

DS 4400

Machine Learning and Data Mining I  
Spring 2021

Alina Oprea

Associate Professor

Khoury College of Computer Science

Northeastern University

March 18 2021

# Outline

- Deep Learning
  - Motivation
  - Goals
- Deep Learning as representation learning
- Perceptron and its limitations
- Feed-forward neural networks

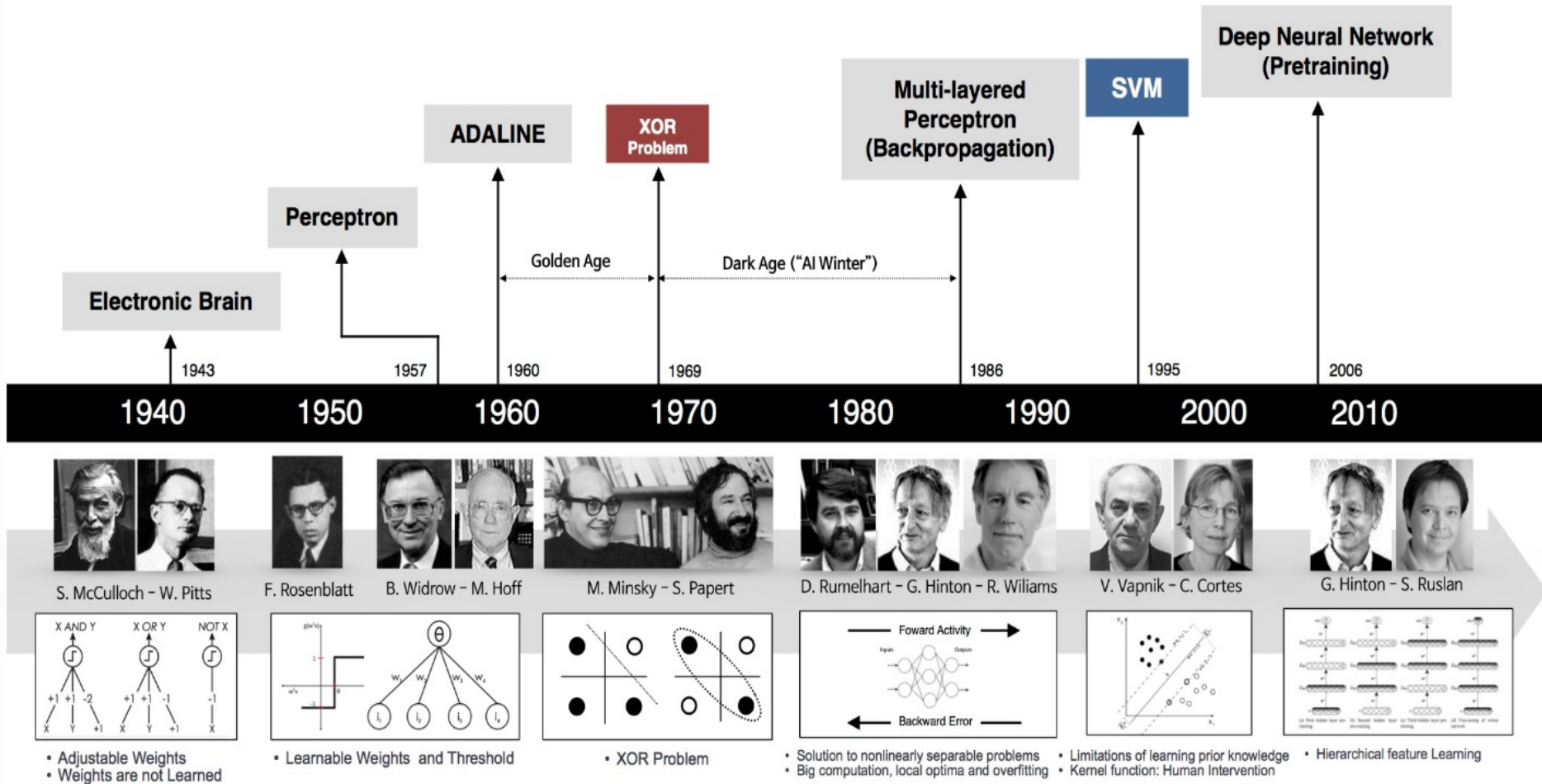
# SVM Classifier

- Support Vector Classifier (SVC, linear SVM)
  - SVC classifier is linear combination of dot product between testing point and support vectors
  - $h(z) = \theta_0 + \sum_{i \in S} \alpha_i \langle z, x_i \rangle$
  - Two methods to train:
    - Solve directly optimization problem to find maximum margin
    - Gradient descent with hinge loss objective
- SVM classifier
  - Select a kernel function  $K$
  - SVM classifier is linear combination of kernel between testing point and support vectors
  - $h(z) = \theta_0 + \sum_{i \in S} \alpha_i K(z, x_i)$

# Comparing SVM with other classifiers

- SVM is resilient to outliers
  - Similar to Logistic Regression
  - LDA or kNN are not
- SVM can be trained with Gradient Descent
  - Hinge loss cost function
- Supports regularization
  - Can add penalty term (ridge or Lasso) to cost function
- Linear SVM is most similar to Logistic Regression

# History of Deep Learning



# References

- Deep Learning books
  - <https://d2l.ai/> (D2L)
  - <https://www.deeplearningbook.org/> (advanced)
- Stanford notes on deep learning
  - [http://cs229.stanford.edu/summer2020/cs229-notes-deep\\_learning.pdf](http://cs229.stanford.edu/summer2020/cs229-notes-deep_learning.pdf)

# Before 2013

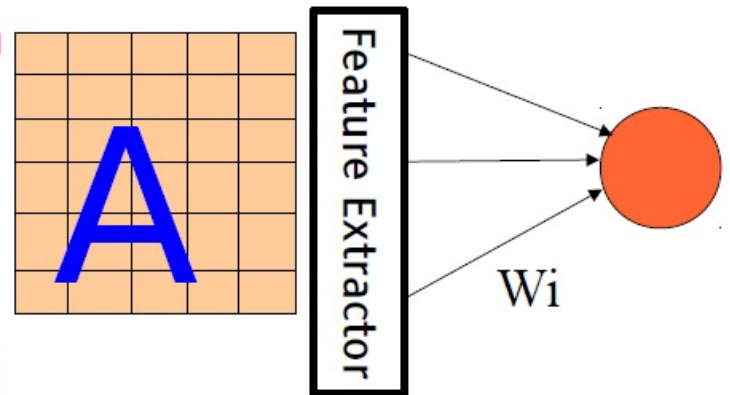
- The first learning machine: the **Perceptron**

