STAT 453: Introduction to Deep Learning and Generative Models

Ben Lengerich

Lecture 02: A Brief History of DL

September 8, 2025

A few notes on course logistics

- Extra credit: lecture notes
  - **Sign-up sheet**
  - Check that you are correctly signed up – any issues, please reach out.
- Project
  - Honors-optional component: please reach out.
- HW1
  - Posted on website
  - Due next Friday via Canvas
  - Assignment: Implement a Perceptron in Python
    - We'll discuss this in detail next Monday

Questions about Course Logistics?

What is Machine Learning?

Formally, a computer program is said to **learn** from experience $\mathcal{E}$ with respect to some task $\mathcal{T}$ and performance measure $\mathcal{P}$ if its **performance at $\mathcal{T}$ as measured by $\mathcal{P}$ improves with** $\mathcal{E}$.

Supervised Learning	› Labeled data › Direct feedback › Predict outcome/future	• Task $\mathcal{T}$: Learn a function $h: \mathcal{X} \rightarrow \mathcal{Y}$ • Experience $\mathcal{E}$: Labeled samples $\{(x_i, y_i)\}_{i=1}^{n}$ • Performance $\mathcal{P}$: A measure of how good h is
Unsupervised Learning	› No labels/targets › No feedback › Find hidden structure in data	• Task $\mathcal{T}$: Discover structure in data • Experience $\mathcal{E}$: Unlabeled samples $\{x_i\}_{i=1}^{n}$ • Performance $\mathcal{P}$: Measure of fit or utility
Reinforcement Learning	› Decision process › Reward system › Learn series of actions	• Task $\mathcal{T}$: Learn a policy $\pi: S \rightarrow A$ • Experience $\mathcal{E}$: Interaction with environment • Performance $\mathcal{P}$: Expected reward

Source: Raschka and Mirjalily (2019). *Python Machine Learning, 3rd Edition*

Data Representation

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Feature vector

Data Representation

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Feature vector

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix}$$

Feature Matrix / Design Matrix

$$\mathbf{X} = \begin{bmatrix} x_1^{[1]} & x_2^{[1]} & \cdots & x_m^{[1]} \\ x_1^{[2]} & x_2^{[2]} & \cdots & x_m^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{[n]} & x_2^{[n]} & \cdots & x_m^{[n]} \end{bmatrix}$$

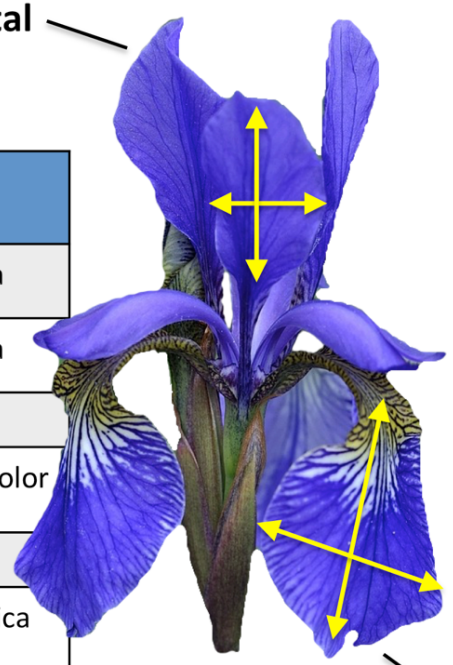
Feature Matrix / Design Matrix

Data Representation (structured data)

$m =$ 4

$n =$ 150

	Sepal length	Sepal width	Petal length	Petal width	
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica



Label

Data Representation (unstructured data; images)

Convolutional Neural Networks

```
Image batch dimensions: torch.Size([128, 1, 28, 28])
Image label dimensions: torch.Size([128])
```

← "NCHW" representation (more on that later)

