PollEverywhere, use Penn Email for attendance!

History of Deep Learning

CIS 522: Lecture 1
1/16/20

Penn Engineering

Key questions in this course

1. How do we decide which problems to tackle with deep learning?
2. Given a problem setting, how do we determine which models are promising?
3. What's the best way to implement said model?
4. How can we best visualize, explain, and justify our findings?
5. How can neuroscience inspire deep learning?

What we're covering

1. Fundamentals of Deep Learning (Weeks 1-4)
2. Computer Vision (Weeks 5-6)
3. NLP (Weeks 7-8)
4. Reinforcement Learning (Week 9-10)
5. Special Topics (Weeks 11-15)

Key questions covered by other courses

1. **CIS 580, 581:** What are the foundations relating vision and computation?
2. **CIS 680:** What is the SOTA* architecture for _ problem domain in computer vision?
3. **CIS 530:** What are the foundations relating natural language and computation?
4. **ESE 546:** How and Why does the mathematics behind different architecture work?
5. **CIS 700-001:** What is the SOTA* architecture for _ problem domain in NLP?
6. **STAT 991:** What is the cutting-edge of deep learning research?

* SOTA = State of the Art

Logistics & Course Materials

- Website (cis522.com)
- Piazza (https://piazza.com/upenn/spring2020/cis522)
- Gradescope (Let us know if you haven't received an invite!)
- Waitlist
- Optional Recitations (Starting next Friday, 1/24 at 11am in Towne 100)

Who is teaching this course

Konrad Kording
Instructor
Prof. Neuroscience / BE
Recovering Neuroscientist

David Rolnick
Instructor
Postdoc / Math. PhD
Founder of Climate Change AI

Sadat Shaik
Head TA
CIS MSE / BSE
Super Senior in Denial

Who is teaching this course

Brandon Lin

Chetan Tutika

Dewang Sultania

Nidhi Seethapathi

Rohan Menezes

Jordan Lei

Saket Karve

Vatsal Chanana

Yonah Mann

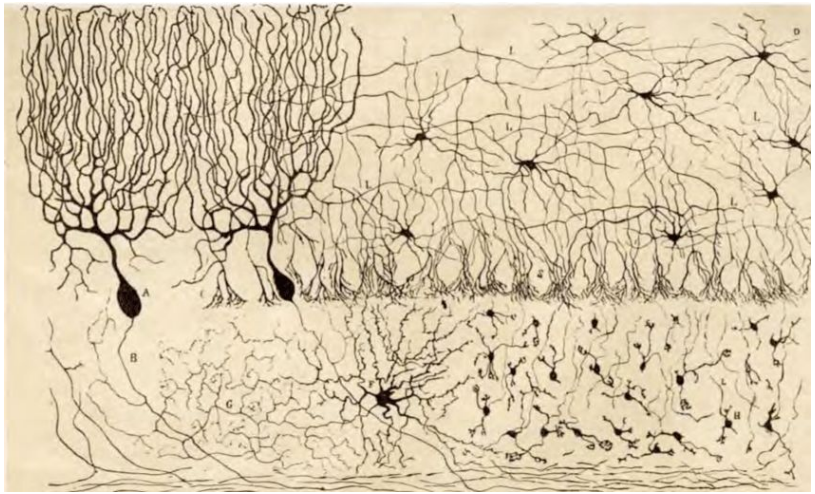
Kushagra Goel

Jianqiao Wangni

Ben Lansdell

History of Neural Networks

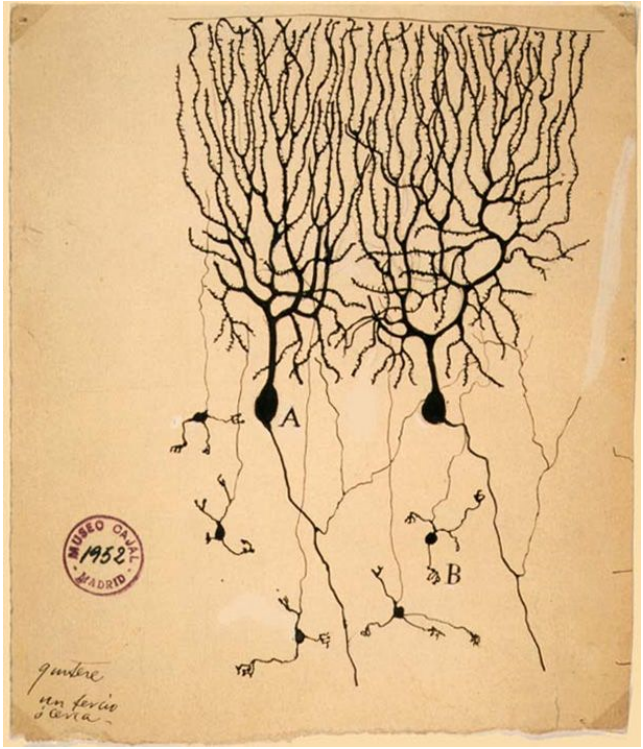
The Neuron Doctrine - Santiago Ramon y Cajal (1888)



<u>Idea:</u> The nervous system is **not** just one continuous thread-like cell, but rather composed of **multiple individual cells**, later called *neurons* by anatomist H. Waldeyer-Hartz.

Picture from *Revista Trimestral de Histología Normal y Patológica by Santiago Ramon y Cajal* from which the Neuron Doctrine originated.

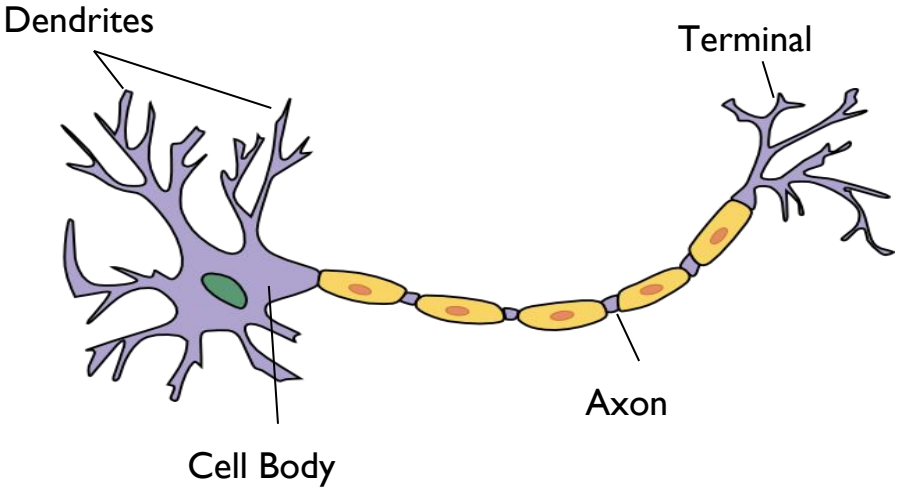
Law of Dynamic Polarization (also Cajal)

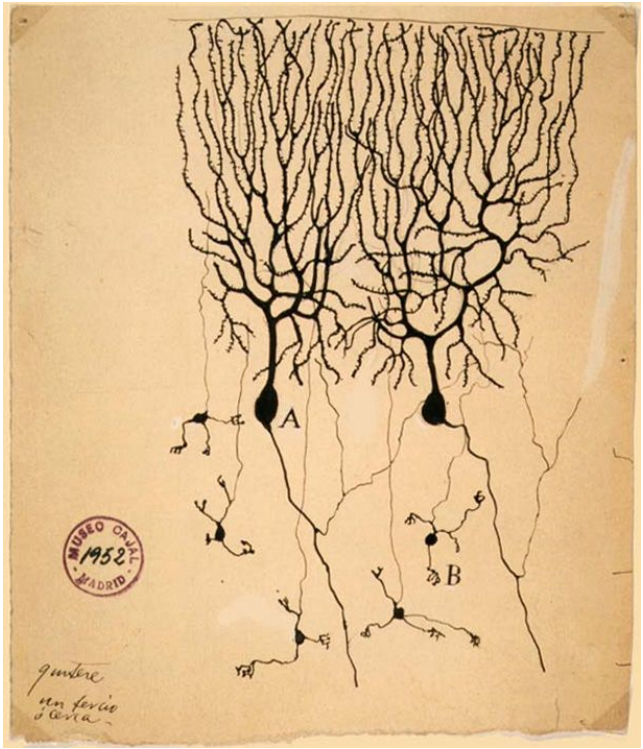


Picture from *Revista Trimestral de Histología Normal y Patológica by Santiago Ramon y Cajal* from which the Neuron Doctrine originated.

Law of Dynamic Polarization (also Cajal)

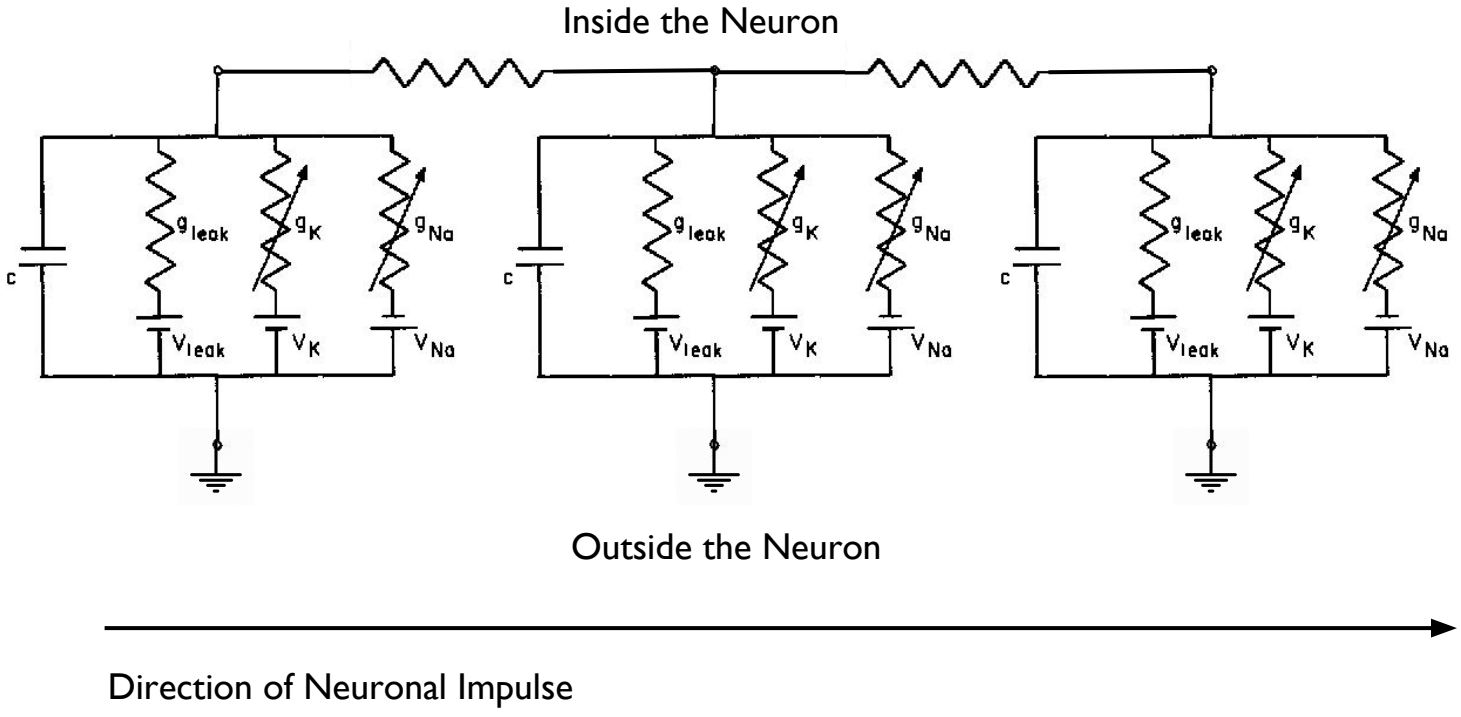
Law: Information travels in **one direction,** from the dendrites to the cell body through the axon and to the terminal.