- ▶ Built at Cornell in 1960

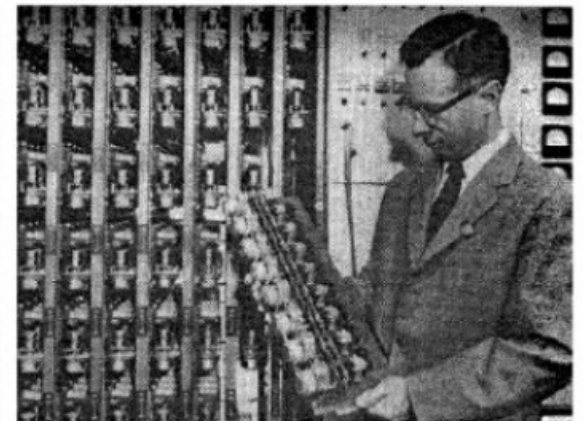
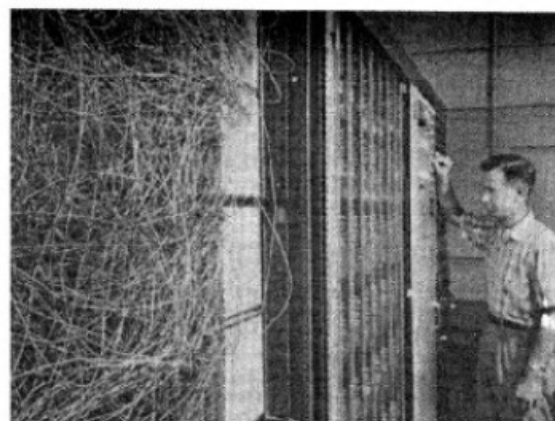
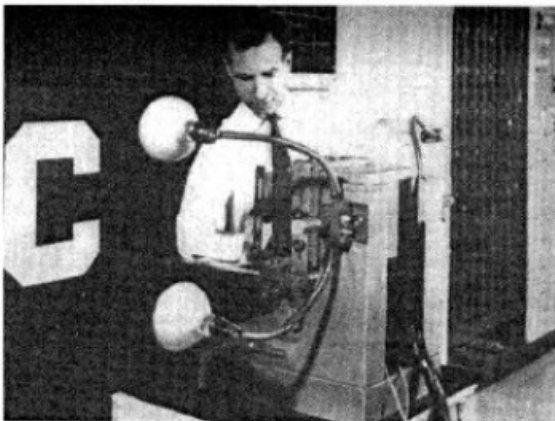
- The Perceptron was a **linear classifier** on top of a simple **feature extractor**

- The vast majority of practical applications of ML today use glorified **linear classifiers** or glorified template matching.

- Designing a feature extractor requires considerable efforts by experts.



$$y = \text{sign} \left( \sum_{i=1}^N W_i F_i(X) + b \right)$$



# Deep Learning

- The traditional model of pattern recognition (since the late 50's)
  - ▶ Fixed/engineered features (or fixed kernel) + trainable classifier



hand-crafted  
Feature Extractor

"Simple" Trainable  
Classifier

- End-to-end learning / Feature learning / Deep learning



Trainable  
Feature Extractor

Trainable  
Classifier



# Trainable Feature Hierarchy

- Hierarchy of representations with increasing level of abstraction

- Each stage is a kind of trainable feature transform

- Image recognition

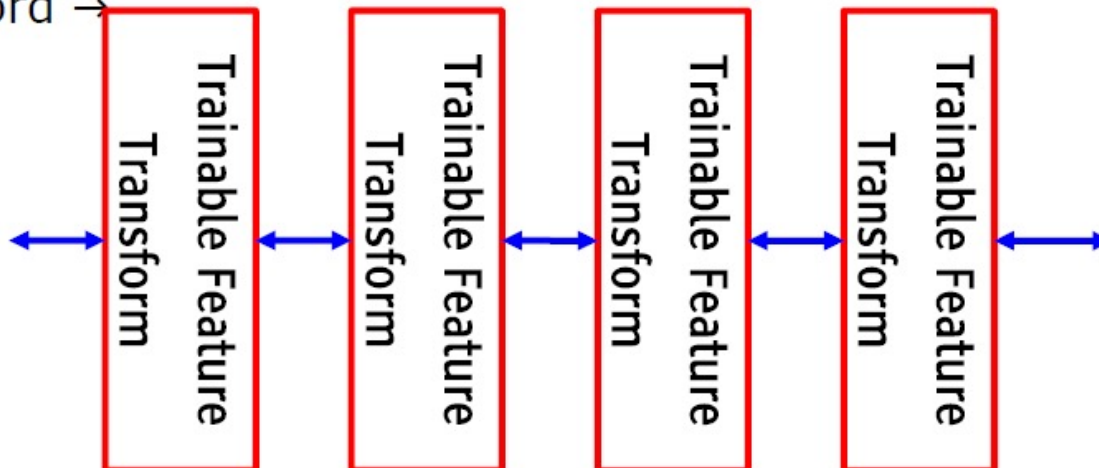
  - ▶ Pixel → edge → texon → motif → part → object

- Text

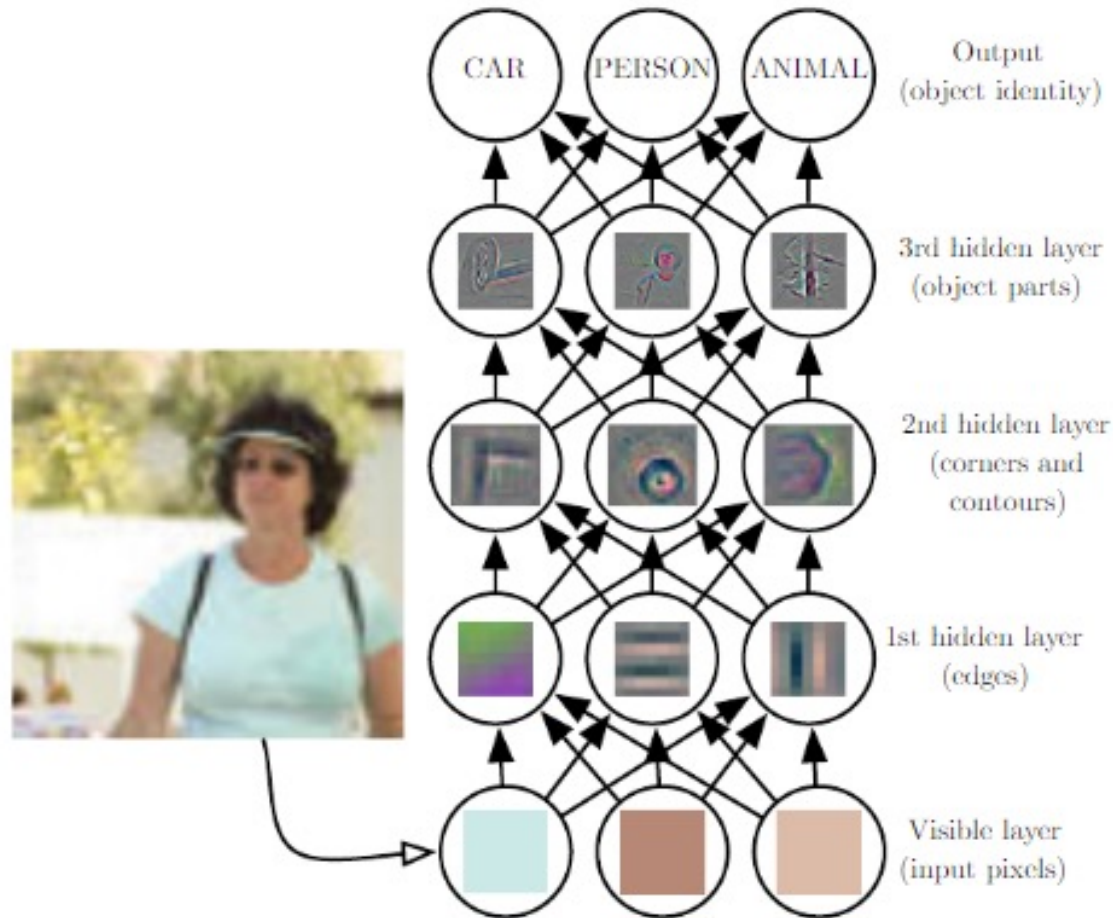
  - ▶ Character → word → word group → clause → sentence → story