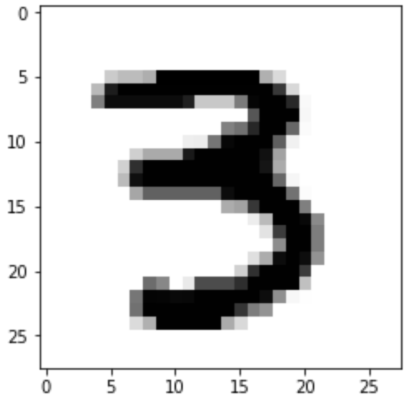
```
print(images[0].size())

  torch.Size([1, 28, 28])

images[0]

tensor([[[0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],

         [0.0000, 0.0000, 0.0000, 0.0000, 0.5020, 0.9529, 0.9529, 0.9529,
          0.9529, 0.9529, 0.9529, 0.8706, 0.2157, 0.2157, 0.2157, 0.5176,
          0.9804, 0.9922, 0.9922, 0.8392, 0.0235, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.6627, 0.9922, 0.9922, 0.9922, 0.0314, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.4980, 0.5529,
          0.8471, 0.9922, 0.9922, 0.5961, 0.0157, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0667, 0.0745, 0.5412, 0.9725, 0.9922,
```



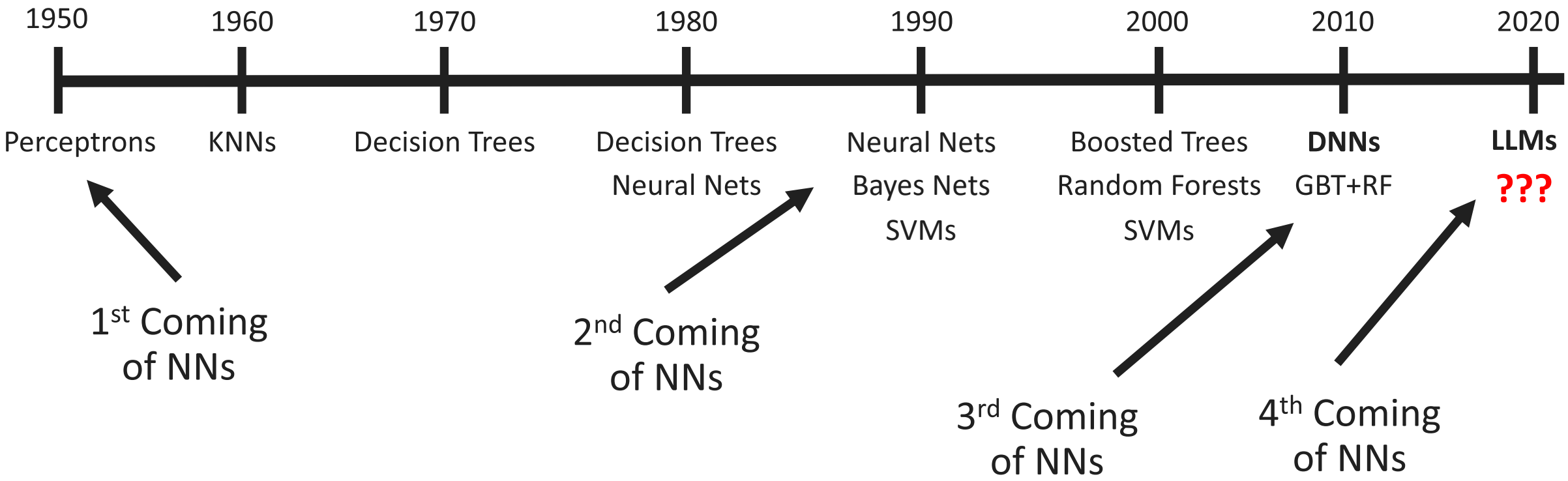
Machine Learning Jargon

- **Training a model** = fitting a model = parameterizing a model = learning from data
- **Training example**, synonymous to training record, training instance, training sample (in some contexts, sample refers to a collection of training examples)
- **Feature**, synonymous to observation, predictor, variable, independent variable, input, attribute, covariate
- **Target**, synonymous to outcome, ground truth, output, response variable, dependent variable, (class) label (in classification)
- **Output / Prediction**, use this to distinguish from targets; here, means output from the model

Today: A Brief History of DL

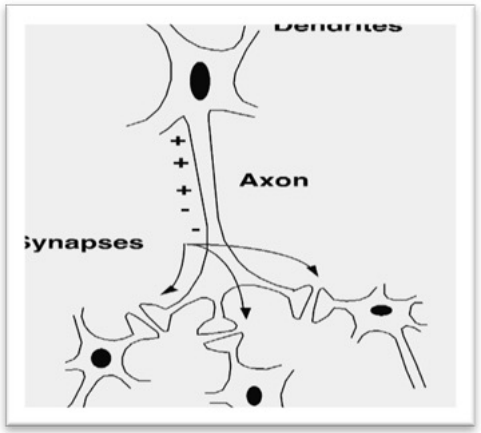
1. **Artificial neurons**
2. Multilayer neural networks
3. Deep Learning
4. The DL Hardware & Software Landscape
5. Current Research Trends

