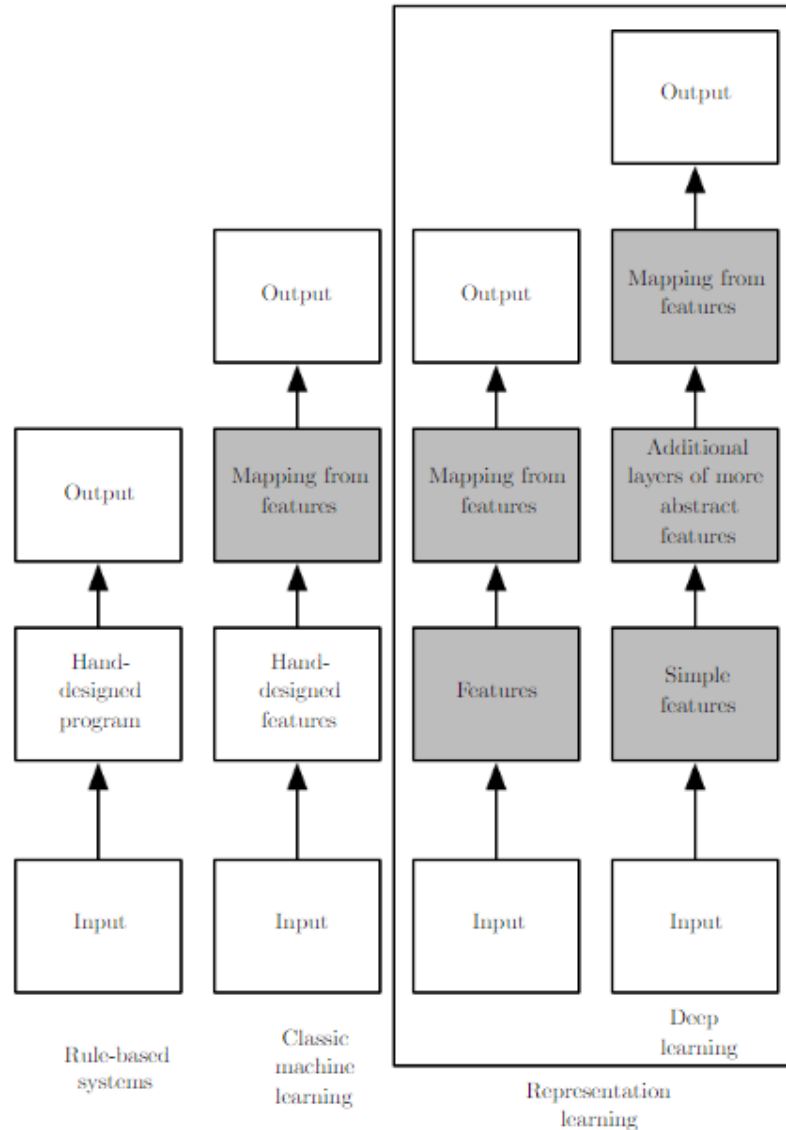
Alina Oprea
Associate Professor, CCIS
Northeastern University

March 14 2019

Outline

- Deep Learning
 - Success stories
 - Types of architectures
- Feed-Forward architectures
 - Terminology
 - Non-linear activations
 - Multi-Layer Perceptron
 - Multi-class classification (softmax unit)
 - Representing Boolean functions