- Speech

  - ▶ Sample → spectral band → sound → ... → phone → phoneme → word →

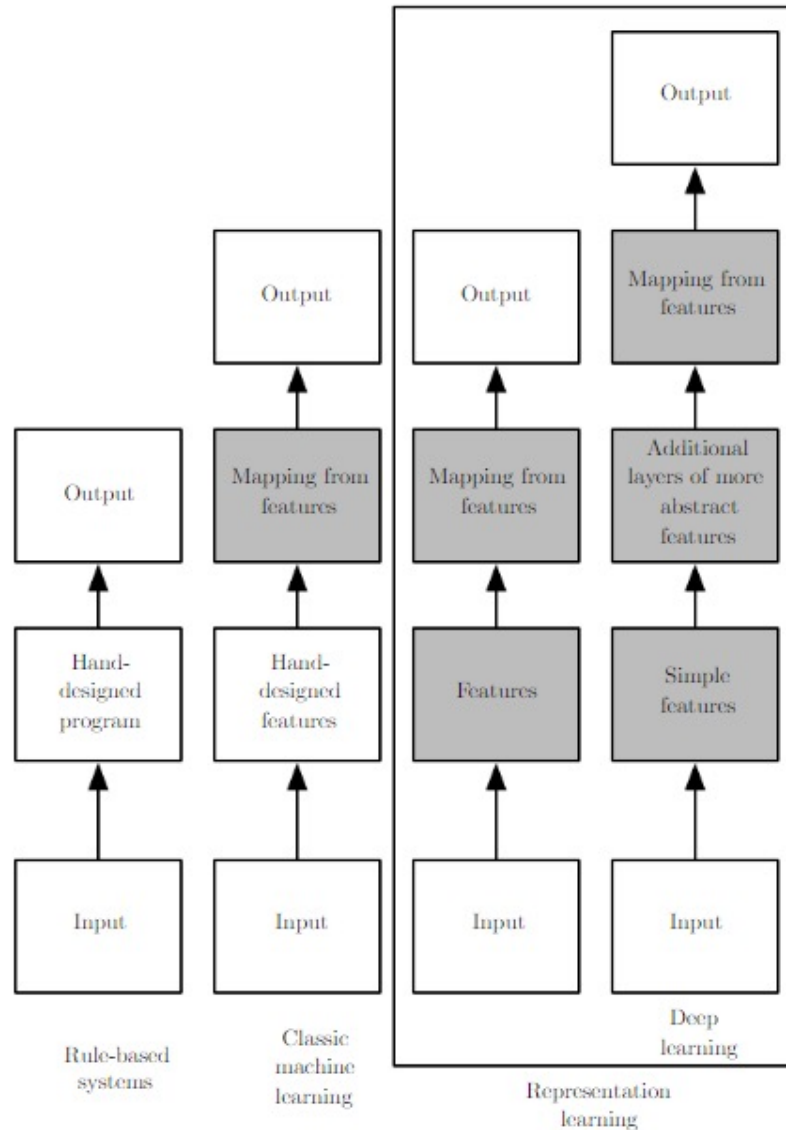


# Learning Representations



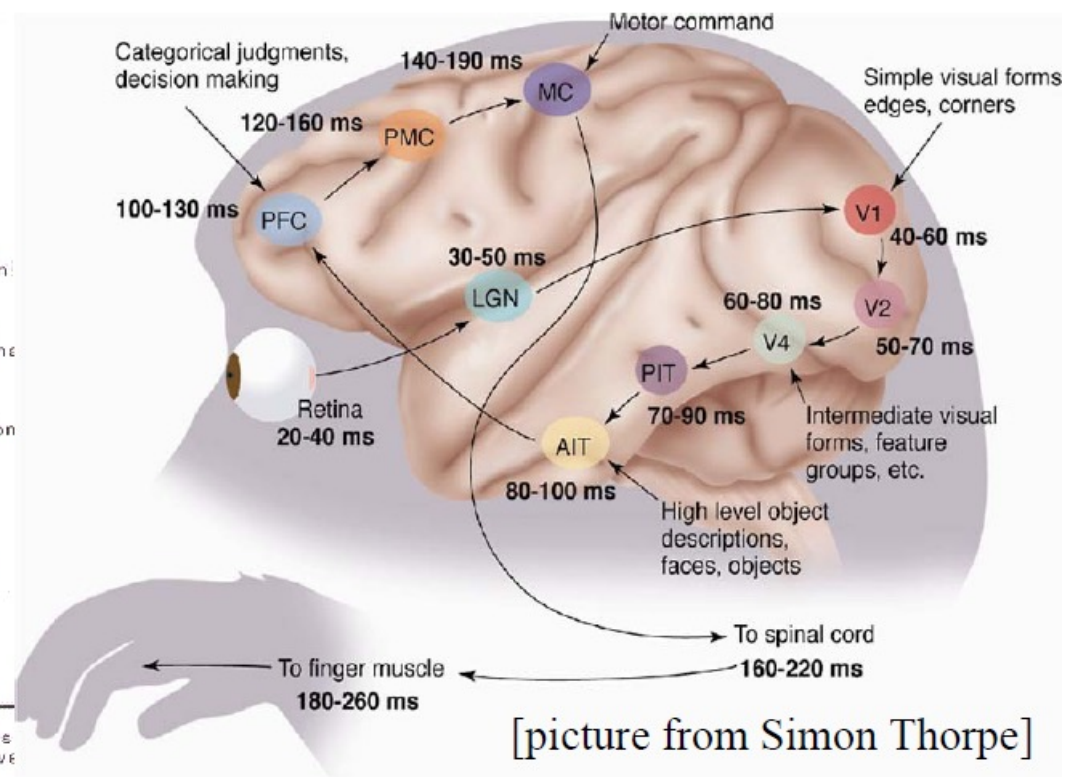
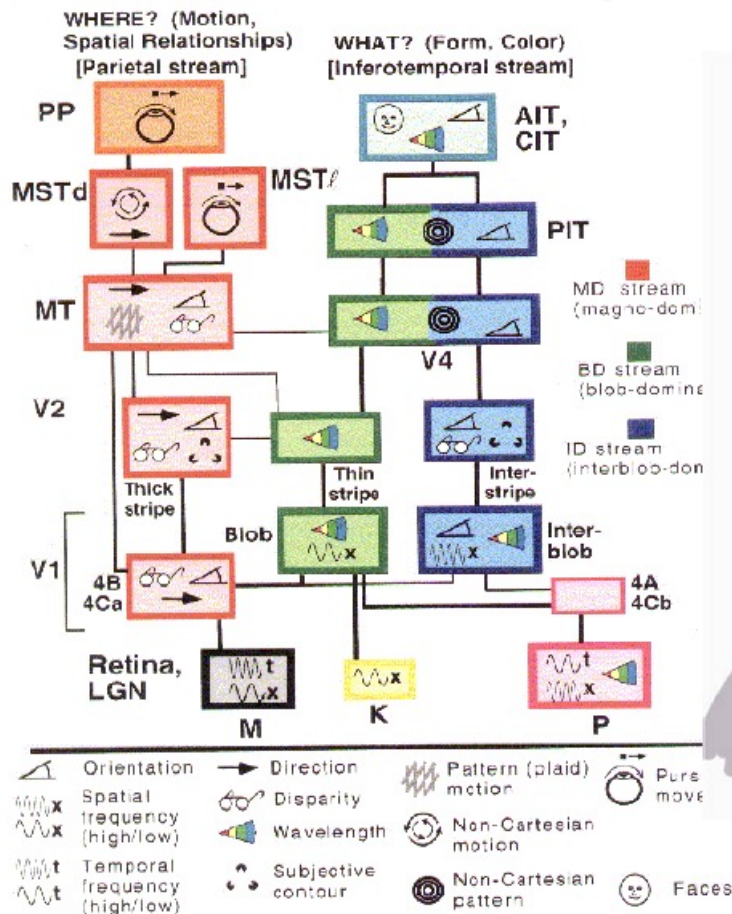
Deep Learning addresses the problem of learning hierarchical representations