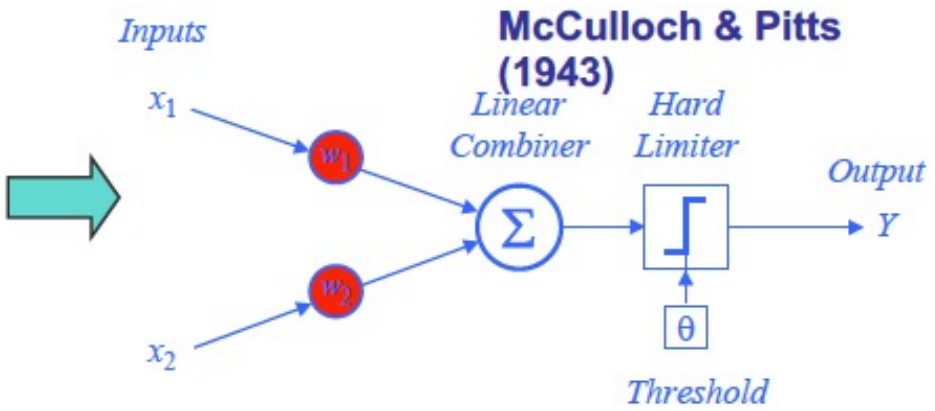
A Brief History of ML

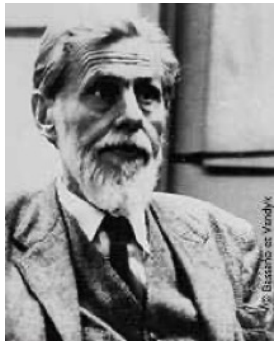


Courtesy of Rich Caruana

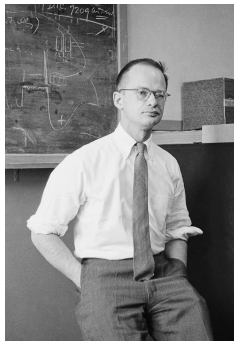
McCulloch & Pitt's neuron model (1943)







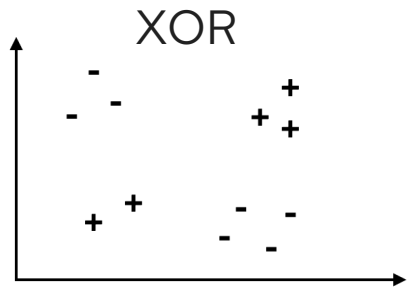
Warren McCulloch

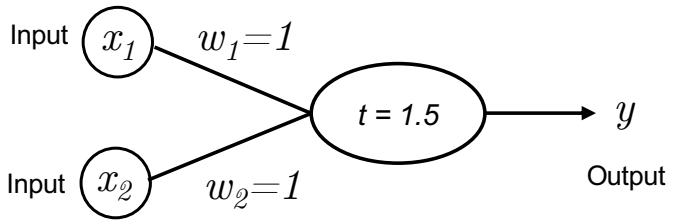


Walter Pitts

From biological neuron to artificial neuron

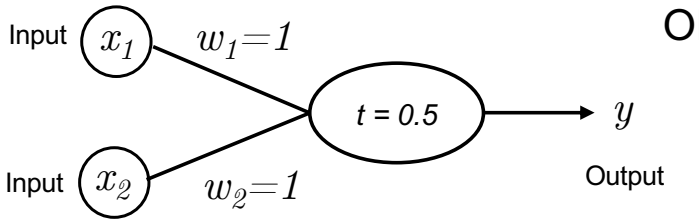
- McCulloch & Pitts neuron: Threshold and (+1, -1) weights
- Can represent "AND", "OR", "NOT"
- But not "XOR"