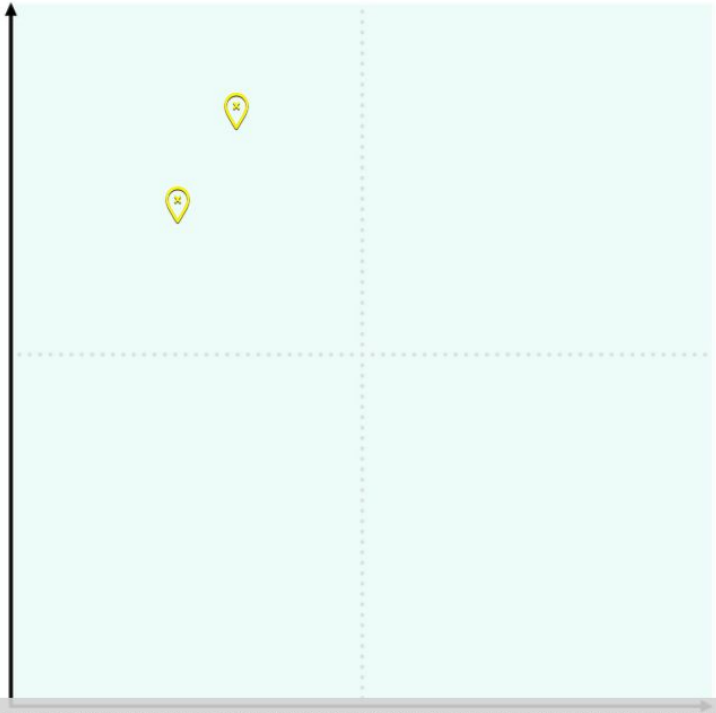


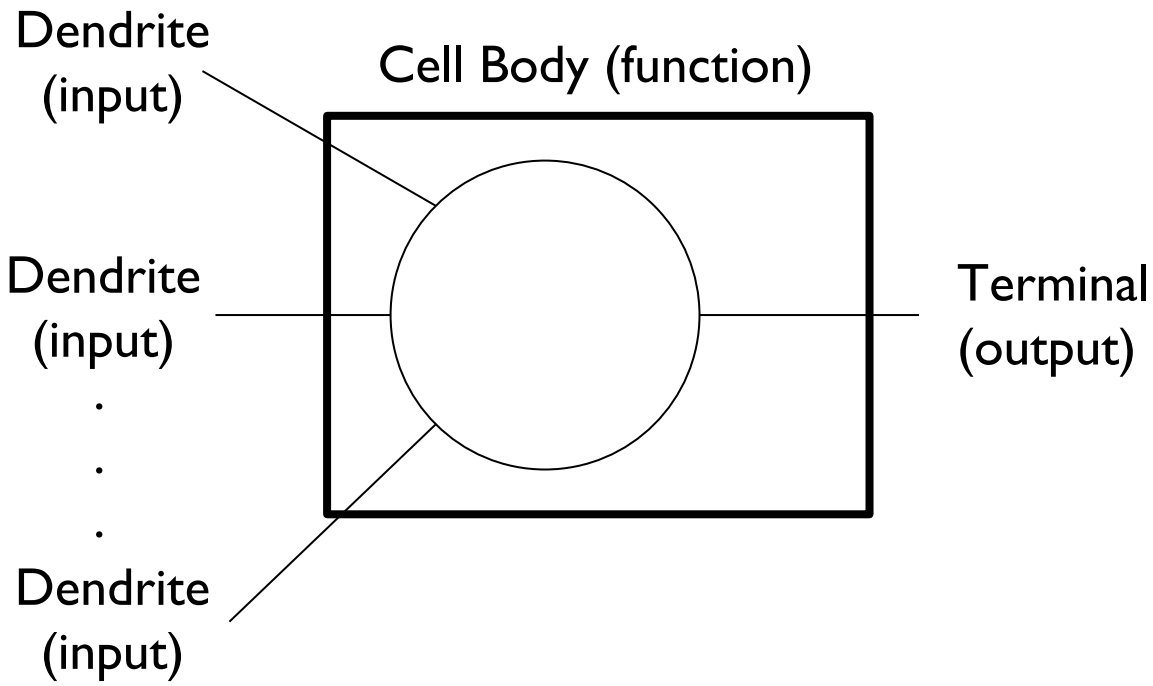
Neuron Diagram from: https://simple.wikipedia.org/wiki/Neuron

An axon

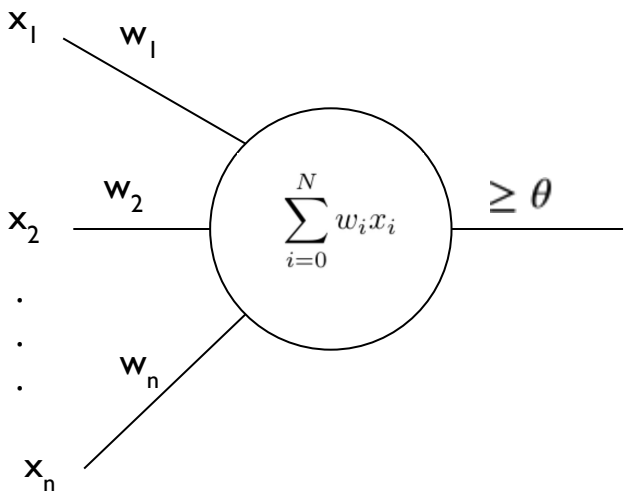

Inside the Neuron
C
g_{leak}
g_K
g_{Na}
V_{leak}
V_K
V_{Na}
Outside the Neuron
Direction of Neuronal Impulse

Love for theory (more =higher) , love for practice (more = right)




Dendrite
(input)
Cell Body (function)
Dendrite
(input)
Terminal
(output)
Dendrite
(input)

McCullochs - Pitts
Linear Threshold Unit (LTU), 1943



Binary inputs

Binary outputs

Inputs:

- $x \in \{0, 1\}$
- $\theta \in \mathbb{Z}^*$
- $w \in \mathbb{R}^n$

Outputs:

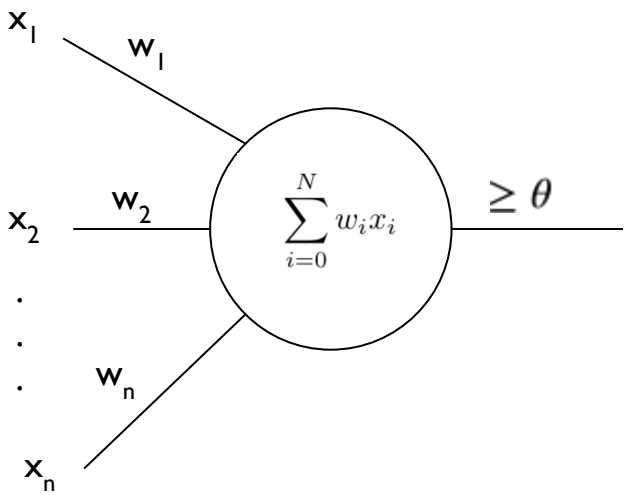
- $y \in \{0, 1\}$

Function:

$$\sum_{i=0}^{N} w_i x_i > \theta$$

Note: We do not learn the weights.

McCullochs - Pitts
Linear Threshold Unit (LTU), 1943



Inputs:

- $x \in \{0, 1\}$
- $\theta \in \mathbb{Z}^*$
- $w \in \mathbb{R}^n$

Outputs:

- $y \in \{0, 1\}$

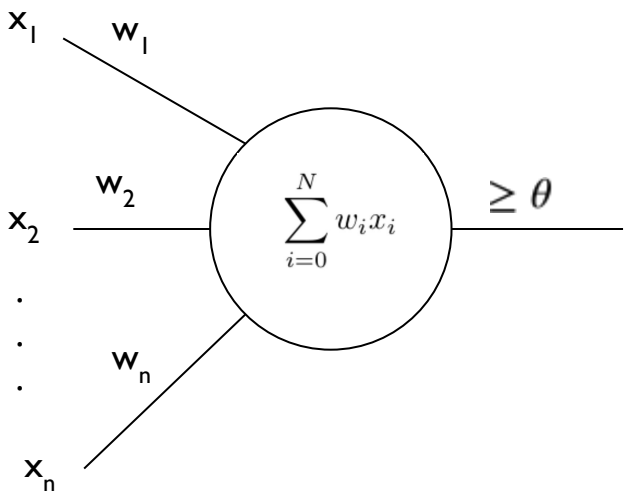
Function:

$$\sum_{i=0}^{N} w_i x_i > \theta$$

A sum is a **linear** transformation and this sum is then **thresholded** by theta.

Hence, ***Linear Threshold Unit!***

McCullochs - Pitts
Linear Threshold Unit (LTU), 1943



Inputs:

- $x \in \{0, 1\}$
- $\theta \in \mathbb{Z}^*$
- $w \in \mathbb{R}^n$

Outputs:

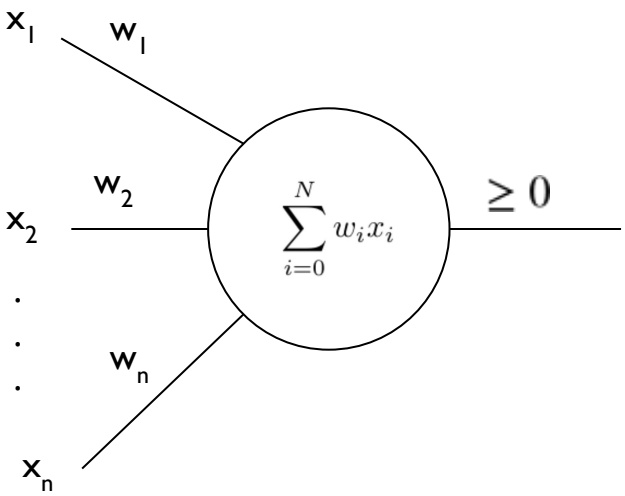
- $y \in \{0, 1\}$

Function:

$$\sum_{i=0}^{N} w_i x_i > \theta$$

Note: We do not learn the weights.

Rosenblatt's perceptron (1958)



Inputs:

- $x \in \mathbb{R}^n$
- $w \in \mathbb{R}^n$

Outputs:

- $y \in \{0, 1\}$

Function:

$$y = x \cdot w \geq 0$$

Note: We will now learn the weights.