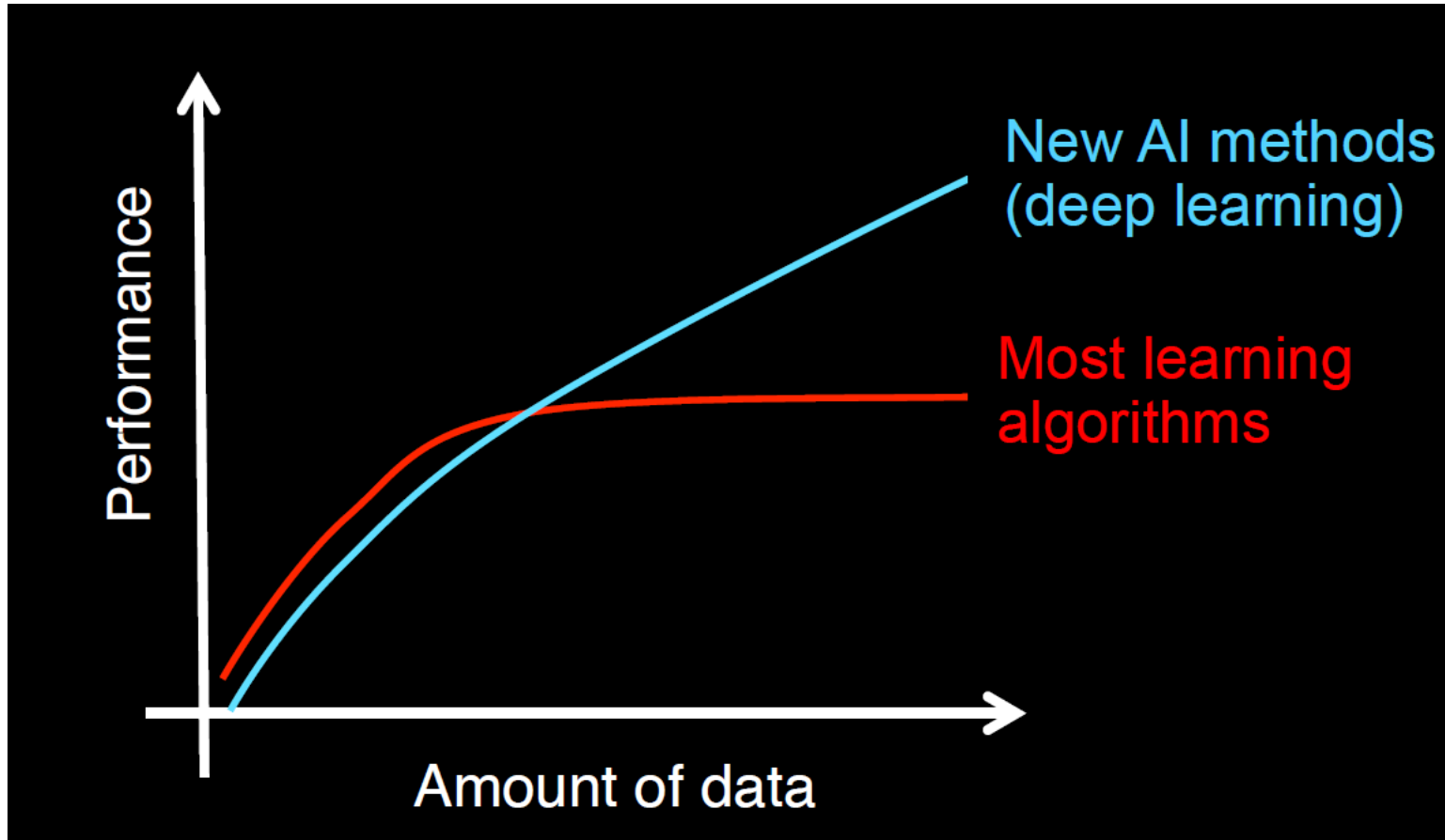
Deep Learning vs Traditional Learning



Neural Networks

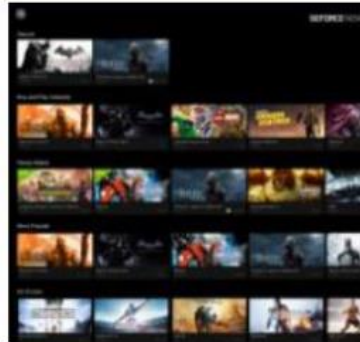
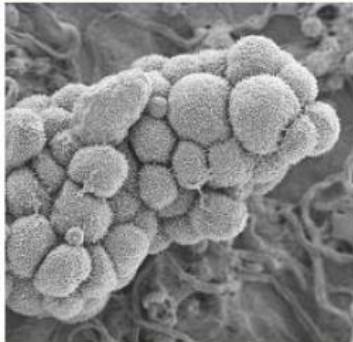
- Origins: Algorithms that try to mimic the brain.
- Very widely used in 80s and early 90s; popularity diminished in late 90s.
- Recent resurgence: State-of-the-art technique for many applications
- Artificial neural networks are not nearly as complex or intricate as the actual brain structure

Performance of Deep Learning



Deep Learning Applications

DEEP LEARNING EVERYWHERE



INTERNET & CLOUD

Image Classification
Speech Recognition
Language Translation
Language Processing
Sentiment Analysis
Recommendation

MEDICINE & BIOLOGY

Cancer Cell Detection
Diabetic Grading
Drug Discovery

MEDIA & ENTERTAINMENT

Video Captioning
Video Search
Real Time Translation

SECURITY & DEFENSE

Face Detection
Video Surveillance
Satellite Imagery

AUTONOMOUS MACHINES

Pedestrian Detection
Lane Tracking
Recognize Traffic Sign

Success stories: Speech recognition

www.technewsworld.com/story/84013.html

40 maps that explain Amazon Web Services Primers | Math n Pro: deeplearning.net/tut- Deep Learning Tutor- deep learning PHILIPS - Golden Ears Language Technology: MyIDCare - Dashbo: Other bookmarks

TECHNEWSWORLD EMERGING TECH SEARCH

Computing Internet IT Mobile Tech Reviews Security Technology Tech Blog Reader Services

Microsoft AI Beats Humans at Speech Recognition

By Richard Adhikari
Oct 20, 2016 11:40 AM PT

Print Email




Image: Adobe Stock

Microsoft's Artificial Intelligence and Research Unit earlier this week reported that its speech recognition technology had surpassed the performance of human transcriptionists.

G+ 5
Tweet 25
Share 45
Share 11
Share 0
share 104

Facebook Twitter LinkedIn Google+ SoundCloud RSS

Most Popular Newsletters News Alerts

How do you feel about Black Friday and Cyber Monday?

- They're great -- I get a lot of bargains!
- The deals are too spread out -- I'd prefer just one day.
- They're a fun way to kick off the holiday season.
- I don't like the commercialization of Thanksgiving Day.
- They're crucial for the retail industry and the economy.
- The deals typically aren't that good.

Vote to See Results

E-Commerce Times

Black Friday Shoppers Hungry for New Experiences, New Tech

Pay TV's Newest Innovation: Giving Users Control

Apple Celebrates Itself in \$300 Coffee Table Tome

AWS Enjoys Top Perch in IaaS, PaaS Markets

US Comptroller Gears Up for Blockchain and

Success stories: Machine Translation



The image shows a screenshot of a web browser displaying a Google blog post. The browser's address bar shows the URL: <https://blog.google/products/translate/round-translation-more-accurate-fluent-sentences-google-translate/>. The browser's bookmark bar contains several items, including '40 maps that explain', 'Amazon Web Services', 'Primers | Math n Pro', 'deeplearning.net/tut...', 'Deep Learning Tutor...', 'deep learning', 'PHILIPS - Golden Ears', 'Language Technolog...', 'MyIDCare - Dashbo...', and 'Other bookmarks'. The page header features the Google logo, navigation links for 'The Keyword', 'Latest Stories', 'Product News', and 'Topics', and a search icon. The main content area has a yellow background and contains the following text:

TRANSLATE NOV 15, 2016

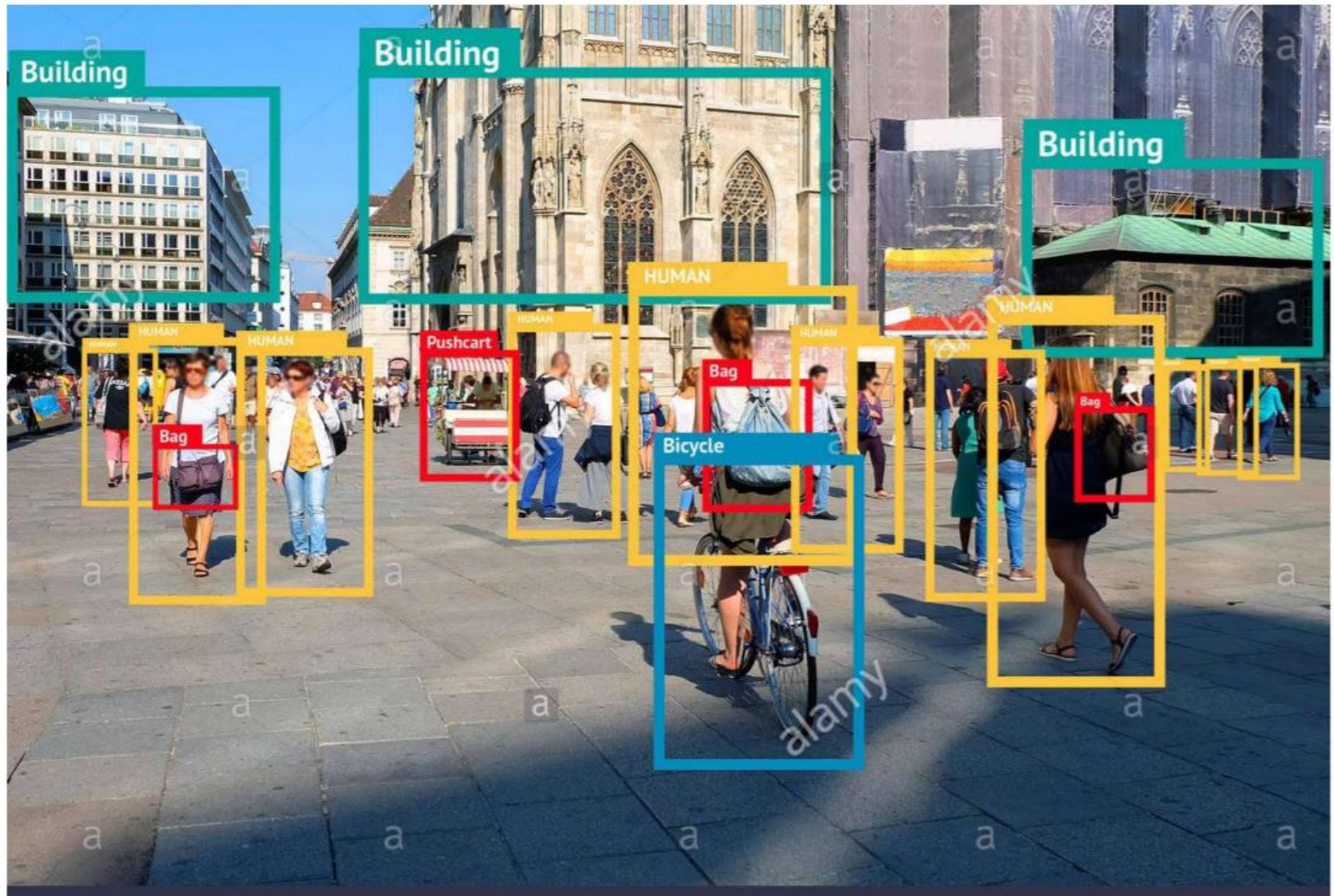
Found in translation: More accurate, fluent sentences in Google Translate

Barak Turovsky
PRODUCT LEAD, GOOGLE TRANSLATE

In 10 years, Google Translate has gone from supporting just a few languages to 103, connecting strangers, reaching across language barriers and even helping

A blue circular share icon is visible in the bottom right corner of the page.









Success stories: Image segmentation



Success stories: Image captioning

cs.stanford.edu/people/karpathy/deepimagesent/

40 maps that explain Amazon Web Services Primers | Math n Pro deeplearning.net/tut Deep Learning Tutor deep learning PHILIPS - Golden Ears Language Technolog MyIDCare - Dashbo Other bookmarks

			
"man in black shirt is playing guitar."	"construction worker in orange safety vest is working on road."	"two young girls are playing with lego toy."	"boy is doing backflip on wakeboard."
			
"girl in pink dress is jumping in air."	"black and white dog jumps over bar."	"young girl in pink shirt is swinging on swing."	"man in blue wetsuit is surfing on wave."

References

- Deep Learning book
 - <https://www.deeplearningbook.org/>
- Stanford notes on deep learning
 - http://cs229.stanford.edu/notes/cs229-notes-deep_learning.pdf
- History of Deep Learning
 - https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