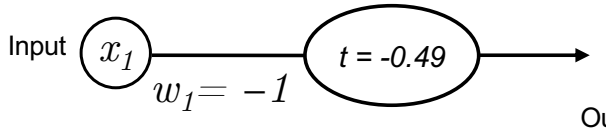
AND

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



OR

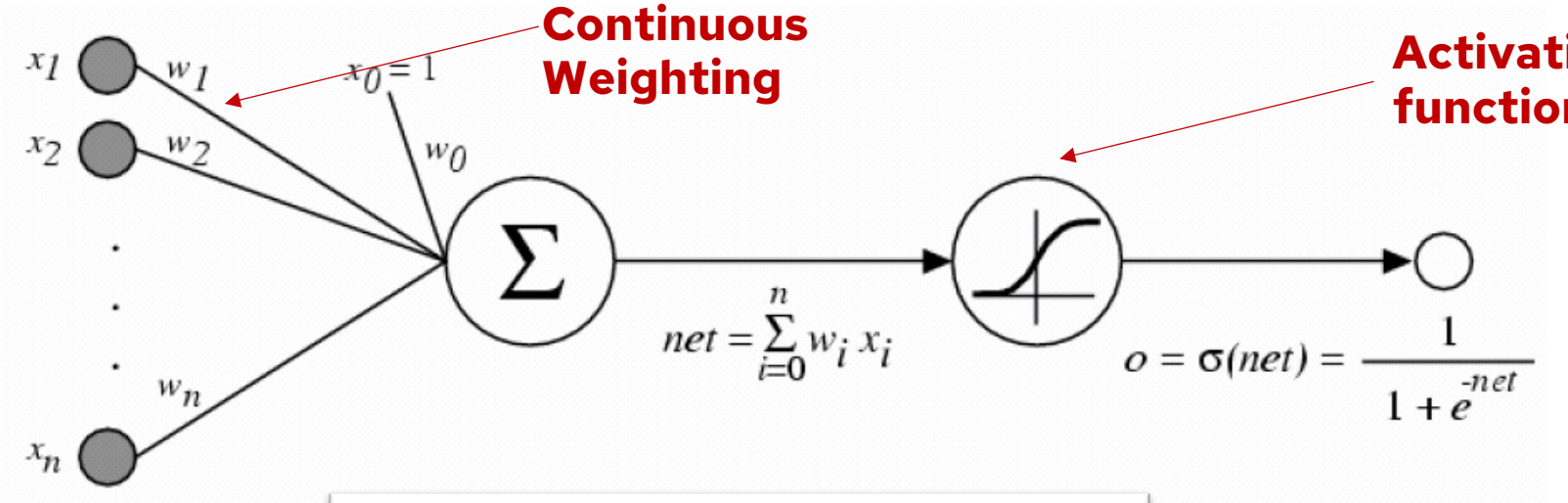
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

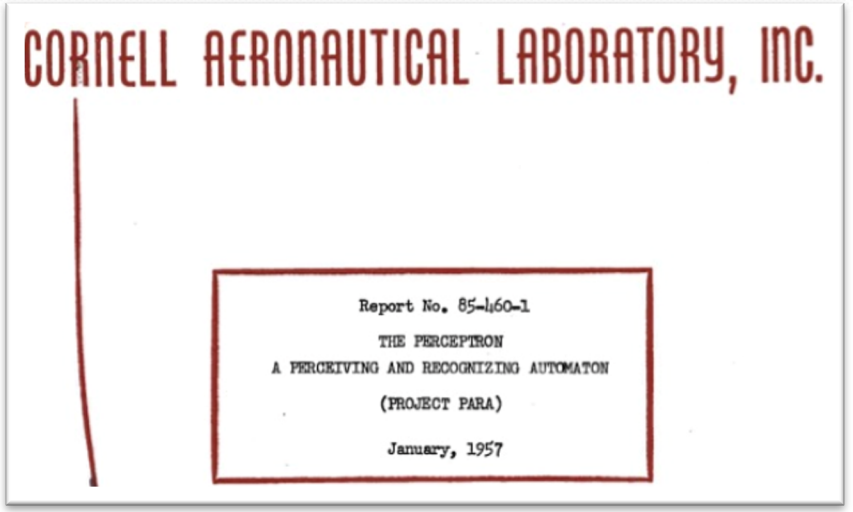


NOT

x_1	y
0	1
1	0

Perceptrons generalize MP neurons




CORNELL AERONAUTICAL LABORATORY, INC.