Continuous inputs

Binary output

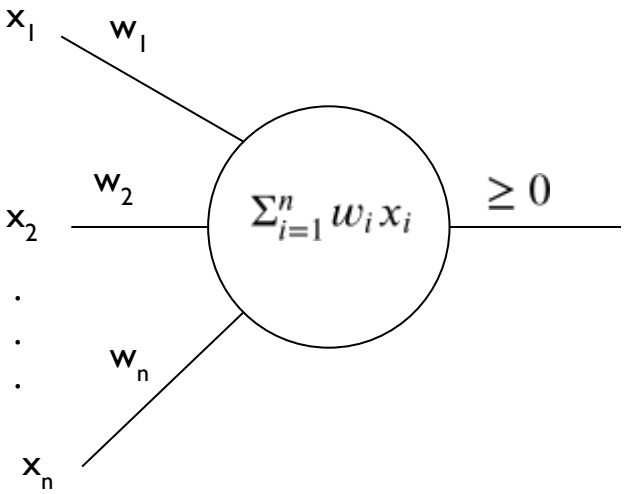
A note on formalism

$$\sum_{i=0}^{N} w_i x_i > \theta$$

$$\sum_{i=0}^{N} w_i x_i - \theta * 1 > 0$$

Biases are irrelevant and ugly
Just append 1 to the vector of all inputs.

Rosenblatt's perceptron (1958)



Inputs:

- $x \in \mathbb{R}^n$
- $w \in \mathbb{R}^n$

Outputs:

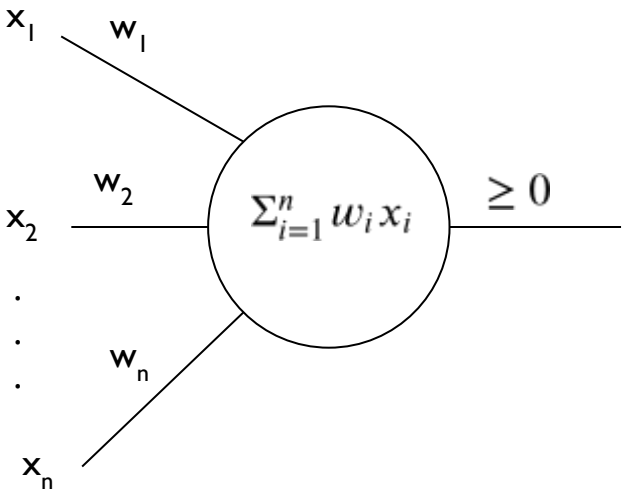
- $y \in \{0, 1\}$

Function:

$$y = x \cdot w \geq 0$$

What happened to θ?

Rosenblatt's perceptron (1958)



Inputs:

- $x \in \mathbb{R}^n$
- $w \in \mathbb{R}^n$

Outputs:

- $y \in \{0, 1\}$

Function:

$$y = x \cdot w \geq 0$$

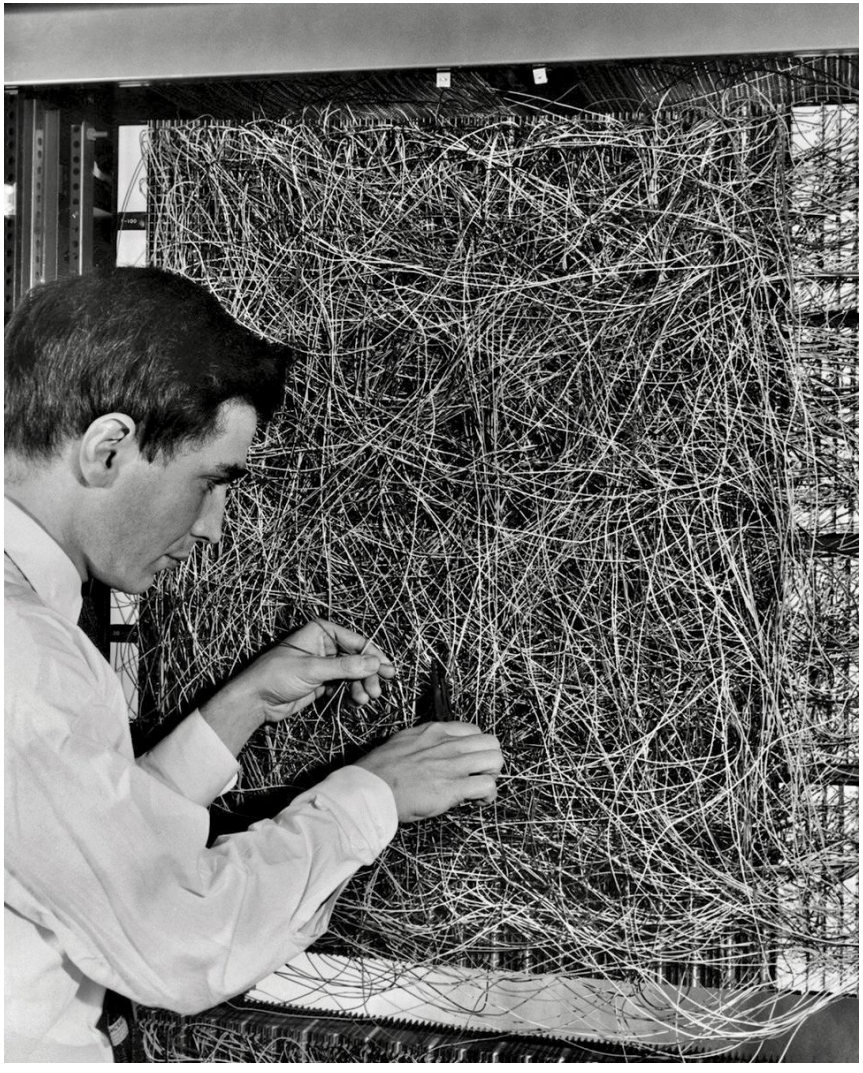
What happened to θ?

One of x's features is 1 and the corresponding weight is $-\theta$.

Rosenblatt's Perceptron - Algorithm

1. Initialize w_0 to a random vector
2. Give it a training example (x, y)
3. Input training example x in to perceptron, denote the output as $\hat{y}$.
4. $(y - \hat{y})$ is the error
5. At iteration t denote $w_{t+1} = w_t + \alpha(y - \hat{y})x$ (α is the learning rate)
6. Repeat sets 2-4 for desired number of iterations.

Mark I



Animated

A perceptron is characterized by a Hyperplane and the two half-spaces

$$\mathcal{H} = \{x : W^T x = 0\}$$ Hyperplane defined by weights

$$\mathcal{H}_+ = \{x : W^T x > 0\}$$ Halfspace where y should be +1

$$\mathcal{H}_- = \{x : W^T x \leq 0\}$$ Halfspace where y should be -1 (or 0)

Batch vs online learning

Give one stimulus at a time

Or all of them at the same time

Or groups of them at a time

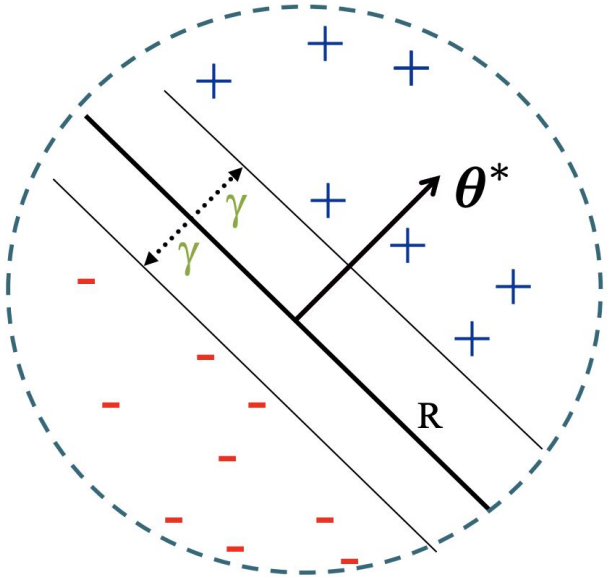
Does it make a difference?

Analysis of Perceptron

Perceptron Mistake Bound

Guarantee: If data has margin γ and all points inside a ball of radius R, then Perceptron makes $\leq (R/\gamma)^2$ mistakes.

(Normalized margin: multiplying all points by 100, or dividing all points by 100, doesn't change the number of mistakes; algo is invariant to scaling.)



Following slides:
Matt Gormley

Perceptron Mistake Bound

Theorem 0.1 (Block (1962), Novikoff (1962)).

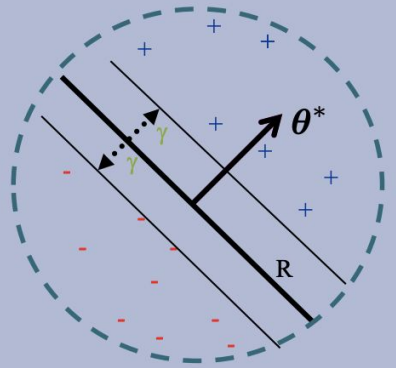
Given dataset: $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$.

Suppose:

1. *Finite size inputs:* $||x^{(i)}|| \leq R$
2. *Linearly separable data:* $\exists \boldsymbol{\theta}^*$ *s.t.* $||\boldsymbol{\theta}^*|| = 1$ *and* $y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)}) \geq \gamma, \forall i$

Then: The number of mistakes made by the Perceptron algorithm on this dataset is

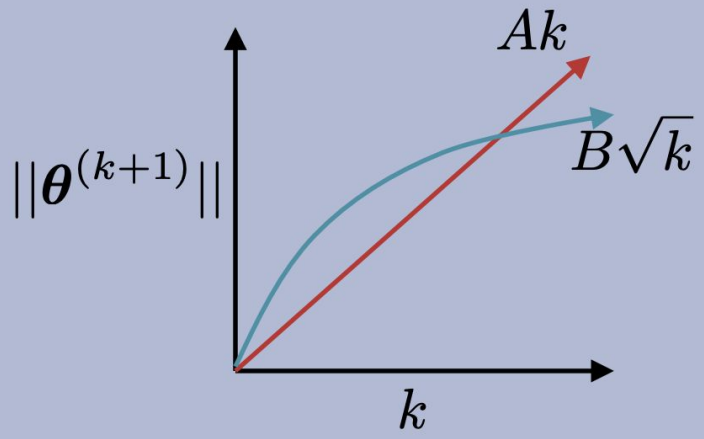
$$k \leq (R/\gamma)^2$$



Proof of Perceptron Mistake Bound:

We will show that there exist constants A and B s.t.

$$Ak \leq ||\boldsymbol{\theta}^{(k+1)}|| \leq B\sqrt{k}$$



Proof of Perceptron Mistake Bound:

Part 1: for some A, $Ak \leq ||\boldsymbol{\theta}^{(k+1)}||$

$$\boldsymbol{\theta}^{(k+1)} \cdot \boldsymbol{\theta}^* = (\boldsymbol{\theta}^{(k)} + y^{(i)}\mathbf{x}^{(i)})\boldsymbol{\theta}^*$$

by Perceptron algorithm update

$$= \boldsymbol{\theta}^{(k)} \cdot \boldsymbol{\theta}^* + y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)})$$

$$\geq \boldsymbol{\theta}^{(k)} \cdot \boldsymbol{\theta}^* + \gamma$$

by assumption

$$\Rightarrow \boldsymbol{\theta}^{(k+1)} \cdot \boldsymbol{\theta}^* \geq k\gamma$$

by induction on k since $\theta^{(1)} = \mathbf{0}$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}|| \geq k\gamma$$

since $||\mathbf{w}|| \times ||\mathbf{u}|| \geq \mathbf{w} \cdot \mathbf{u}$ and $||\theta^*|| = 1$

Cauchy-Schwartz inequality

Proof of Perceptron Mistake Bound:

Part 2: for some B, $||\boldsymbol{\theta}^{(k+1)}|| \leq B\sqrt{k}$

$$||\boldsymbol{\theta}^{(k+1)}||^2 = ||\boldsymbol{\theta}^{(k)} + y^{(i)}\mathbf{x}^{(i)}||^2$$

by Perceptron algorithm update

$$= ||\boldsymbol{\theta}^{(k)}||^2 + (y^{(i)})^2||\mathbf{x}^{(i)}||^2 + 2y^{(i)}(\boldsymbol{\theta}^{(k)} \cdot \mathbf{x}^{(i)})$$

$$\leq ||\boldsymbol{\theta}^{(k)}||^2 + (y^{(i)})^2||\mathbf{x}^{(i)}||^2$$

since kth mistake $\Rightarrow y^{(i)}(\boldsymbol{\theta}^{(k)} \cdot \mathbf{x}^{(i)}) \leq 0$

$$= ||\boldsymbol{\theta}^{(k)}||^2 + R^2$$

since $(y^{(i)})^2||\mathbf{x}^{(i)}||^2 = ||\mathbf{x}^{(i)}||^2 = R^2$ by assumption and $(y^{(i)})^2 = 1$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}||^2 \leq kR^2$$

by induction on k since $(\theta^{(1)})^2 = 0$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}|| \leq \sqrt{k}R$$

Proof of Perceptron Mistake Bound:

Part 3: Combining the bounds finishes the proof.

$$k\gamma \leq ||\boldsymbol{\theta}^{(k+1)}|| \leq \sqrt{k}R$$

$$\Rightarrow k \leq (R/\gamma)^2$$

The total number of mistakes must be less than this

Perceptron summary

If the problem is linearly separable
Perceptron problem is relatively quickly solved

So great hype:
"the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence"

Rosenblatt's Perceptron - Proof

Done on chalkboard / in lecture notes.