# Deep Learning vs Traditional Learning



# The Visual Cortex is Hierarchical

- The ventral (recognition) pathway in the visual cortex has multiple stages
- Retina - LGN - V1 - V2 - V4 - PIT - AIT ....
- Lots of intermediate representations



[Gallant & Van Essen]

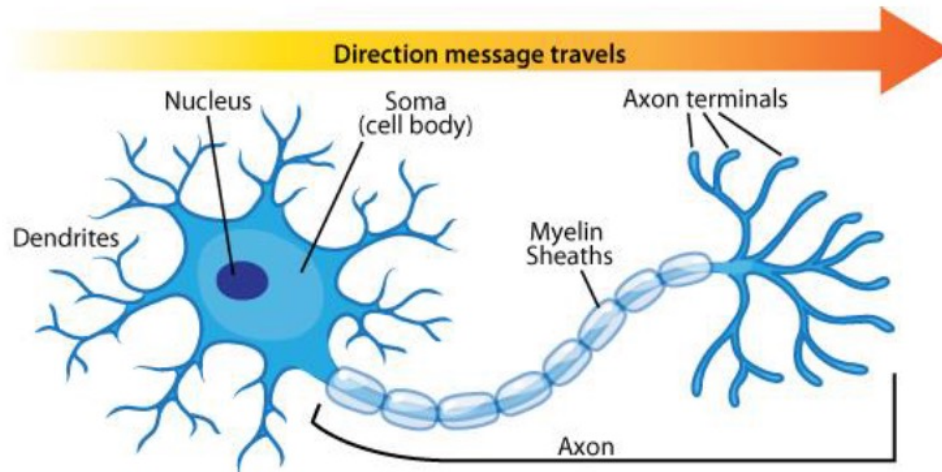


# Neural Function

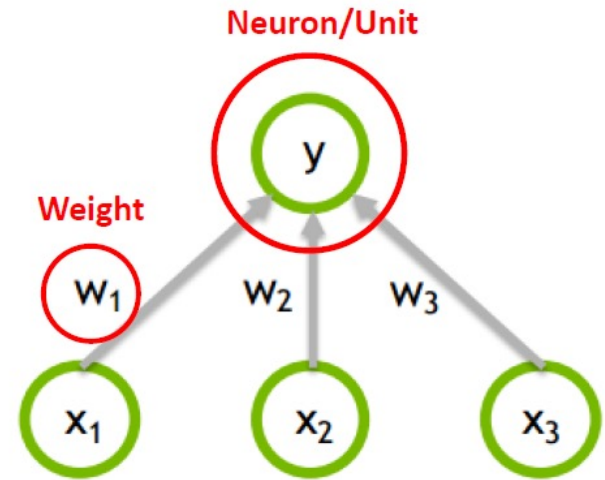
- Brain function (thought) occurs as the result of the firing of **neurons**
- Neurons connect to each other through **synapses**, which propagate **action potential** (electrical impulses) by releasing **neurotransmitters**
  - Synapses can be **excitatory** (potential-increasing) or **inhibitory** (potential-decreasing), and have varying **activation thresholds**
  - Learning occurs as a result of the synapses' **plasticity**: They exhibit long-term changes in connection strength
- There are about  $10^{11}$  neurons and about  $10^{14}$  synapses in the human brain!