Report No. 85-460-1

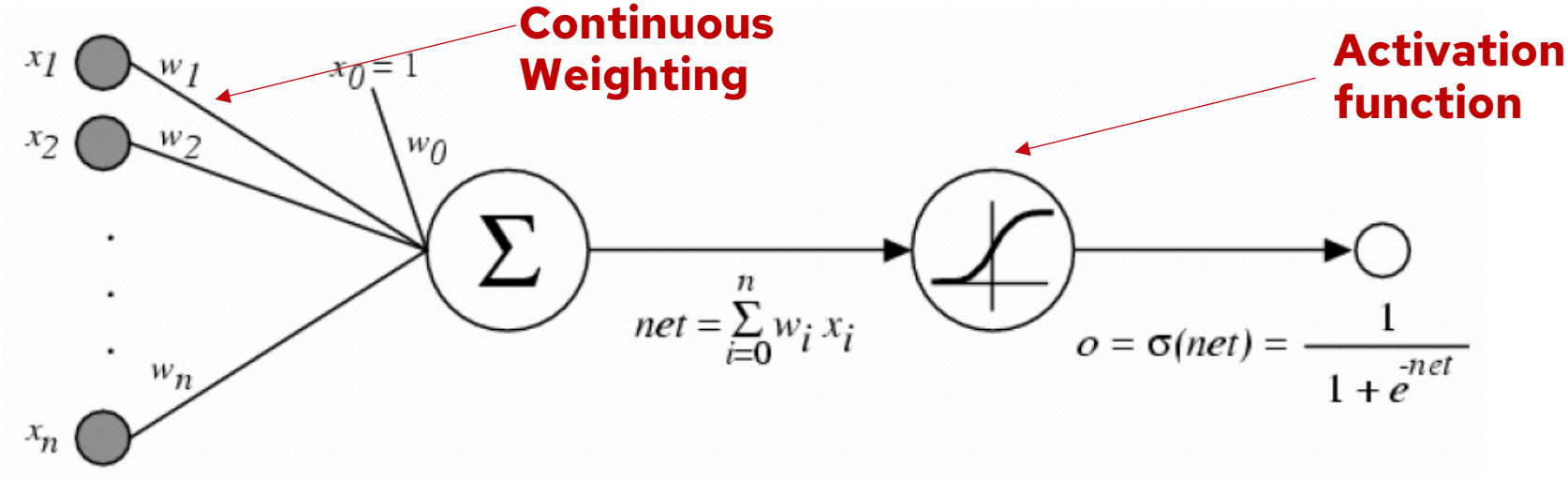
THE PERCEPTRON

A PERCEIVING AND RECOGNIZING AUTOMATON

(PROJECT PARA)

January, 1957

Perceptrons generalize MP neurons



- Consider regression problem f: X→Y for scalar Y
  - Let $Y \sim N(f(x), \Sigma^2)$
  - Then $\text{argmax}_w \log \prod_i P(y_i \mid x_i; w) = \text{argmin}_w \sum_i \frac{1}{2}\left(y_i - f(x_i; w)\right)^2$
    - We can find $\text{argmax}_w \log \prod_i P(y_i \mid x_i; w)$ by iterating:

$$w_d = w_d + \eta \sum_i (y_i - o_i)\, o_i (1 - o_i) x_d^i$$

Bonus

Deriving the perceptron learning algorithm

- Recall the nice property of sigmoid: $\frac{d\sigma}{dt} = \sigma(1-\sigma)$

$$
\begin{aligned}
\frac{\partial E_D[\vec{w}])}{\partial w_j} &= \frac{\partial}{\partial w_i}\frac{1}{2}\sum_d (t_d - o_d)^2 \\
&= \frac{1}{2}\sum_d 2(t_d - o_d)\frac{\partial}{\partial w_i}(t_d - o_d) \\
&= \sum_d (t_d - o_d)\left(-\frac{\partial o_d}{\partial w_i}\right) \\
&= -\sum_d (t_d - o_d)\frac{\partial o_d}{\partial net_i}\frac{\partial net_d}{\partial w_i} \\
&= -\sum_d (t_d - o_d) o_d(1-o_d) x_d^i
\end{aligned}
$$

x_d = input

t_d = target output

o_d = observed output

w_i = weight i

Batch mode:

Do until converge:

1. compute gradient $\nabla E_D[w]$
2. $\vec{w} = \vec{w} - \eta \nabla E_D[\vec{w}]$

Incremental mode:

Do until converge:

- For each training example *d* in *D*
  1. compute gradient $\nabla E_d[w]$
  2. $\vec{w} = \vec{w} - \eta \nabla E_d[\vec{w}]$

where

$\nabla E_d[\vec{w}] = -(t_d - o_d) o_d (1 - o_d) \vec{x}_d$

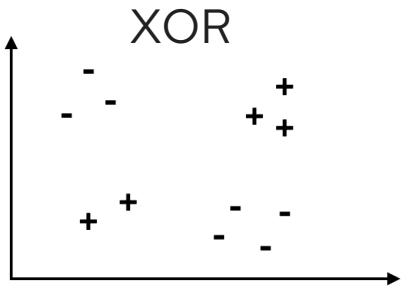
Optimism vs. pessimism

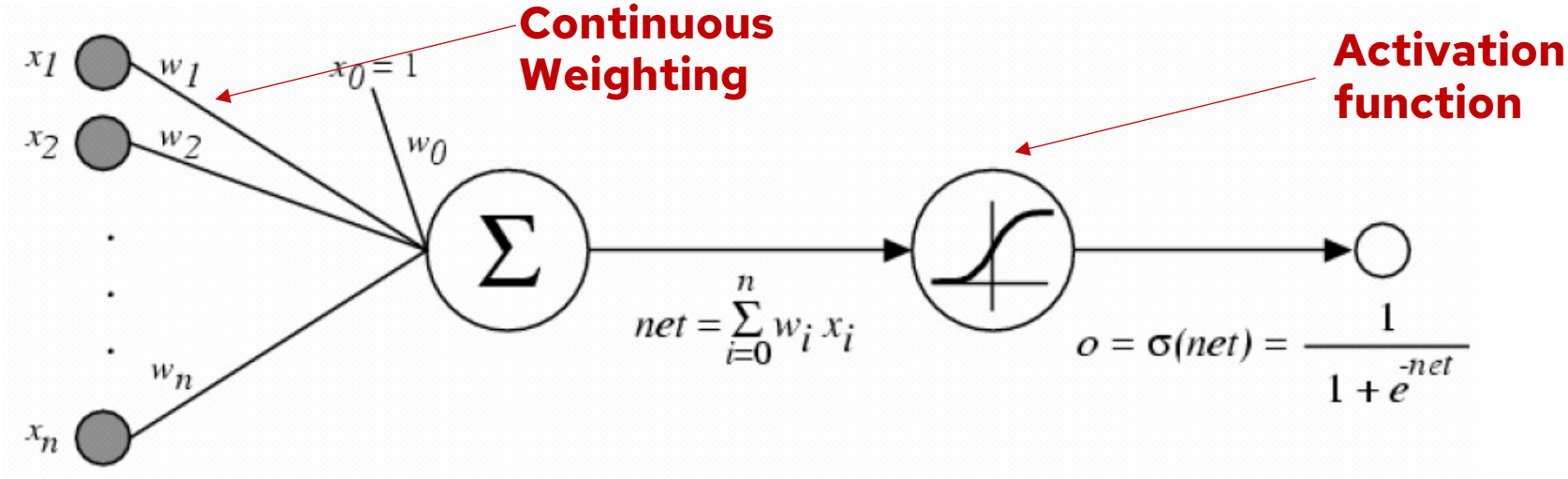
- Optimism in the early history of AI (from wiki)

> In a 1958 press conference organized by the US Navy, Rosenblatt made statements about the perceptron that caused a heated controversy among the fledgling AI community; based on Rosenblatt's statements, *The New York Times* reported the perceptron to be "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."[5]

- Discussion: what do you think about the current AI gold rush?

Can a Perceptron represent XOR?