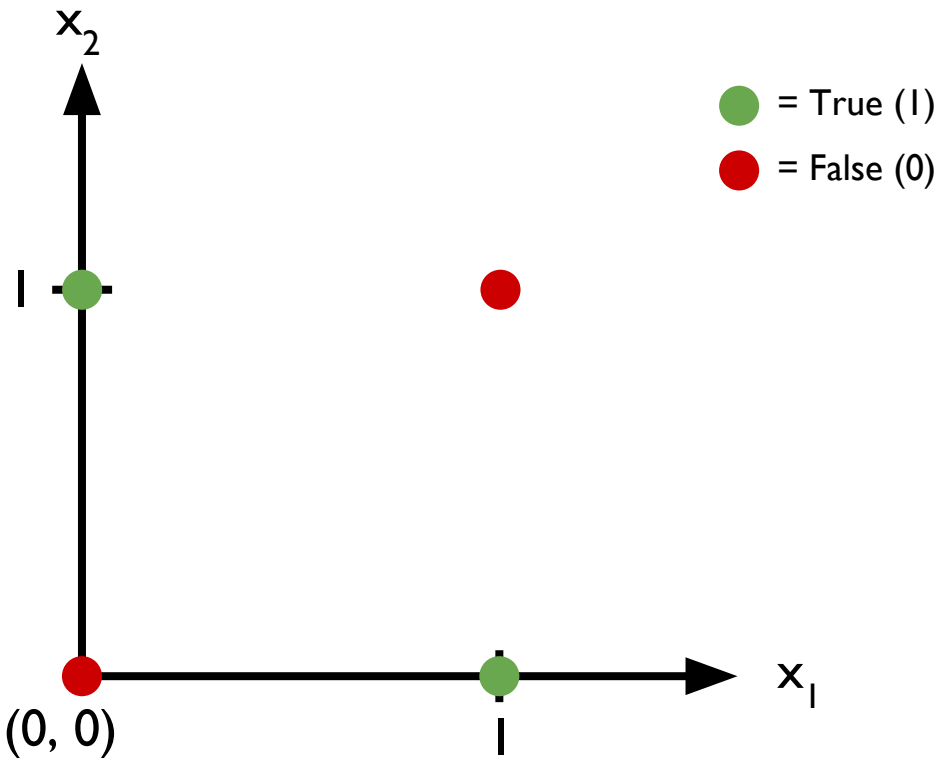
XOR Problem

Problem: Consider the truth table below for the XOR function. Pretty simple right?

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

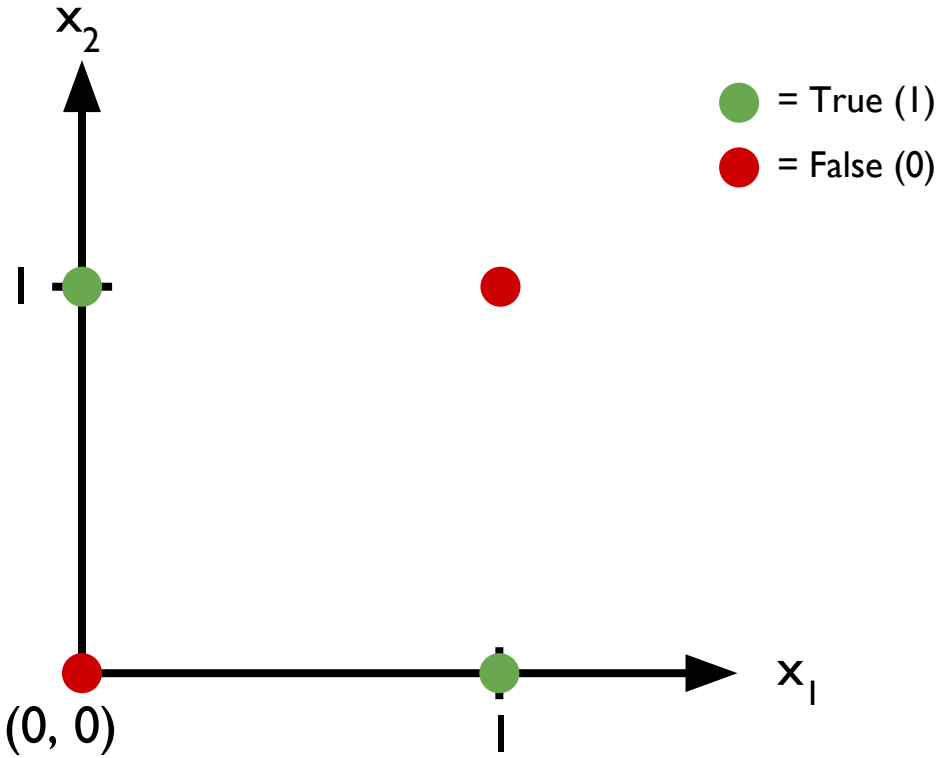
XOR Problem

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0



XOR Problem

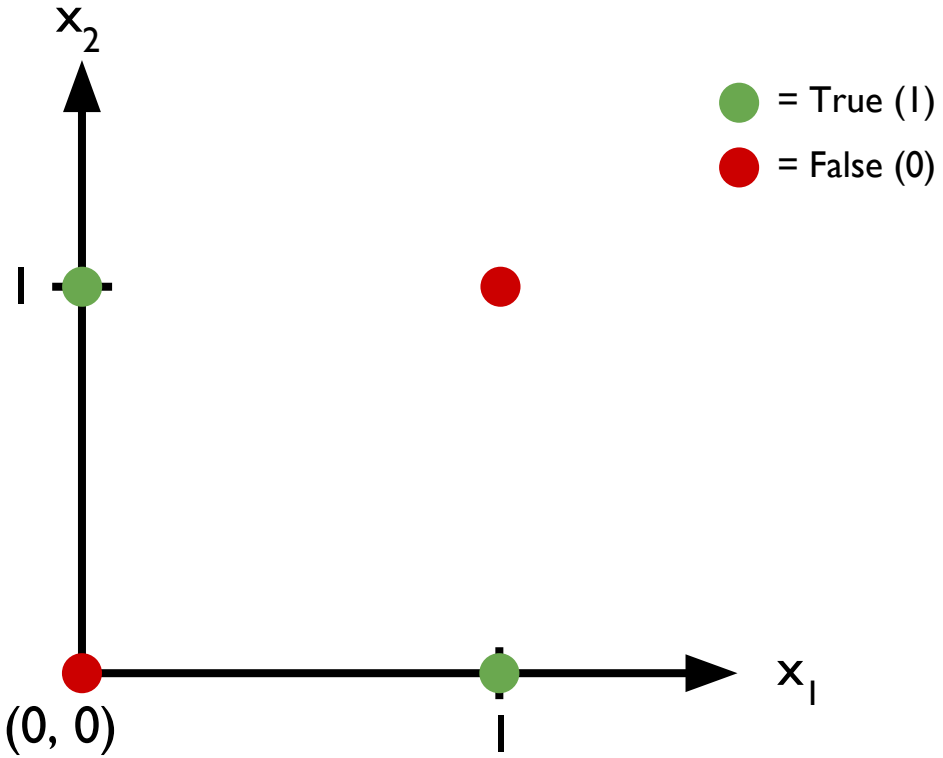
Can you draw a line that separates the positive from negative examples?



XOR Problem

Can you draw a line that separates the positive from negative examples?

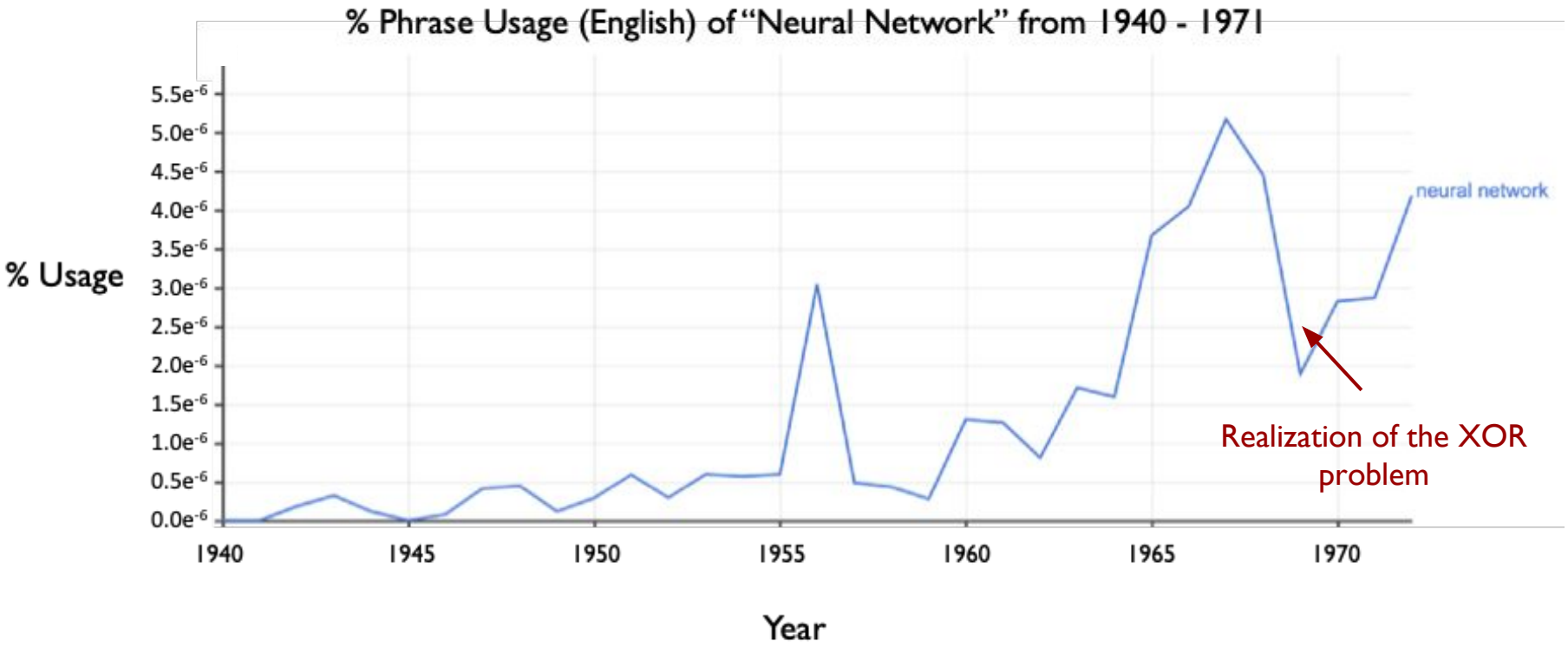
Impossible with the XOR function.