# Analogy to Human Brain

## Human Brain



Biological Neuron



$$y = F(w_1x_1 + w_2x_2 + w_3x_3)$$

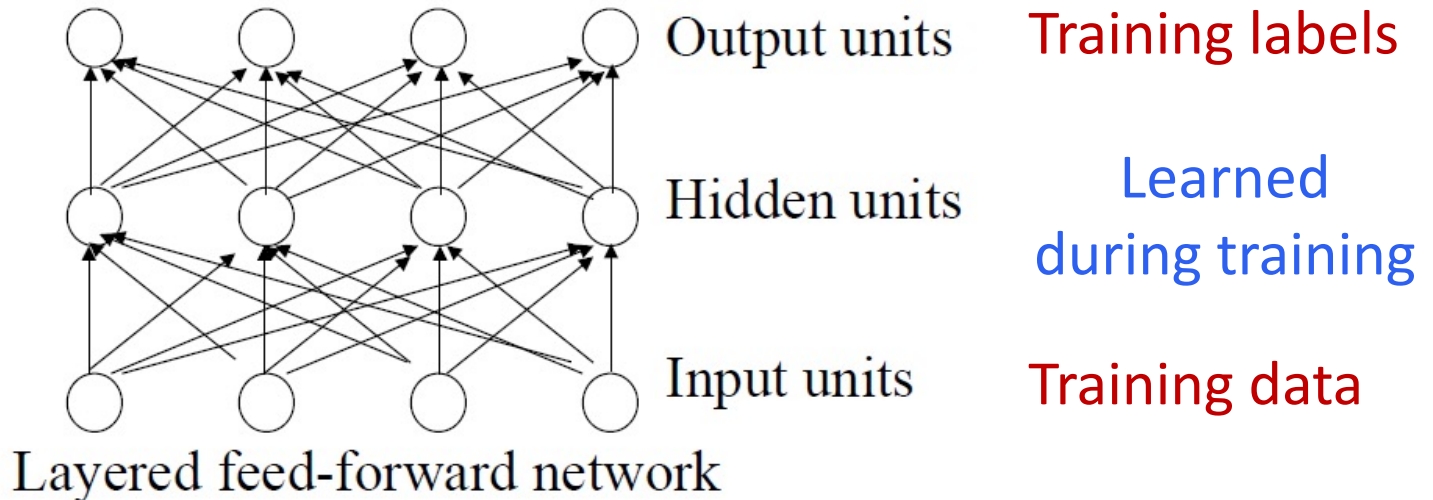
$$F(x) = \max(0, x)$$

Artificial Neuron

# 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

# 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**



# Example

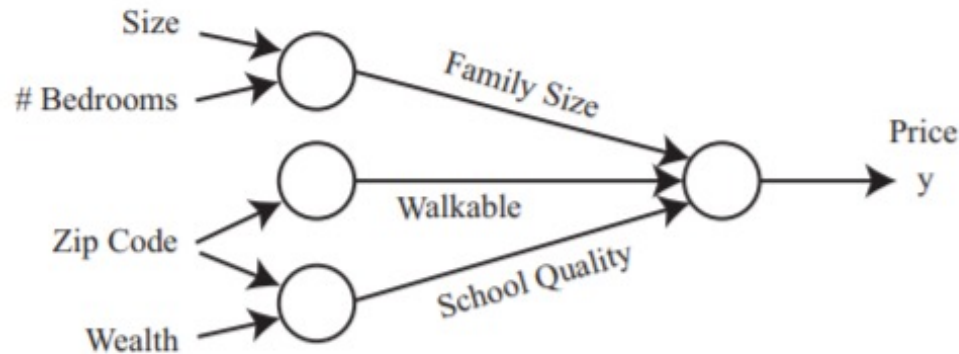
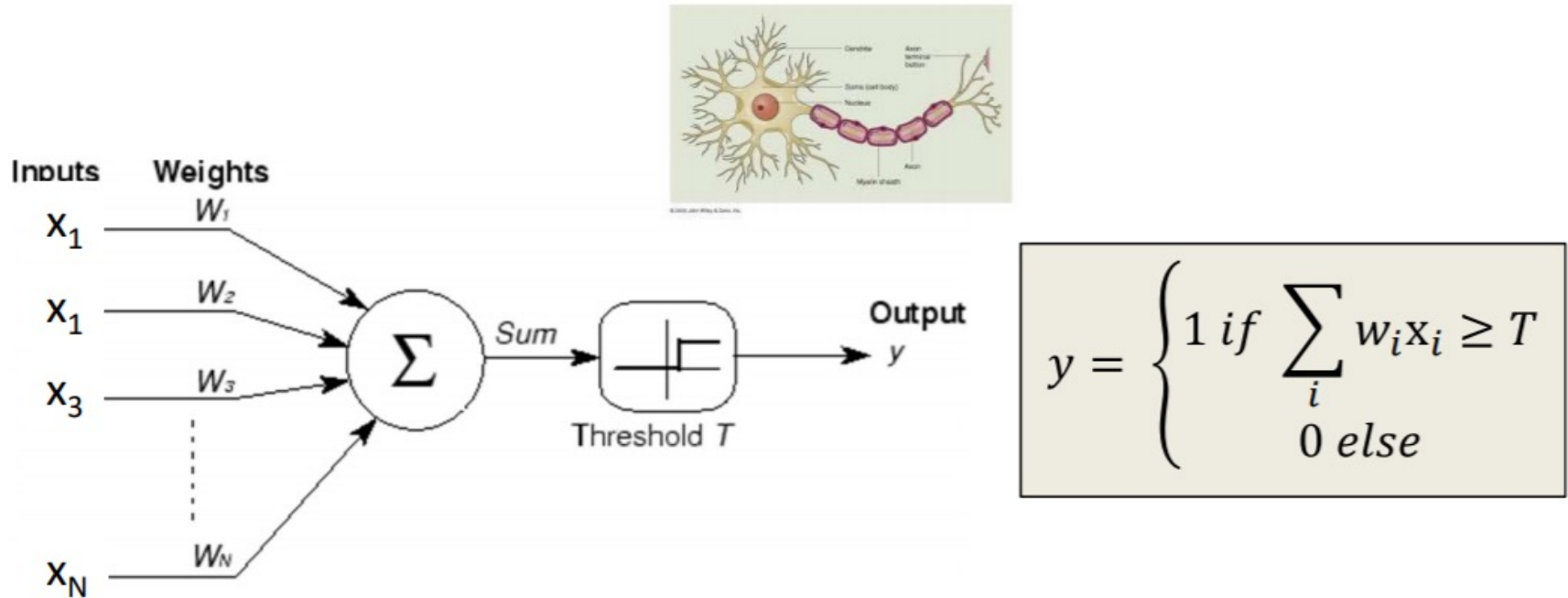


Figure 2: Diagram of a small neural network for predicting housing prices.

Intermediate Features: Family size,  
Walkable, School quality

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

# Perceptron



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

# 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  $(x_i, y_i)$

$$\theta_j \leftarrow \theta_j - \frac{1}{2} (h_{\theta}(x_i) - y_i) x_{ij}$$

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

# The Perceptron

- The perceptron uses the following update rule each time it receives a new training instance  $(x_i, y_i)$

$$\theta_j \leftarrow \theta_j - \frac{1}{2} (h_{\theta}(x_i) - y_i) x_{ij}$$

either 2 or -2

- Re-write as  $\theta_j \leftarrow \theta_j + y_i x_{ij}$  (only upon misclassification)

Perceptron Rule: If  $x_i$  is misclassified, do  
 $\theta \leftarrow \theta + y_i x_i$

# Online Perceptron

Let  $\theta \leftarrow [0, 0, \dots, 0]$

Repeat:

    Receive training example  $(x_i, y_i)$

    If  $y_i \theta^T x_i \leq 0$  // prediction is incorrect

$\theta \leftarrow \theta + y_i x_i$

Until stopping condition

**Online learning** – the learning mode where the model update is performed each time a single observation is received

**Batch learning** – the learning mode where the model update is performed after observing the entire training set

# Batch Perceptron

Let  $\theta \leftarrow [0,0,\dots,0]$

Repeat:

$\Delta = [0,0, \dots, 0]$

For  $i = 1$  to  $N$                       // Consider all training examples

    If  $y_i \theta^T x_i \leq 0$               // Prediction is incorrect

$\Delta \leftarrow \Delta + y_i x_i.$     // Accumulate all errors

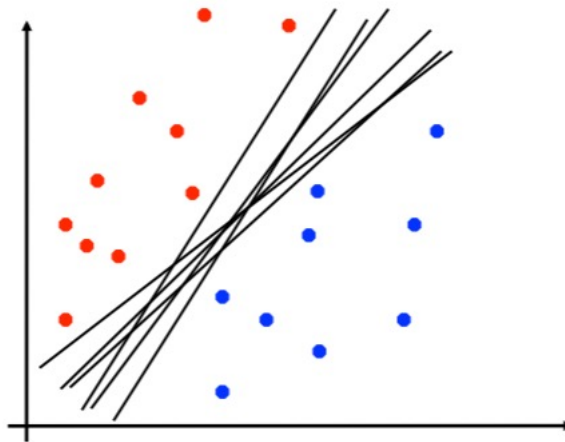
$\theta \leftarrow \theta + \frac{\Delta}{N}$               // Parameter update rule

Until stopping condition

- Guaranteed to find separating hyperplane if data is linearly separable

# Perceptron Limitations

- Is dependent on starting point
- It could take many steps for convergence
- Perceptron can overfit
  - Move the decision boundary for every example



Which of this is optimal?