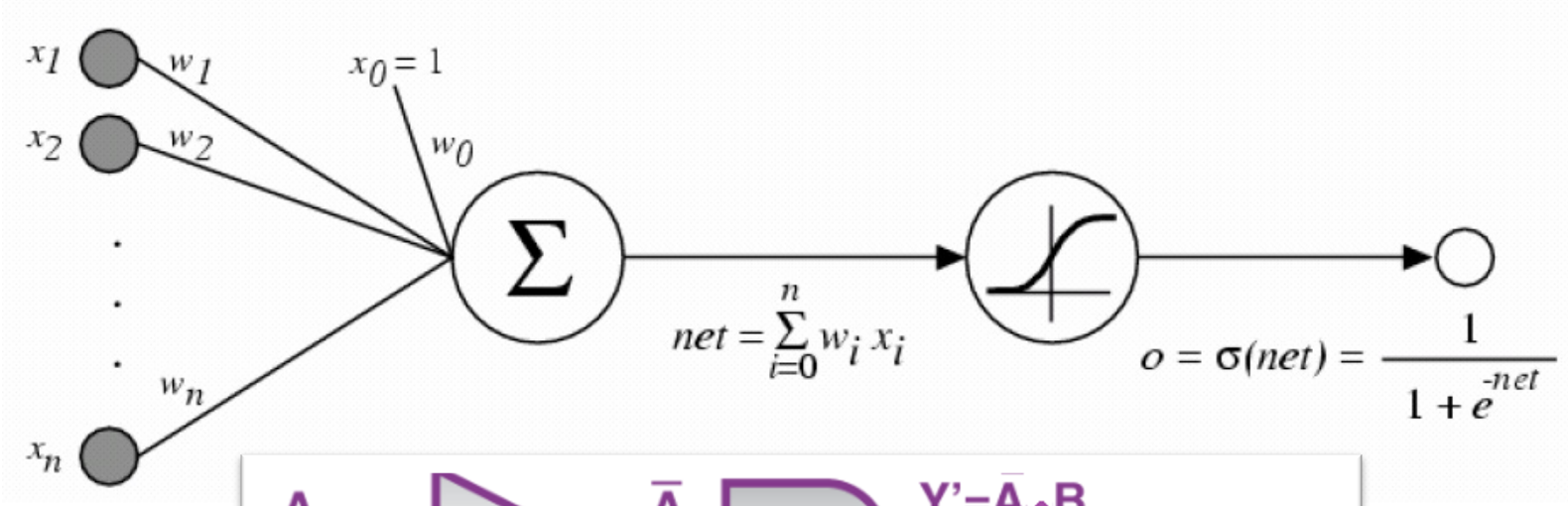




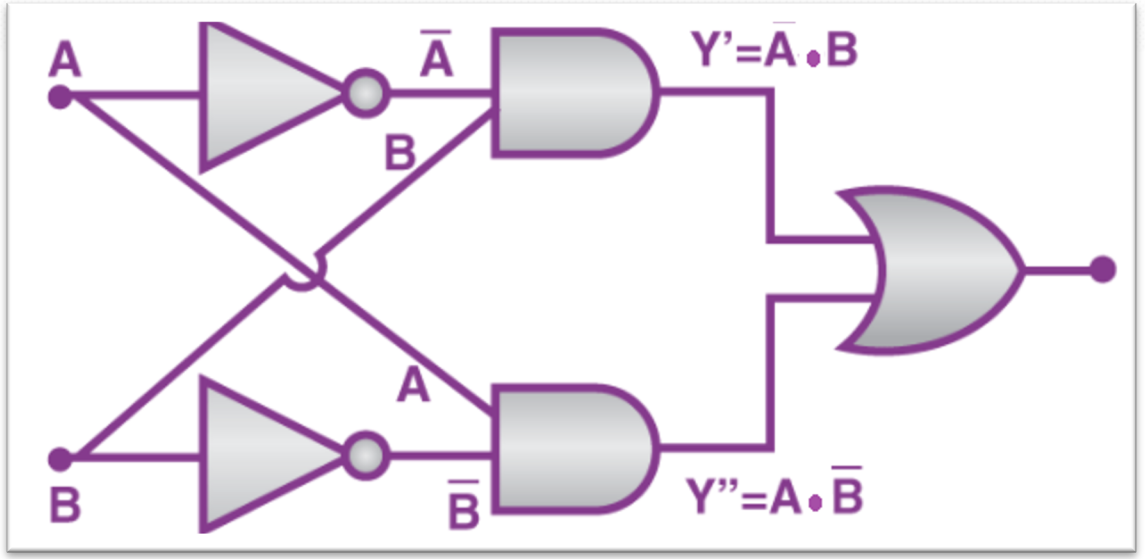
- No
- If there were, then there would be constants w_1 and w_2 such that:
  - When $x_1 = x_2$, then $\sigma(w_1 x_1 + w_2 x_2) < \theta$
  - When $x_1 \neq x_2$, then $\sigma(w_1 x_1 + w_2 x_2) \geq \theta$
  - Let $x_1 = 1, x_2 = 0$
    - Eq. (1): $\sigma(w_1) \geq \theta$
  - Let $x_1 = 0, x_2 = 1$
    - Eq. (2): $\sigma(w_2) \geq \theta$
  - Let $x_1 = 1, x_2 = 1$:
    - Eq. (3): $\sigma(w_1 + w_2) < \theta$

Eq. (1) + Eq. (2) contradicts Eq. (3)

An XOR Logic Gate



Multi-layer Perceptron?