As of 1970, there was one huge problem with neural networks

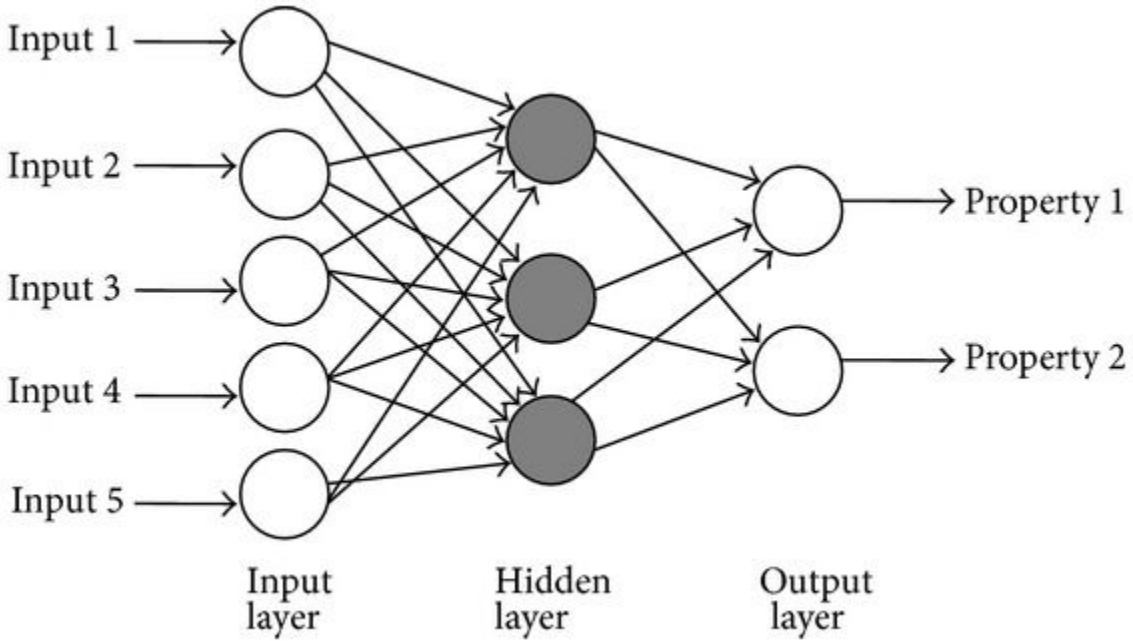
1. Neural networks could not solve any problem that wasn't linearly separable

Winter #1



Graph generated using: https://books.google.com/ngrams

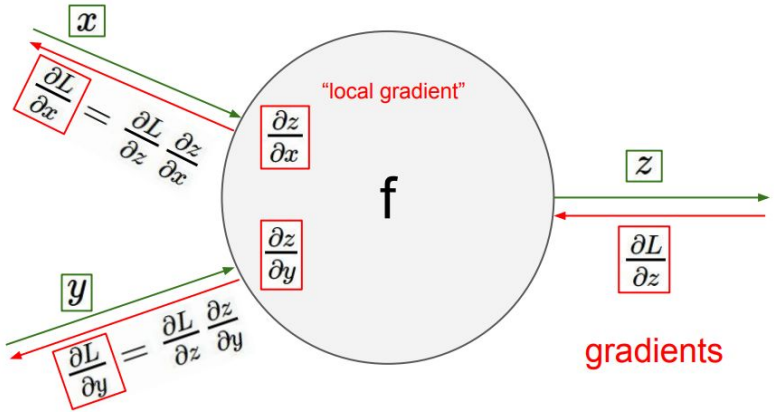
Multi-layer perceptrons



Picture taken from: https://medium.com/pankajmathur/a-simple-multilayer-perceptron-with-tensorflow-3effe7bf3466

Solution: Differentiation/ Backpropagation

- Using chain rule to calculate partial derivative of cost with respect to every parameter.
- Every float operation performed by a computer at some level involves:
  - Elementary binary operators (+, - , x, /)
  - Elementary functions (sin x, cos x, e^x, etc.)

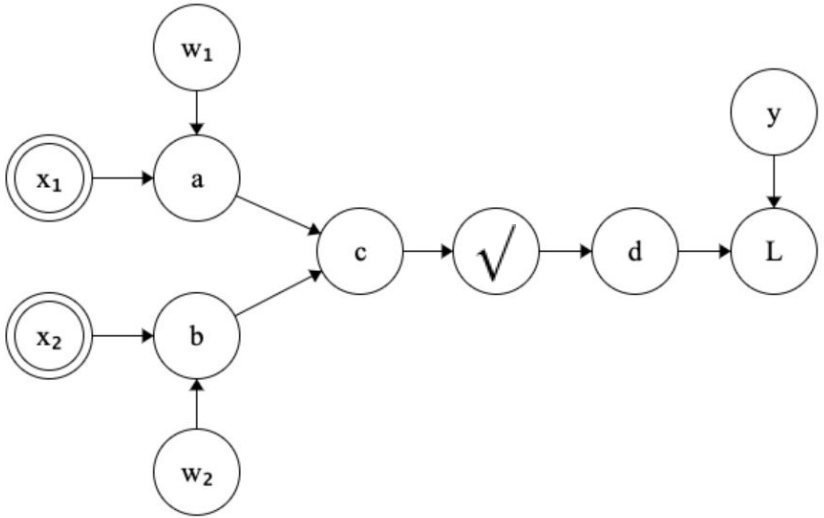


Picture taken from: https://www.quora.com/How-does-backpropagation-happen-in-a-feed-forward-neural-network

Backpropagation Poll

Consider the equation the right, represented visually by the graph below it. L is the Mean Squared Error Loss function. What is $\frac{\partial L}{\partial w_1}$?

Equation: $\hat{y} = \sqrt{w_1 x_1 + w_2 x_2}$



A. $\frac{1}{2}(\frac{\partial L}{\partial c})^{-\frac{1}{2}} \frac{\partial c}{\partial a} \frac{\partial a}{\partial w_1}$

B. $\frac{\partial L}{\partial c} \frac{\partial c}{\partial a} \frac{\partial a}{\partial x_1} \frac{\partial x_1}{\partial w_1}$

C. $\frac{\partial L}{\partial d} \frac{\partial d}{\partial c} \frac{\partial c}{\partial a} \frac{\partial a}{\partial w_1}$

D. None of the above

What is the correct derivative?

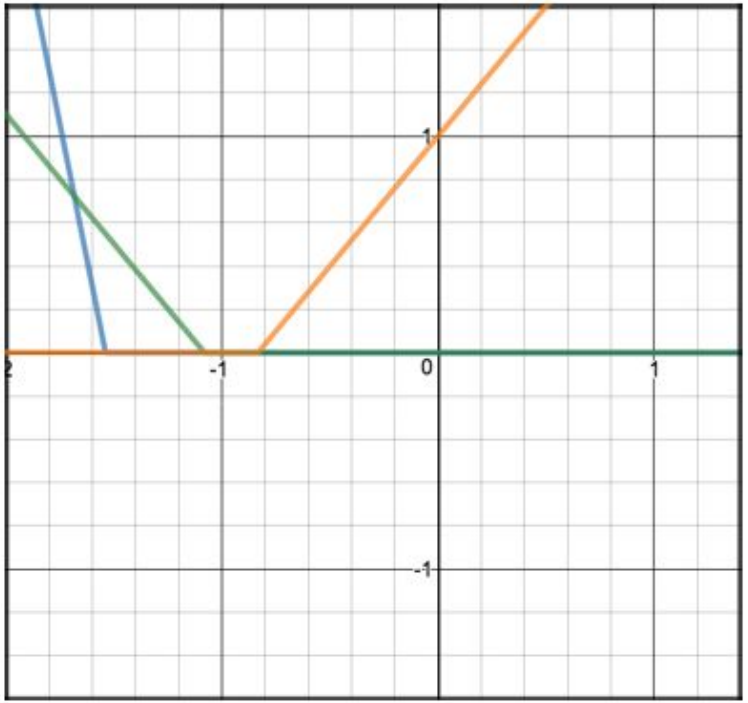
A

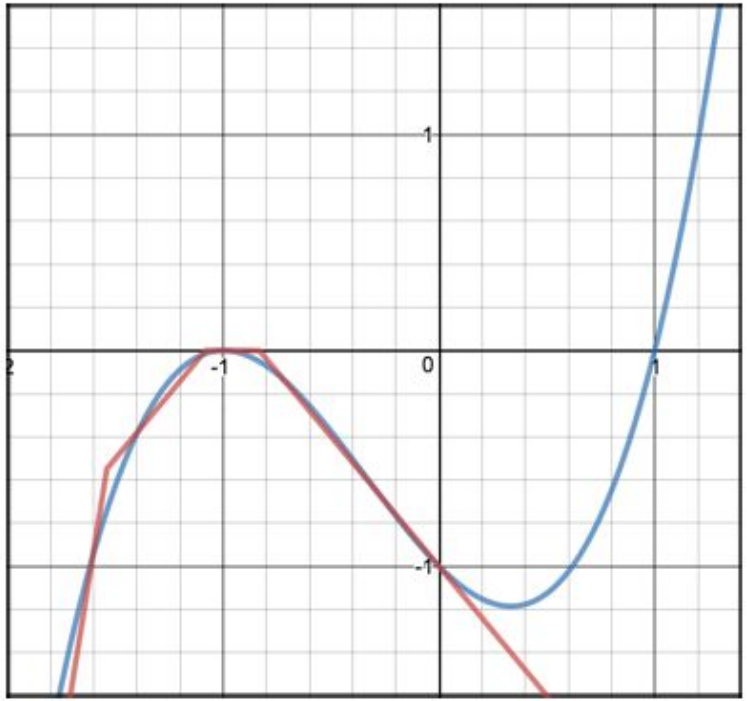
B

C

D

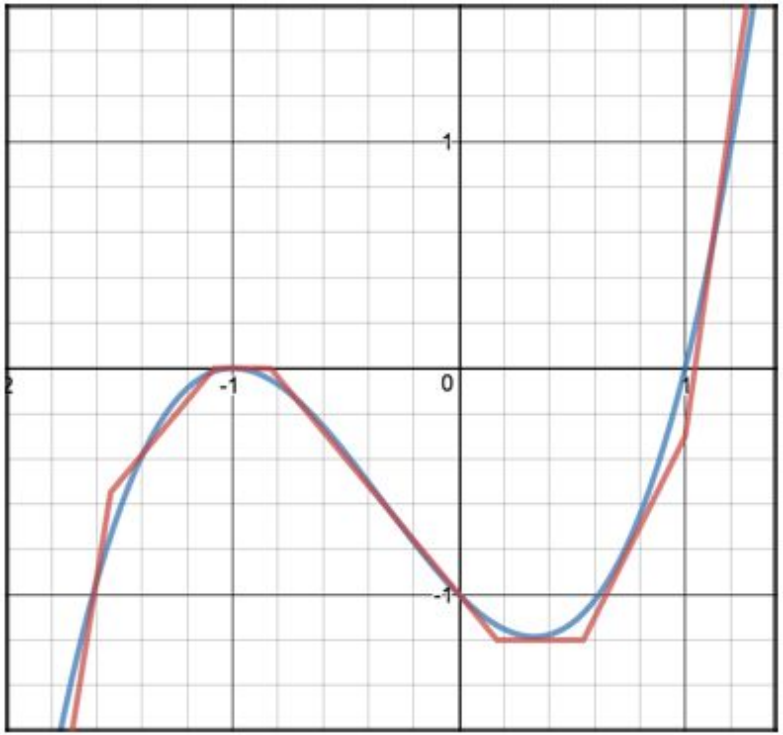
Universal Function Approximator





Images taken from: https://towardsdatascience.com/can-neural-networks-really-learn-any-function-65e106617fc6 (read it! great info)