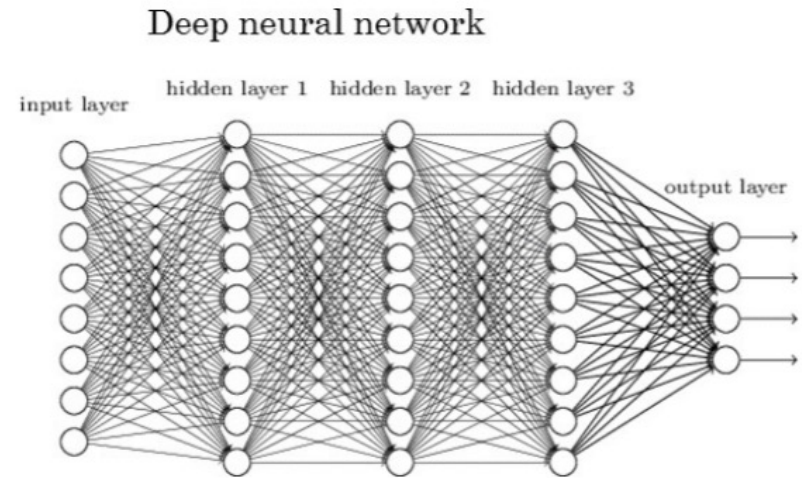
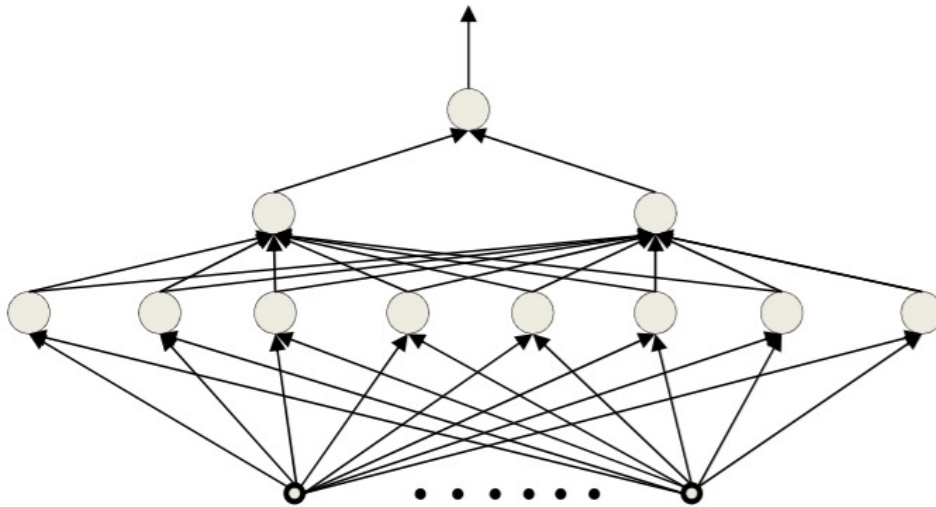
# History of Perceptrons

- They were popularised by Frank Rosenblatt in the early 1960's.
  - They appeared to have a very powerful learning algorithm.
  - Lots of grand claims were made for what they could learn to do.
- In 1969, Minsky and Papert published a book called “Perceptrons” that analysed what they could do and showed their limitations.
  - Many people thought these limitations applied to all neural network models.
- The perceptron learning procedure is still widely used today for tasks with enormous feature vectors that contain many millions of features.

They are the basic building blocks for  
Deep Neural Networks



# Multi-Layer Perceptron

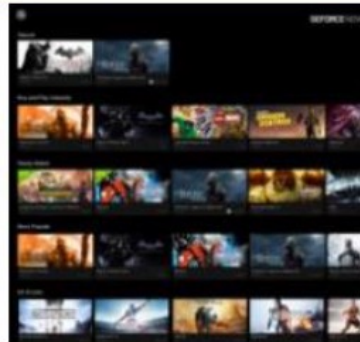
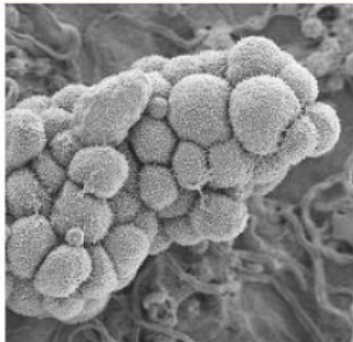


- A network of perceptrons
  - Generally “layered”



# 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.techneworld.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

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

Google+ 5  
Tweet 25  
Share 45  
LinkedIn Share 11  
Share 0  
share 104




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.

**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**

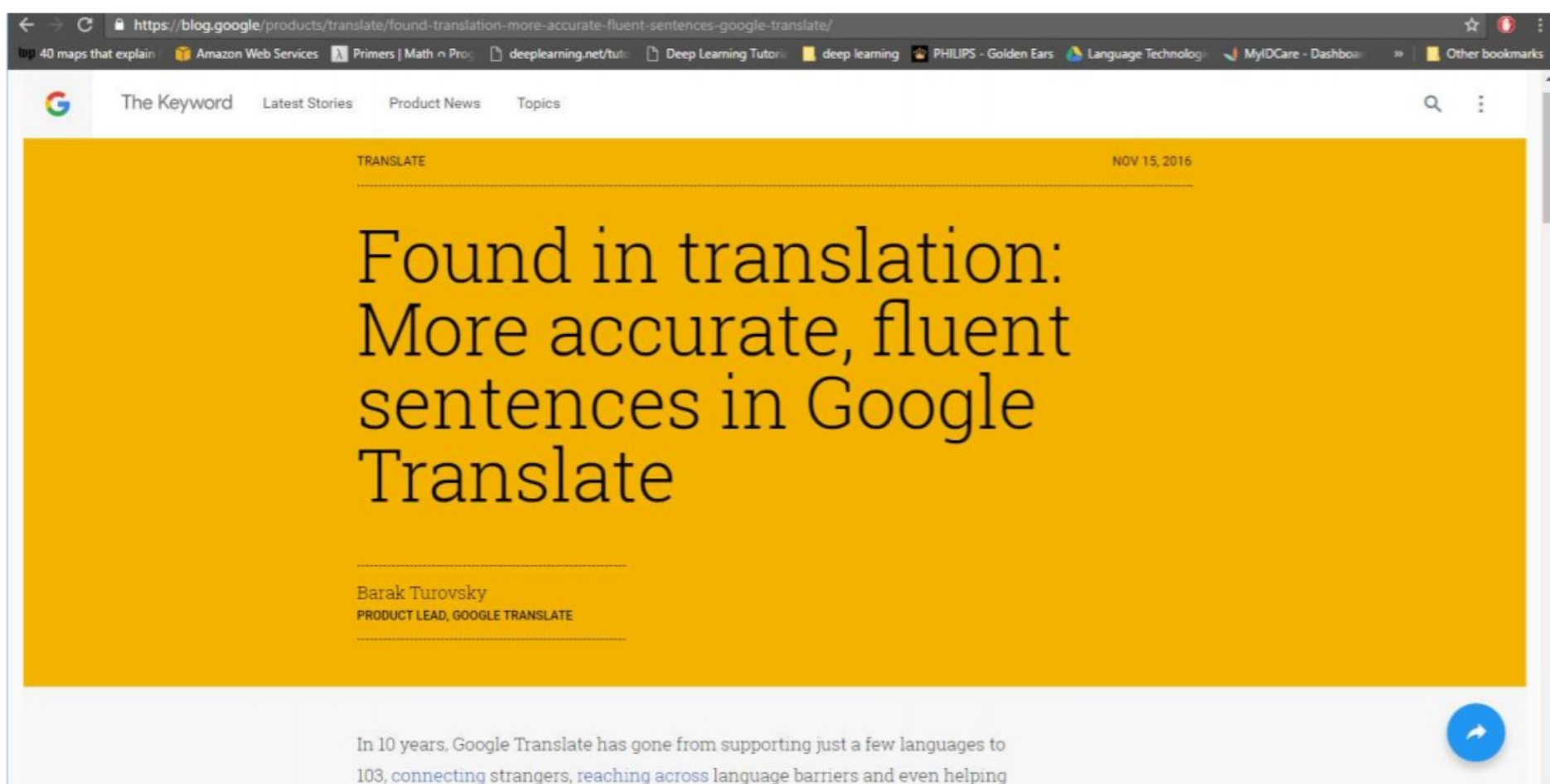
**Pay TV's Newest Innovation: Giving Users Control**