Example

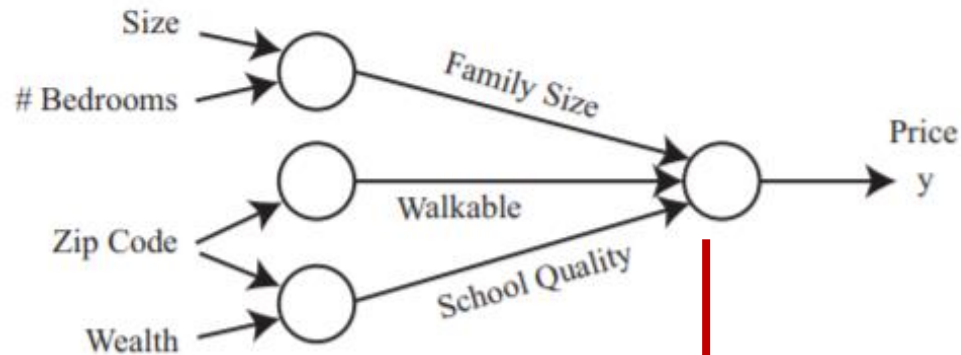
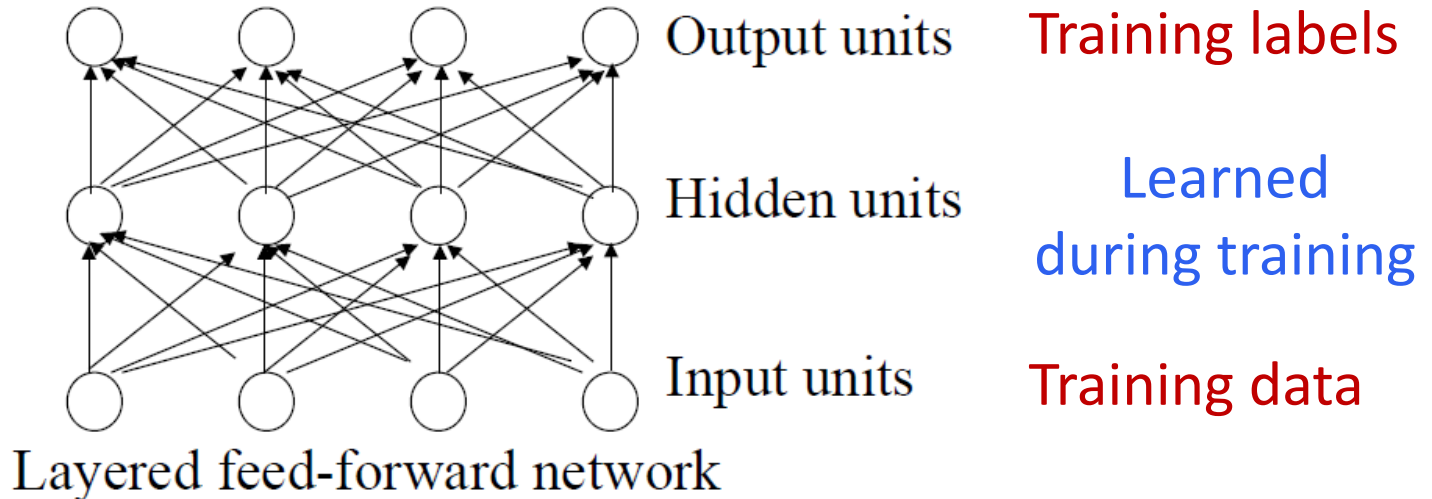


Figure 2: Diagram of a small neural network for predicting housing prices.

Intermediate Features

- Provide as input only training data: input and label
- Neural Networks automatically learn intermediate features!

Neural Networks



- Neural networks are made up of **nodes** or **units**, connected by **links**
- Each link has an associated **weight** and **activation level**
- Each node has an **input function** (typically summing over weighted inputs), an **activation function**, and an **output**

Recall: The Perceptron

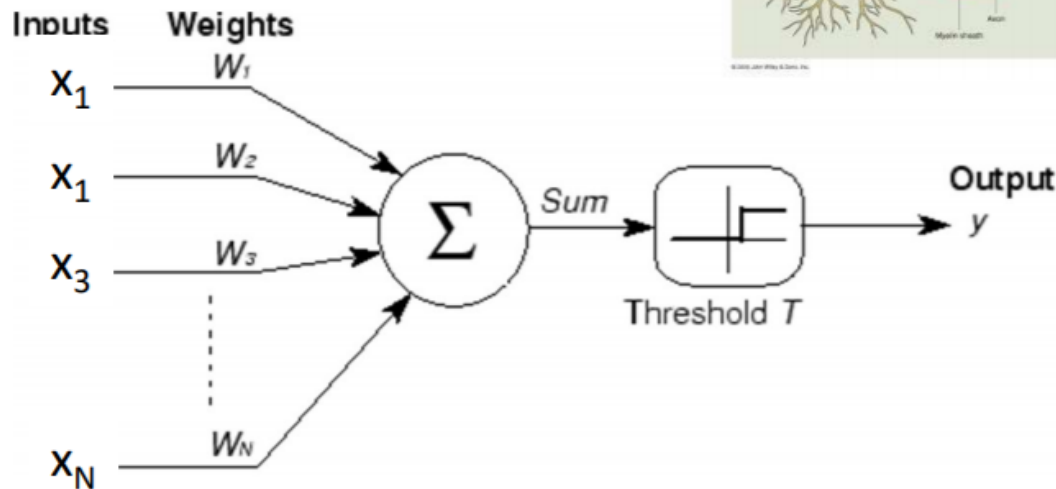
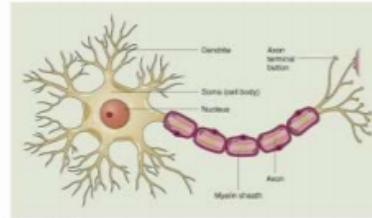
$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^\top \mathbf{x}) \quad \text{where} \quad \text{sign}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases}$$

- The perceptron uses the following update rule each time it receives a new training instance $(\mathbf{x}^{(i)}, y^{(i)})$

$$\theta_j \leftarrow \theta_j - \frac{1}{2} \underbrace{\left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)} \right)}_{\text{either 2 or -2}} x_j^{(i)}$$

- If the prediction matches the label, make no change
- Otherwise, adjust $\boldsymbol{\theta}$