Universal Function Approximator

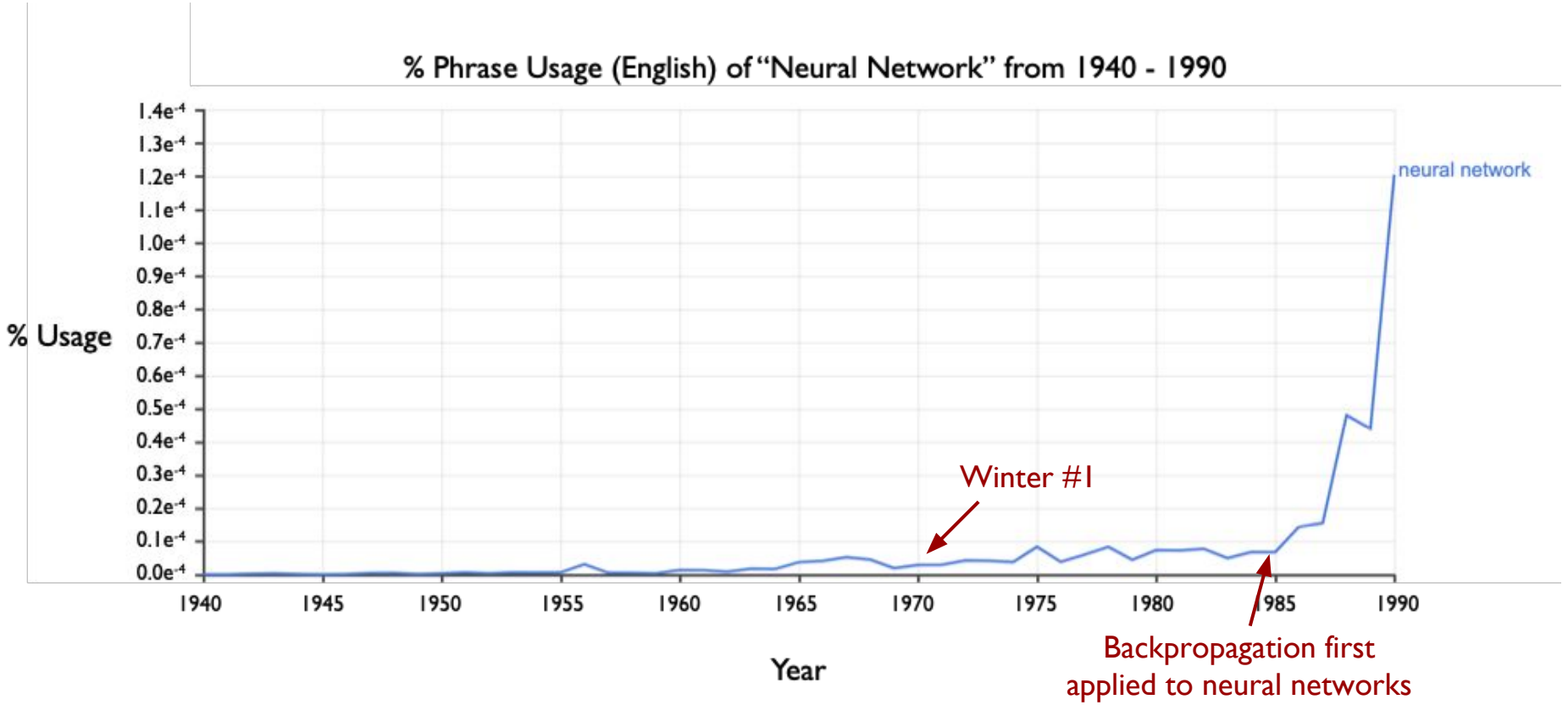


$$n_1(x) = Relu(-5x - 7.7)$$
$$n_2(x) = Relu(-1.2x - 1.3)$$
$$n_3(x) = Relu(1.2x + 1)$$
$$n_4(x) = Relu(1.2x - .2)$$
$$n_5(x) = Relu(2x - 1.1)$$
$$n_6(x) = Relu(5x - 5)$$

$$Z(x) = -n_1(x) - n_2(x) - n_3(x) + n_4(x) + n_5(x) + n_6(x)$$

Images taken from: https://towardsdatascience.com/can-neural-networks-really-learn-any-function-65e106617fc6 (read it! great info)

Resurgence of Interest in Neural Nets



Graph generated using: https://books.google.com/ngrams

As of 1986, there were 2 huge problems with neural nets.

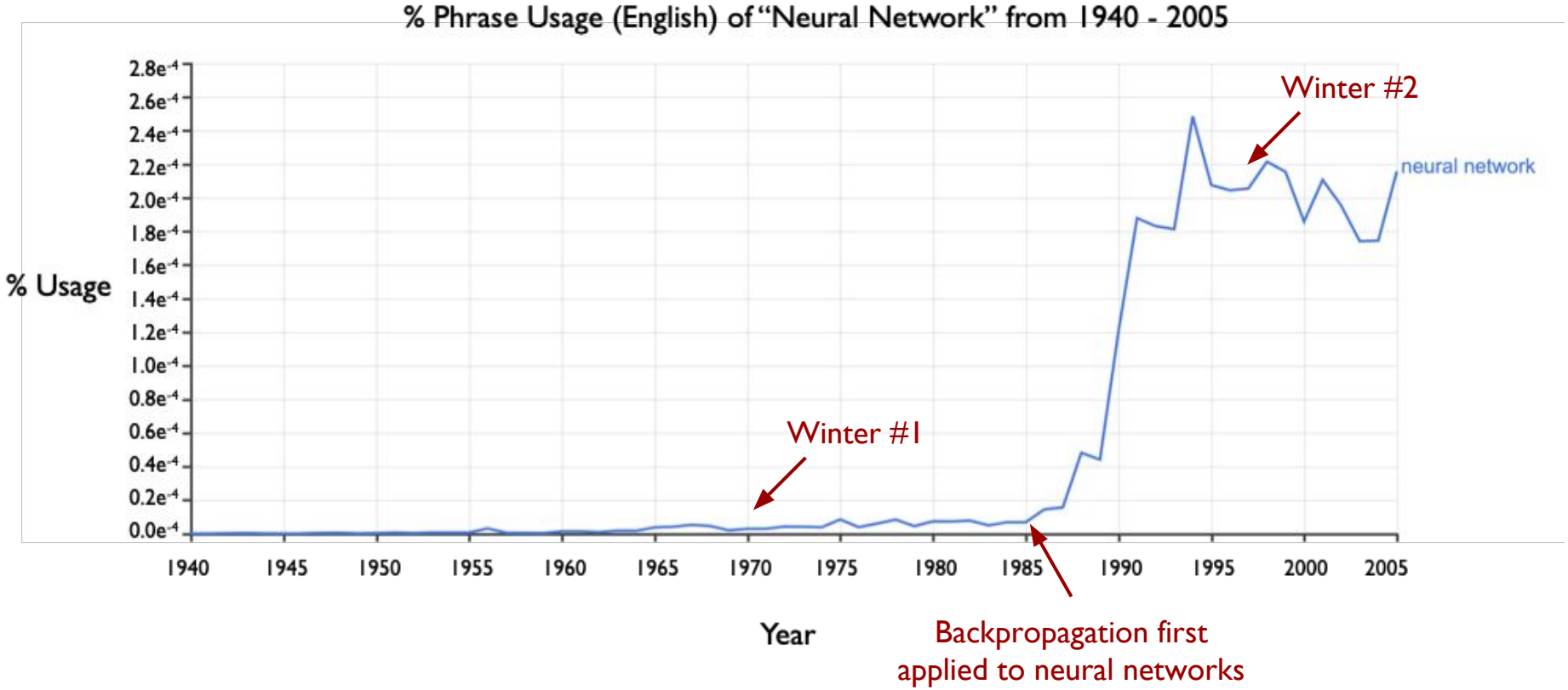
1. ~~Neural networks could not solve any problem that wasn't linearly separable~~
    a. Solved by backpropagation and depth.
2. Backpropagation takes forever to converge!
    a. Not enough compute power to run the model
    b. Not enough labeled data to train the neural net

As of 1986, there were 2 huge problems with neural nets.

1. ~~Neural networks could not solve any problem that wasn't linearly separable~~
    a. Solved by backpropagation and depth.
2. Backpropagation takes forever to converge!
    a. Not enough compute power to run the model
    b. Not enough labeled data to train the neural net