**Apple Celebrates Itself in \$300 Coffee Table Tome**

**AWS Enjoys Top Perch in IaaS, PaaS Markets**

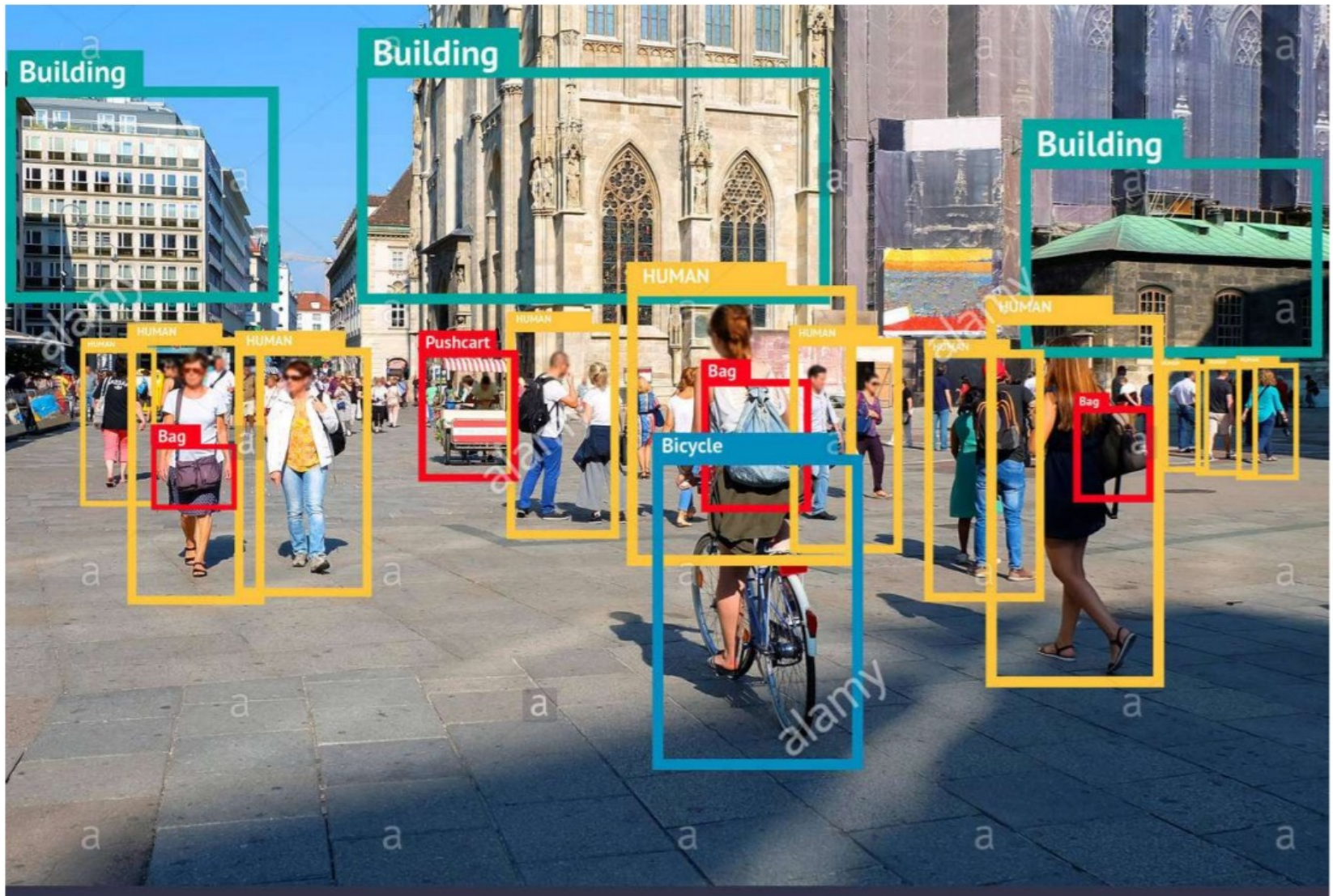
**US Comptroller Gears Up for Blockchain and**

# Success stories: Machine Translation





# Success stories: Image segmentation

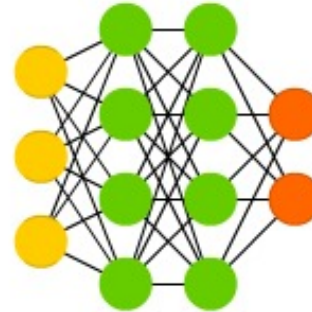


# Neural Network Architectures

## Feed-Forward Networks

- Neurons from each layer connect to neurons from next layer

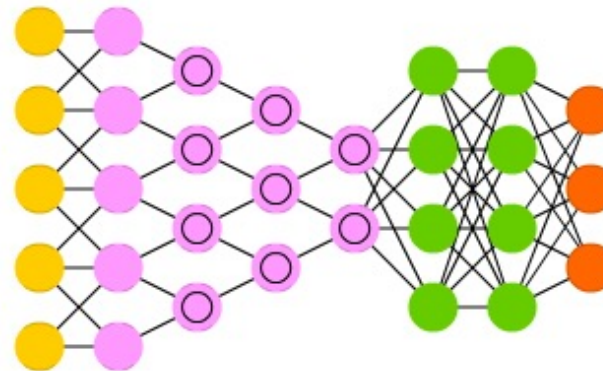
Deep Feed Forward (DFF)



## Convolutional Networks

- Includes convolution layer for feature reduction
- Learns hierarchical representations

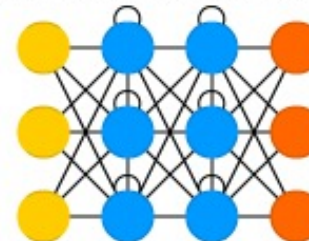
Deep Convolutional Network (DCN)