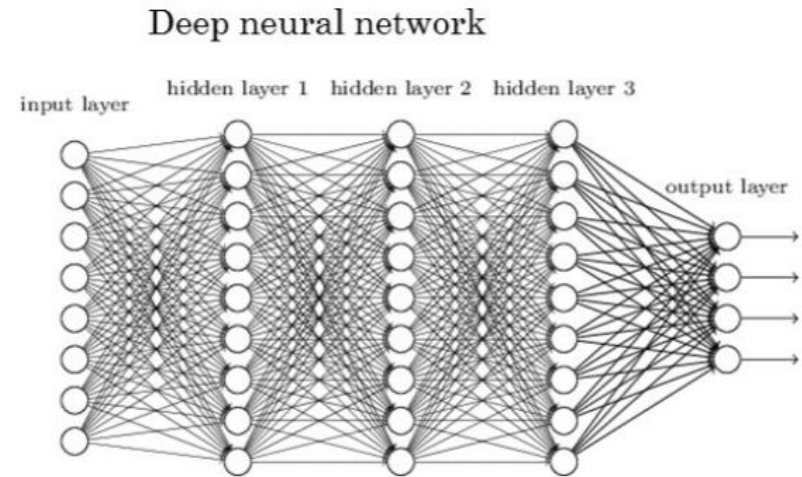
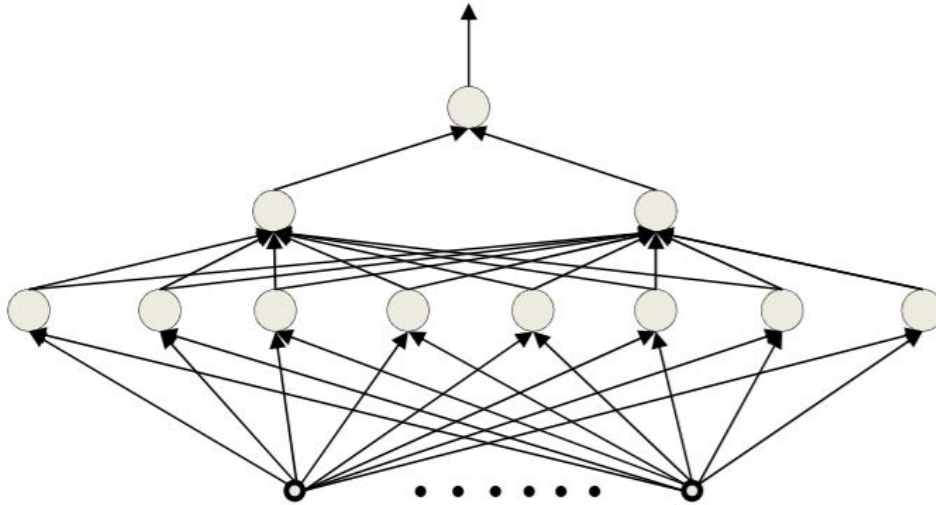
Perceptron



$$y = \begin{cases} 1 & \text{if } \sum_i w_i x_i \geq T \\ 0 & \text{else} \end{cases}$$

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

Multi-Layer Perceptron



- A network of perceptrons
 - Generally “layered”

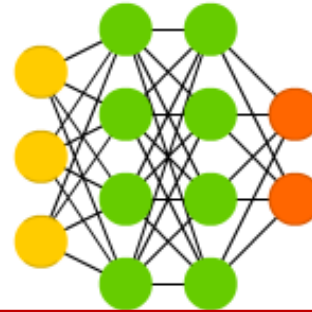


Neural Network Architectures

Feed-Forward Networks

- Neurons from each layer connect to neurons from next layer

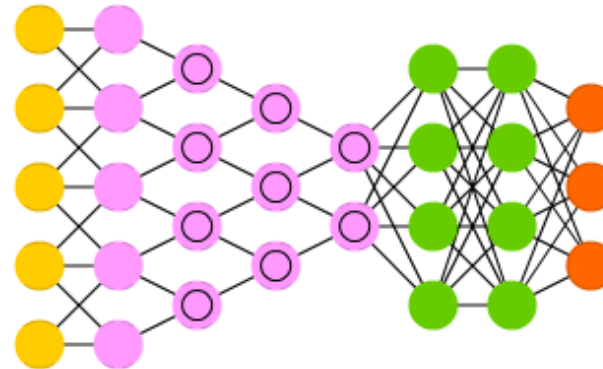
Deep Feed Forward (DFF)



Convolutional Networks

- Includes convolution layer for feature reduction
- Learns hierarchical representations

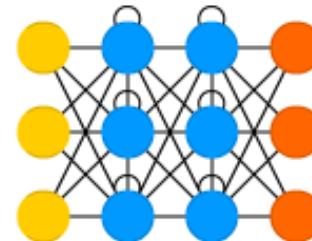
Deep Convolutional Network (DCN)