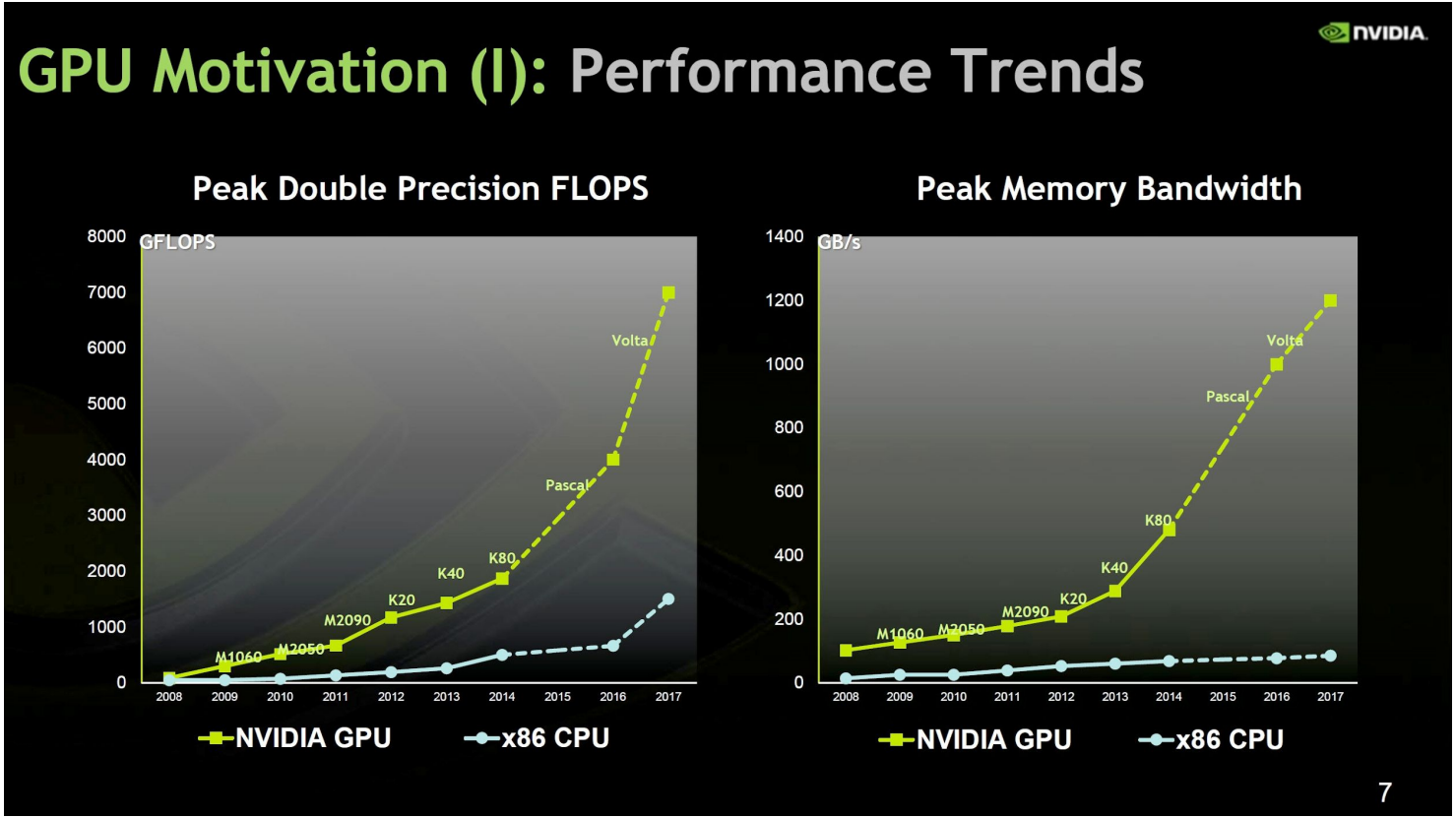
} Outclassed by SVM

Winter #2



Graph generated using: https://books.google.com/ngrams

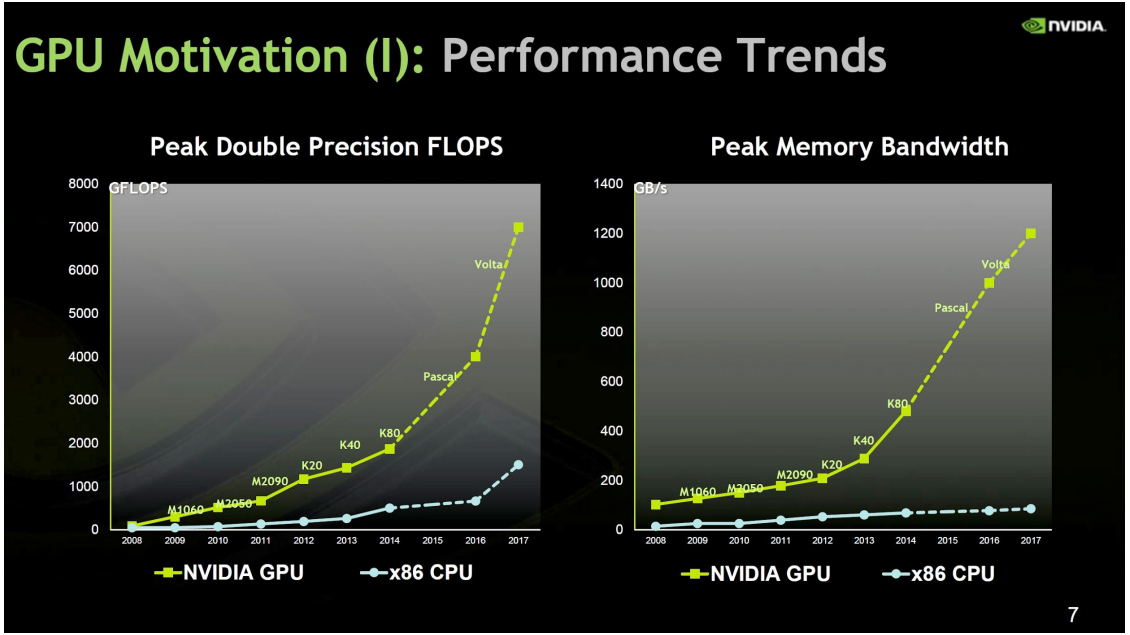
Part of the solution: the GPU



Picture taken from: https://articles2.marketrealist.com/2017/02/whats-nvidias-gpu-strategy-for-2017-and-2018

Part of the solution: the GPU

1. Much higher bandwidth than CPUs.
2. Better parallelization.
3. More register memory.

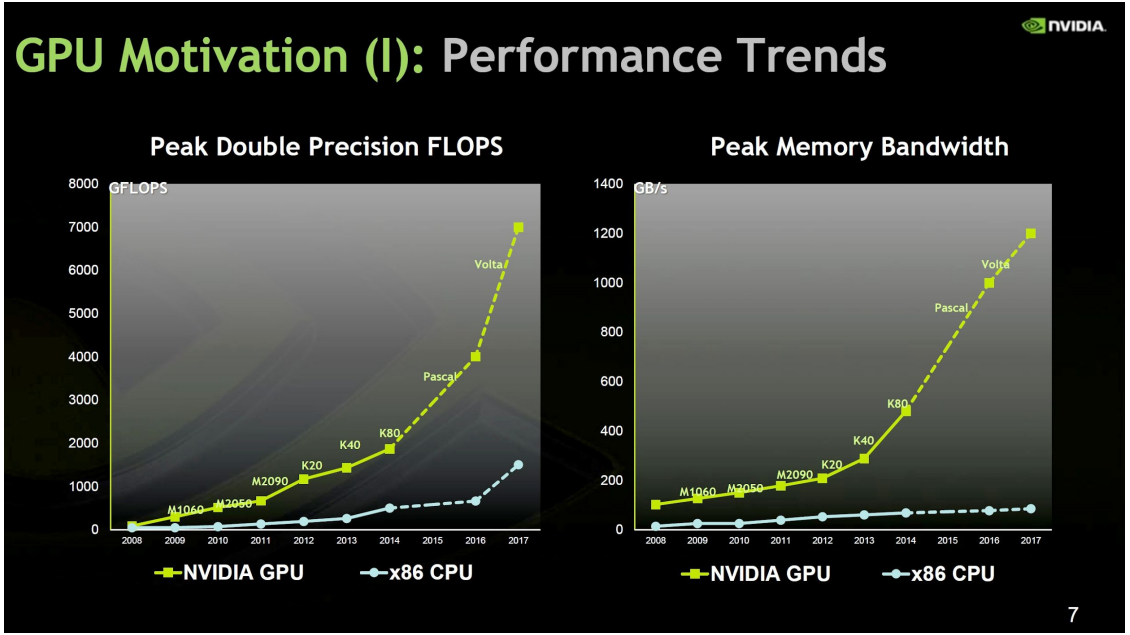


Picture taken from: https://articles2.marketrealist.com/2017/02/whats-nvidias-gpu-strategy-for-2017-and-2018

Part of the solution: the GPU

1. Much higher bandwidth than CPUs.
2. Better parallelization.
3. More register memory.

Why does this matter?



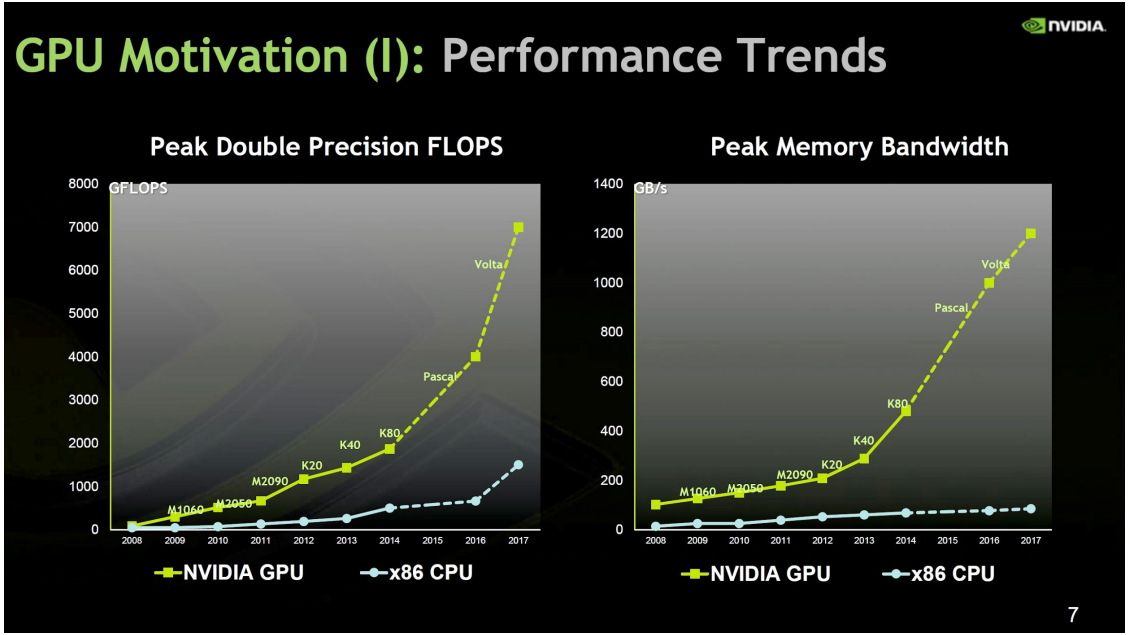
Picture taken from: https://articles2.marketrealist.com/2017/02/whats-nvidias-gpu-strategy-for-2017-and-2018

Part of the solution: the GPU

1. Much higher bandwidth than CPUs.
2. Better parallelization.
3. More register memory.

Why does this matter?

Faster matrix multiplication!



Picture taken from: https://articles2.marketrealist.com/2017/02/whats-nvidias-gpu-strategy-for-2017-and-2018

GPU vs. CPU Poll

ResNet-50 is a popular neural network used to classify objects. How long do you think it takes to perform a classification on the GPU vs CPU?

GPU: GTX 1080Ti
CPU: Dual Xeon E5-2630 v3

A. GPU: 34 ms, CPU: 247ms
B. GPU: 34 ms, CPU: 2477ms
C. GPU: 341 ms, CPU: 247ms
D. GPU: 341 ms, CPU: 2477ms

Source: https://github.com/jcjohnson/cnn-benchmarks

GPU vs CPU

A

B

C

D

As of 2007, there was one huge problem with neural nets.

1. ~~They couldn't solve any problem that wasn't linearly separable.~~
    a. Solved by backpropagation and depth.
2. Backpropagation takes forever to converge!
    a. ~~Not enough compute power to run the model~~
        i. Solved by GPU
    b. Not enough labeled data to train the neural net

Lots of Data!