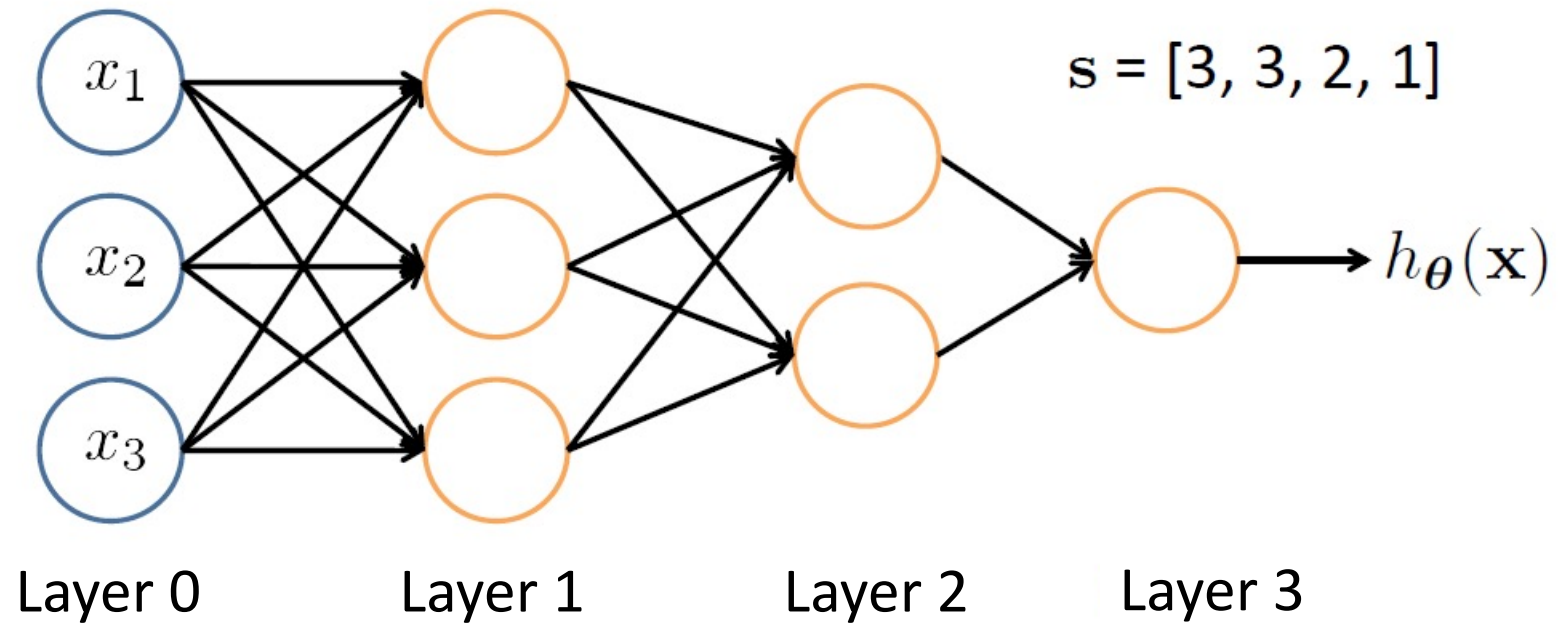
## Recurrent Networks

- Keep hidden state
- Have cycles in computational graph

Recurrent Neural Network (RNN)



# Feed-Forward Networks



$L$  denotes the number of layers

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

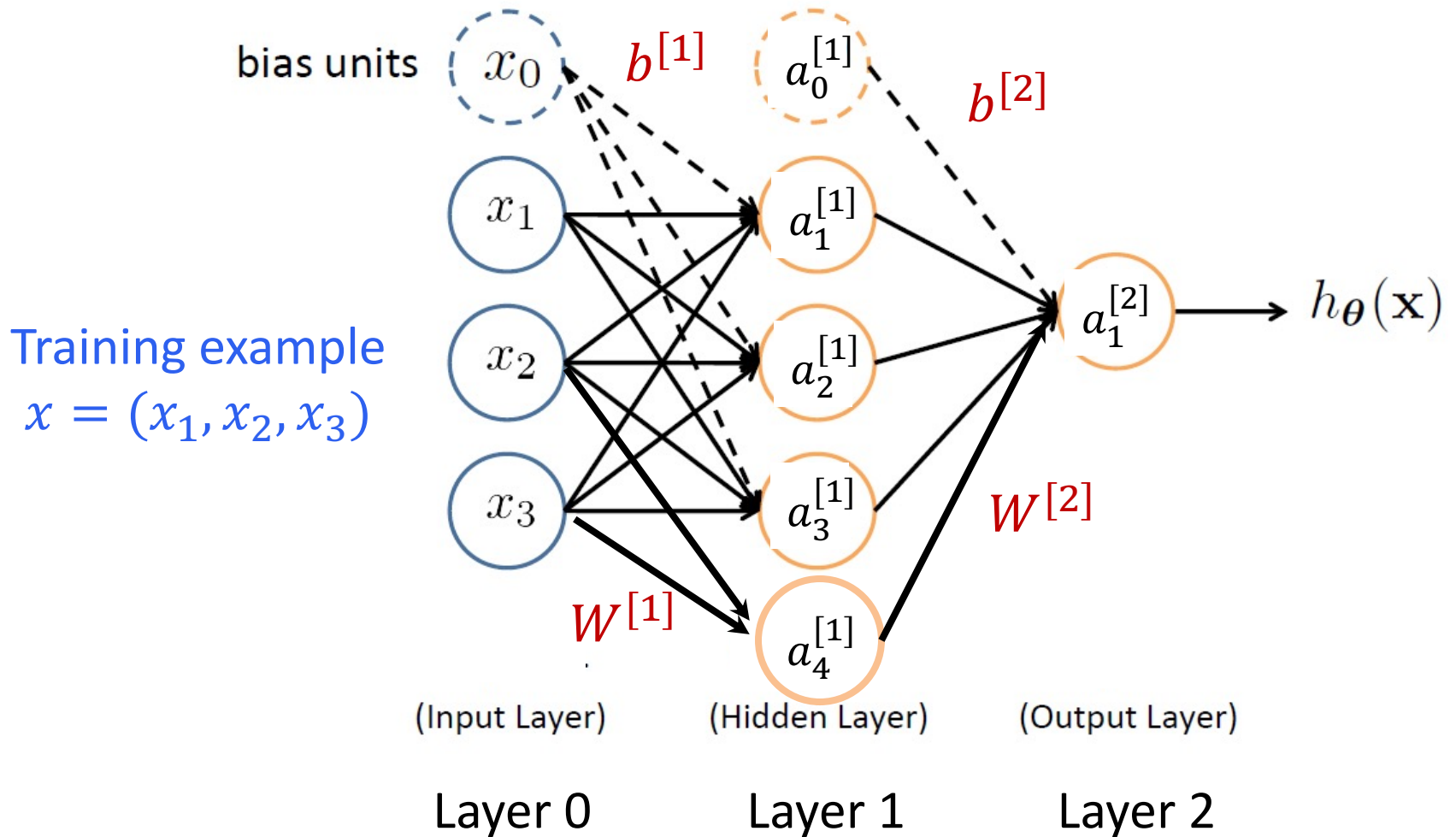
- Not counting bias units
- Typically,  $s_0 = d$  (# input features) and  $s_{L-1} = K$  (# classes)

# 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



# Feed-Forward Neural Network



No cycles

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

# Vectorization

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

$$\vdots$$

$$\vdots$$

$$\vdots$$

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

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

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

Linear

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

Non-Linear

# Review

- Perceptrons are limited in their ability to learn
- Feed-Forward Neural Networks are the most common neural networks architectures
  - Fully connected networks are called Multi-Layer Perceptron
- Input, output, and hidden layers
  - Linear matrix operations followed by non-linear activations at every layer
- Activations:
  - Examples: ReLU, sigmoid
  - Non-linear function

# Acknowledgements

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