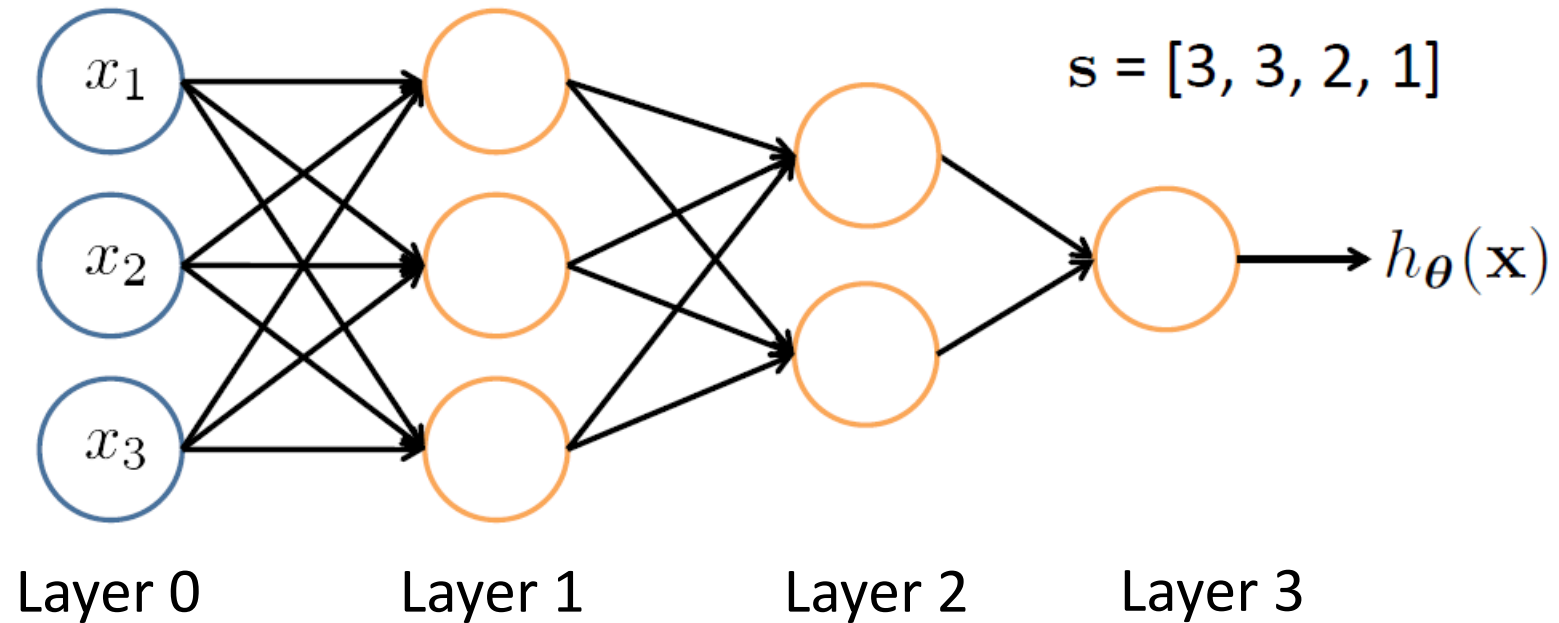
Recurrent Networks

- Keep hidden state
- Have cycles in computational graph

Recurrent Neural Network (RNN)



Feed-Forward Networks



L denotes the number of layers

$\mathbf{s} \in \mathbb{N}^{+L}$ contains the numbers of nodes at each layer

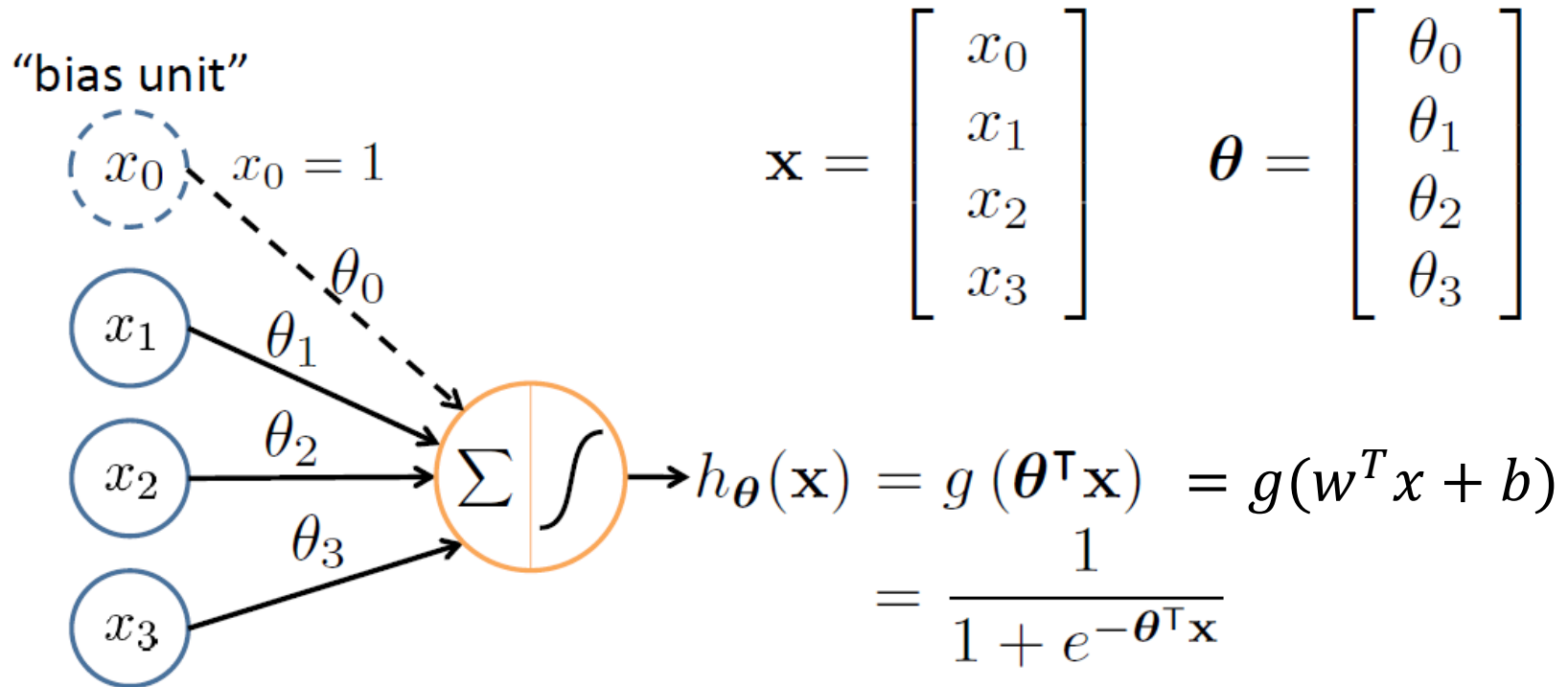
- Not counting bias units

- Typically, $s_0 = d$ (# input features) and $s_{L-1} = K$ (# classes)

Feed-Forward NN

- Hyper-parameters
 - Number of layers
 - Architecture (how layers are connected)
 - Number of hidden units per layer
 - Number of units in output layer
 - Activation functions
- Other
 - Initialization
 - Regularization