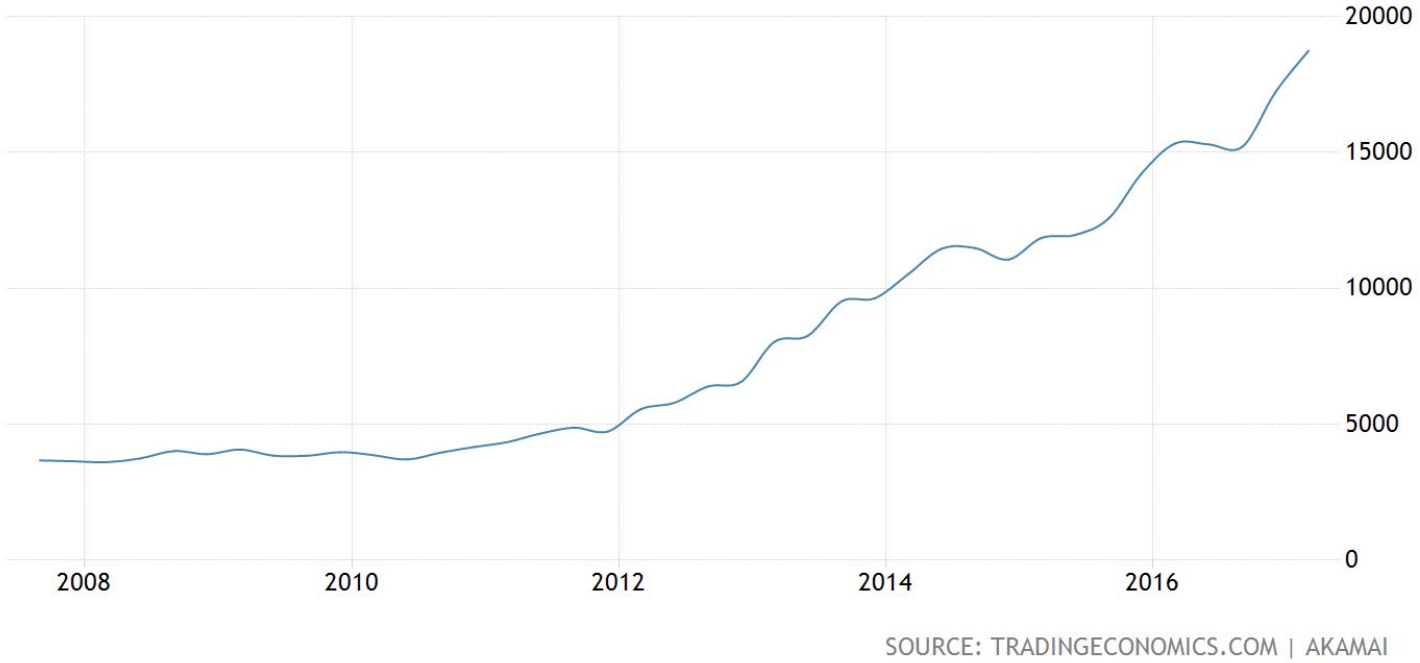
- 2005-2012: Pascal Visual Object Classes
  - 20 classes, 27.5k annotations (in most recent)
- 2010: ImageNet
  - 27 categories, 21.3k subcategories, ~1M images with annotations!
- 2014-2017 COCO
  - 80 classes, ~250k images with bounding boxes and segmentations
- 2017-2019 OpenImages
  - 9M images, 16M bounding boxes, 600 classes, 2.8M segmentation masks

… And lots of data augmentation! (e.g. rotation, crop, add noise, etc)

Source: Taken directly from dataset websites

Who cares if we can’t download it?

Who cares if we can't download it?



Average household download speed (Kb/s)

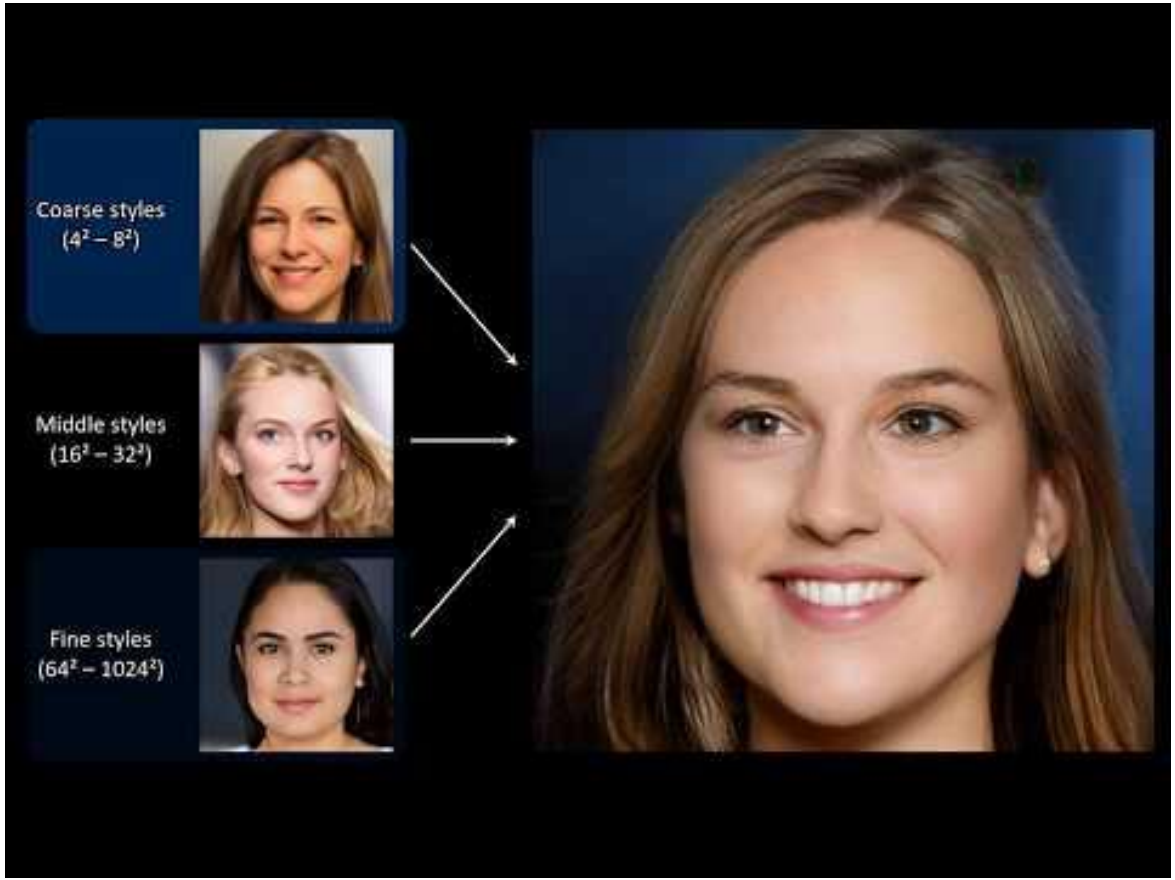
The 2010s were the decade of domain applications

1. ~~They couldn't solve any problem that wasn't linearly separable.~~
2. ~~Backpropagation takes forever to converge!~~
3. Variable-length problems cause gradient problems!
4. Data is rarely labeled!
5. Neural nets are uninterpretable!

The 2010s are the decade of domain applications

1. ~~They couldn't solve any problem that wasn't linearly separable.~~
2. ~~Backpropagation takes forever to converge!~~
3. Variable-length problems cause gradient problems!
    a. Solved by the forget-gate.
4. Data is rarely labeled!
    a. Addressed by emerging unsupervised techniques.
5. Neural nets don’t use information well
    a. Attention allows the focus on a small number of informative items.
    b. Encoders create a more compact, semantic representation (VAE)

Computer Vision, Style Transfer GANs (2019)



Source: https://www.youtube.com/watch?v=kSLJriaOumA

Natural Language Processing/CV - Image Captioning

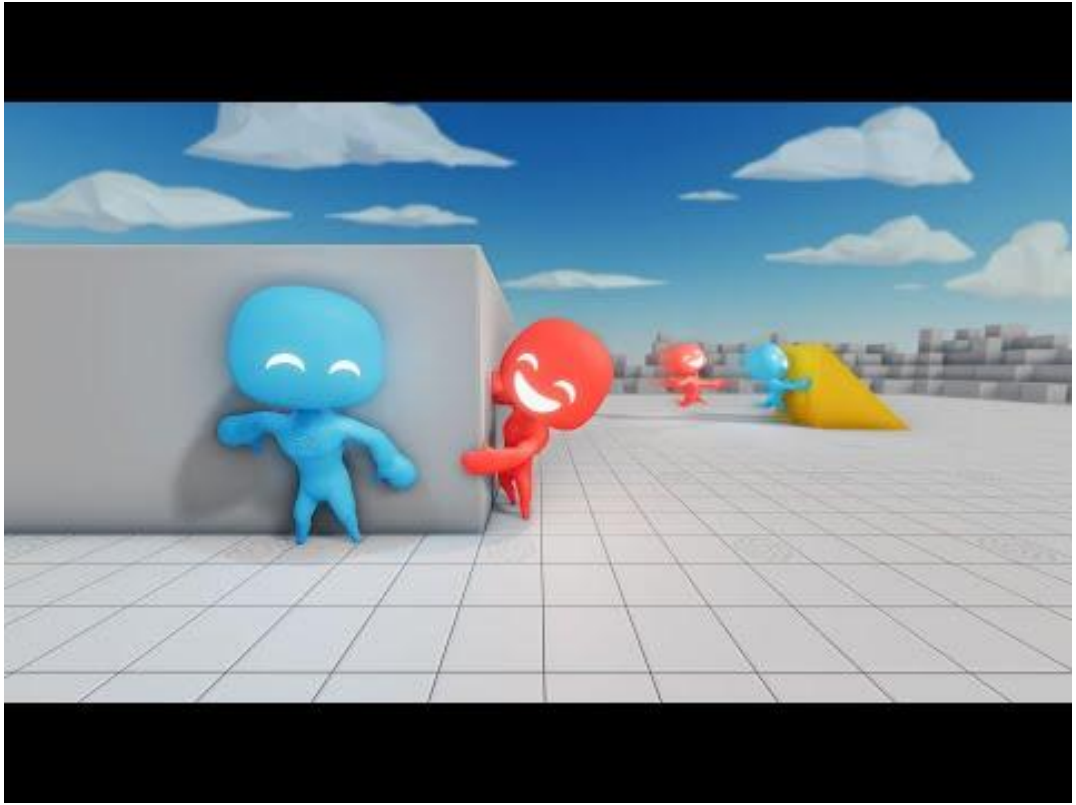
"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

Pictures taken from: https://velement.io/state-of-art-image-captioning/

Deep Reinforcement Learning - Hide and Seek



Source: https://www.youtube.com/watch?v=kopoLzvh5jY

New decade, old problems

1. Extrapolates poorly when the dataset is too specialized
2. Can't transfer between domains easily
3. Can't be audited easily
4. Still too data-hungry
5. No causal understanding
6. Is about the past, not the future
7. And many, many more.

"There is almost as much BS being written about a purported impending AI winter as there is around a purported impending AGI explosion." -- Yann Lecun, FAIR

Course Logistics

Grading

- Homework Assignments (6) 40%
- Final Project 40%
- Exam (1) 15%
- Lecture Attendance / Participation 5%

Late submissions and collaborations

- We have a late submission policy. Read it. Don't be late. Your grade will suffer.
- Do not work together unless it is officially a group project. Read the policy!

Late Policy / Collab Policy: https://www.seas.upenn.edu/~cis522/syllabus.html

Looking forward

- On Tuesday: **Introduction to PyTorch**
- HW 0: due on 1/21 (worth 50% of normal credits)
  - We know this is soon! Don't worry, this assignment is just to set up the infrastructure, should be quick!
- HW 1: due on 1/30
  - Introduction to deep learning / review of classical machine learning techniques. Start early!