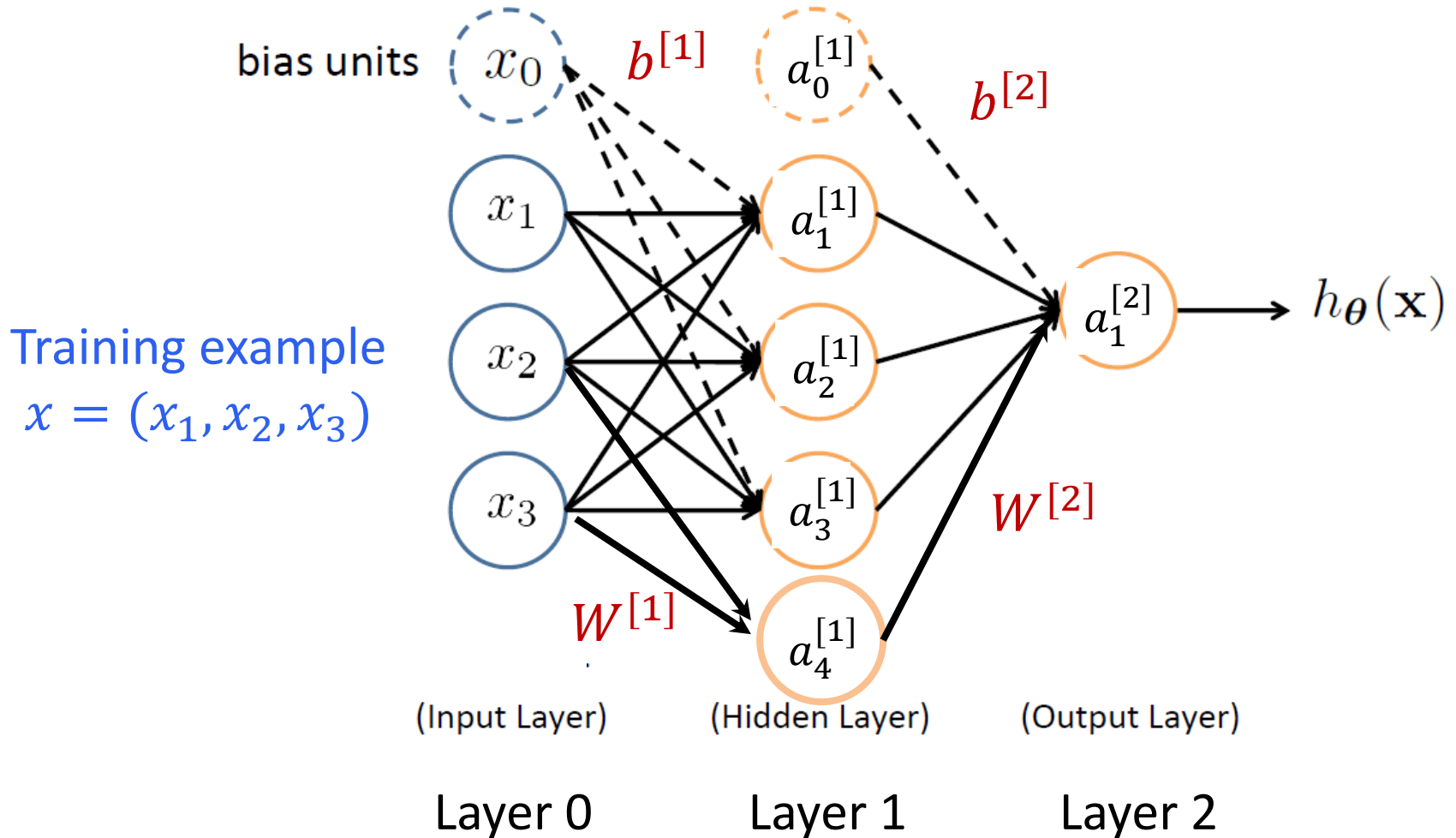
Logistic Unit: A simple NN



Sigmoid (logistic) activation function: $g(z) = \frac{1}{1 + e^{-z}}$

No hidden layers

Feed-Forward Neural Network



No cycles

$$\theta = (b^{[1]}, W^{[1]}, b^{[2]}, W^{[2]})$$

Vectorization

$$z_1^{[1]} = W_1^{[1]T} x + b_1^{[1]} \quad \text{and} \quad a_1^{[1]} = g(z_1^{[1]})$$

$$\vdots \qquad \qquad \qquad \vdots \qquad \qquad \qquad \vdots$$

$$z_4^{[1]} = W_4^{[1]T} x + b_4^{[1]} \quad \text{and} \quad a_4^{[1]} = g(z_4^{[1]})$$

$$\underbrace{\begin{bmatrix} z_1^{[1]} \\ \vdots \\ \vdots \\ z_4^{[1]} \end{bmatrix}}_{z^{[1]} \in \mathbb{R}^{4 \times 1}} = \underbrace{\begin{bmatrix} - & W_1^{[1]T} & - \\ - & W_2^{[1]T} & - \\ & \vdots & \\ - & W_4^{[1]T} & - \end{bmatrix}}_{W^{[1]} \in \mathbb{R}^{4 \times 3}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{x \in \mathbb{R}^{3 \times 1}} + \underbrace{\begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ \vdots \\ b_4^{[1]} \end{bmatrix}}_{b^{[1]} \in \mathbb{R}^{4 \times 1}}$$

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

$$a^{[1]} = g(z^{[1]})$$

Vectorization

Output layer

$$z_1^{[2]} = W_1^{[2]T} a^{[1]} + b_1^{[2]} \quad \text{and} \quad a_1^{[2]} = g(z_1^{[2]})$$

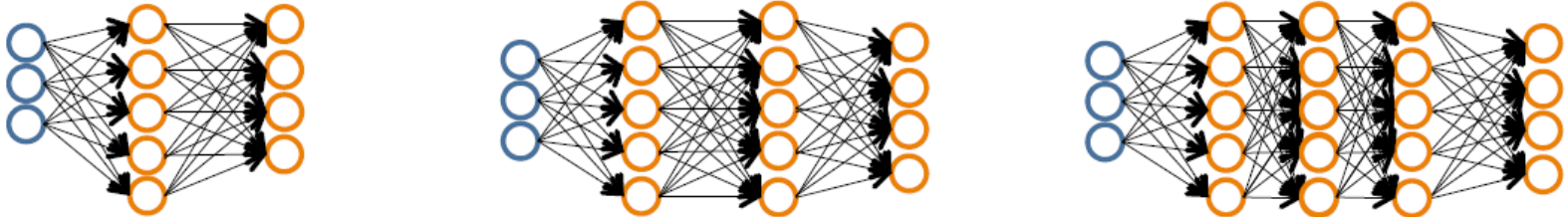
$$\underbrace{z^{[2]}}_{1 \times 1} = \underbrace{W^{[2]}}_{1 \times 4} \underbrace{a^{[1]}}_{4 \times 1} + \underbrace{b^{[2]}}_{1 \times 1} \quad \text{and} \quad \underbrace{a^{[2]}}_{1 \times 1} = g(\underbrace{z^{[2]}}_{1 \times 1})$$

Hidden Units

- Layer 1
 - First hidden unit:
 - Linear: $z_1^{[1]} = W_1^{[1]T} x + b_1^{[1]}$
 - Non-linear: $a_1^{[1]} = g(z_1^{[1]})$
 - ...
 - Fourth hidden unit:
 - Linear: $z_4^{[1]} = W_4^{[1]T} x + b_4^{[1]}$
 - Non-linear: $a_4^{[1]} = g(z_4^{[1]})$
- Terminology
 - $a_i^{[j]}$ - **Activation** of unit i in layer j
 - g - **Activation** function
 - $W^{[j]}$ - **Weight vector** controlling mapping from layer j-1 to j
 - $b^{[j]}$ - **Bias vector** from layer j-1 to j

How to pick architecture?

Pick a network architecture (connectivity pattern between nodes)



- # input units = # of features in dataset
- # output units = # classes

Reasonable default: 1 hidden layer

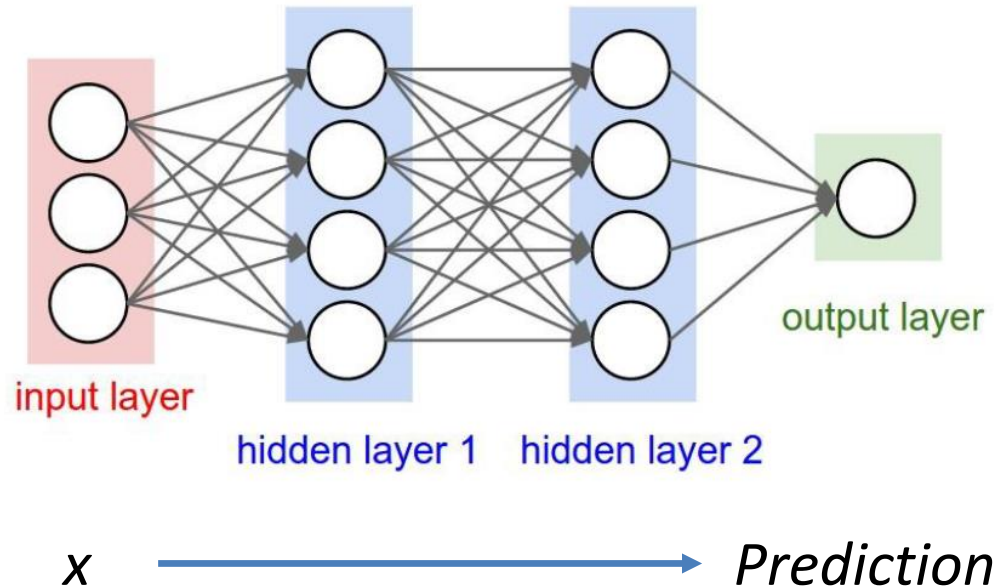
- or if >1 hidden layer, have same # hidden units in every layer (usually the more the better)

Training Neural Networks

- Input training dataset D
 - Number of features: d
 - Labels from K classes
- First layer has $d+1$ units (one per feature and bias)
- Output layer has K units
- Training procedure determines parameters that optimize loss function
 - Backpropagation
 - Learn optimal $W^{[i]}, b^{[i]}$ at layer i
- Testing done by forward propagation

Forward Propagation

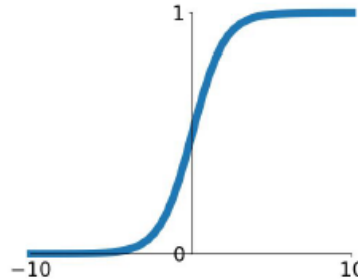
- The input neurons first receive the data features of the object. After processing the data, they send their output to the first hidden layer.
- The hidden layer processes this output and sends the results to the next hidden layer.
- This continues until the data reaches the final output layer, where the output value determines the object's classification.
- This entire process is known as **Forward Propagation**, or **Forward prop.**



Activation Functions

Sigmoid

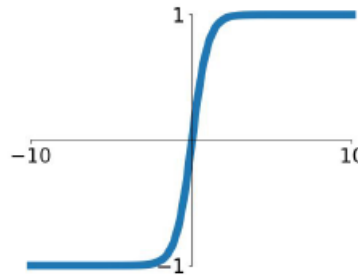
$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Binary
Classification

tanh

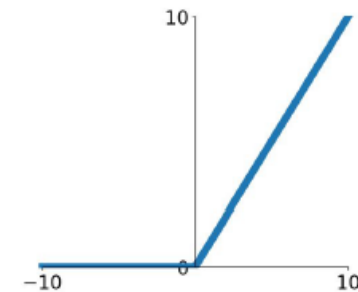
$$\tanh(x)$$



Regression

ReLU

$$\max(0, x)$$



Intermediary
layers

Why Non-Linear Activations?

- Assume g is linear: $g(z) = Uz$
 - At layer 1: $z^{[1]} = W^{[1]}x + b^{[1]}$
 - $a^{[1]} = g(z^{[1]}) = Uz^{[1]} = UW^{[1]}x + Ub^{[1]}$
- Layer 2:
 - $a^{[2]} = g(z^{[2]}) = Uz^{[2]} = UW^{[2]}a^{[1]} + Ub^{[2]} =$
 $= UW^{[2]}UW^{[1]}x + UW^{[2]}Ub^{[1]} + Ub^{[2]}$
- Last layer
 - Output is linear in input!
 - Then NN will only learn linear functions

Acknowledgements

- Slides made using resources from:
 - Yann LeCun
 - Andrew Ng
 - Eric Eaton
 - David Sontag
 - Andrew Moore
- Thanks!