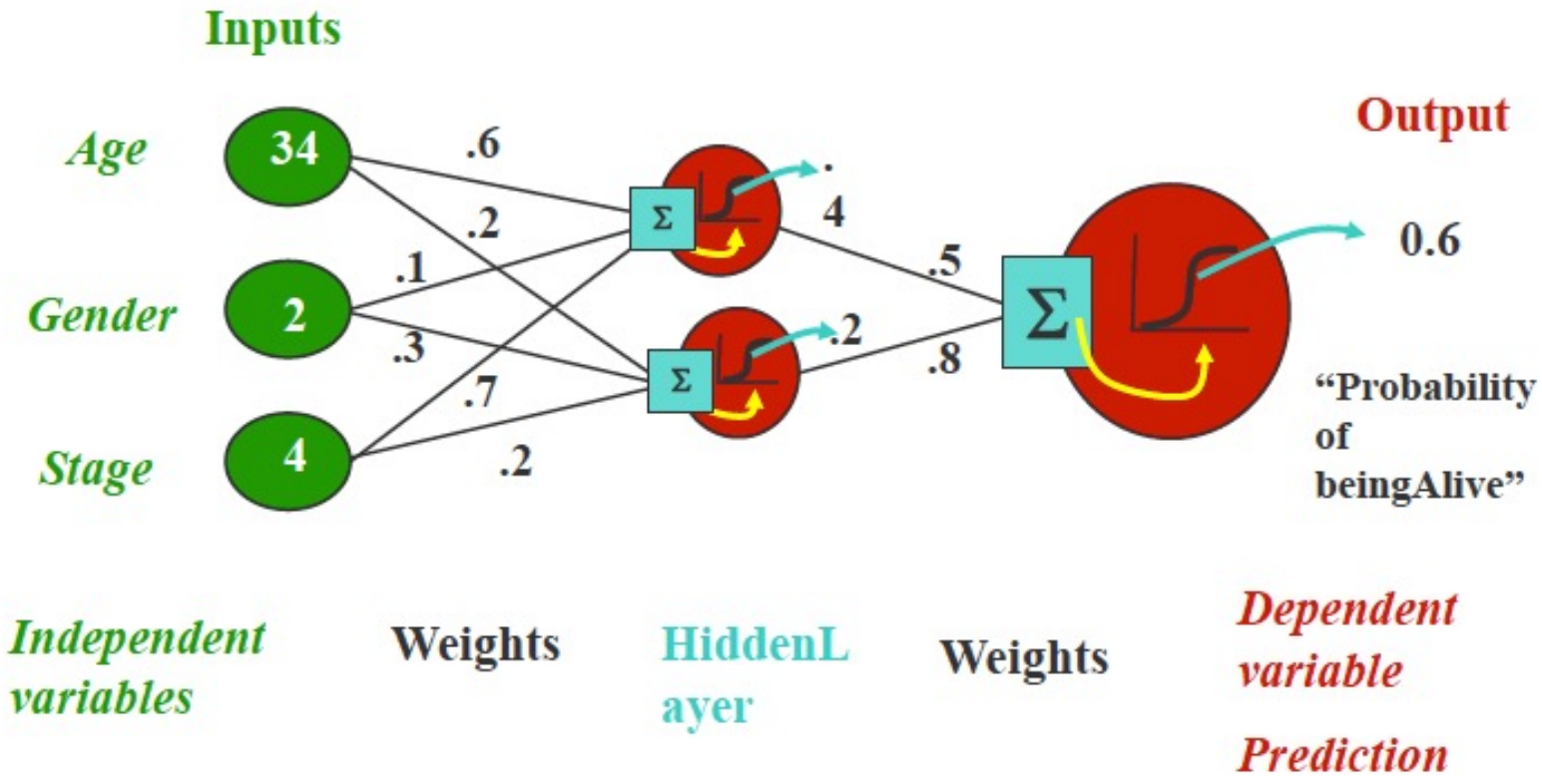
https://byjus.com/jee/basic-logic-gates/

Today: A Brief History of DL

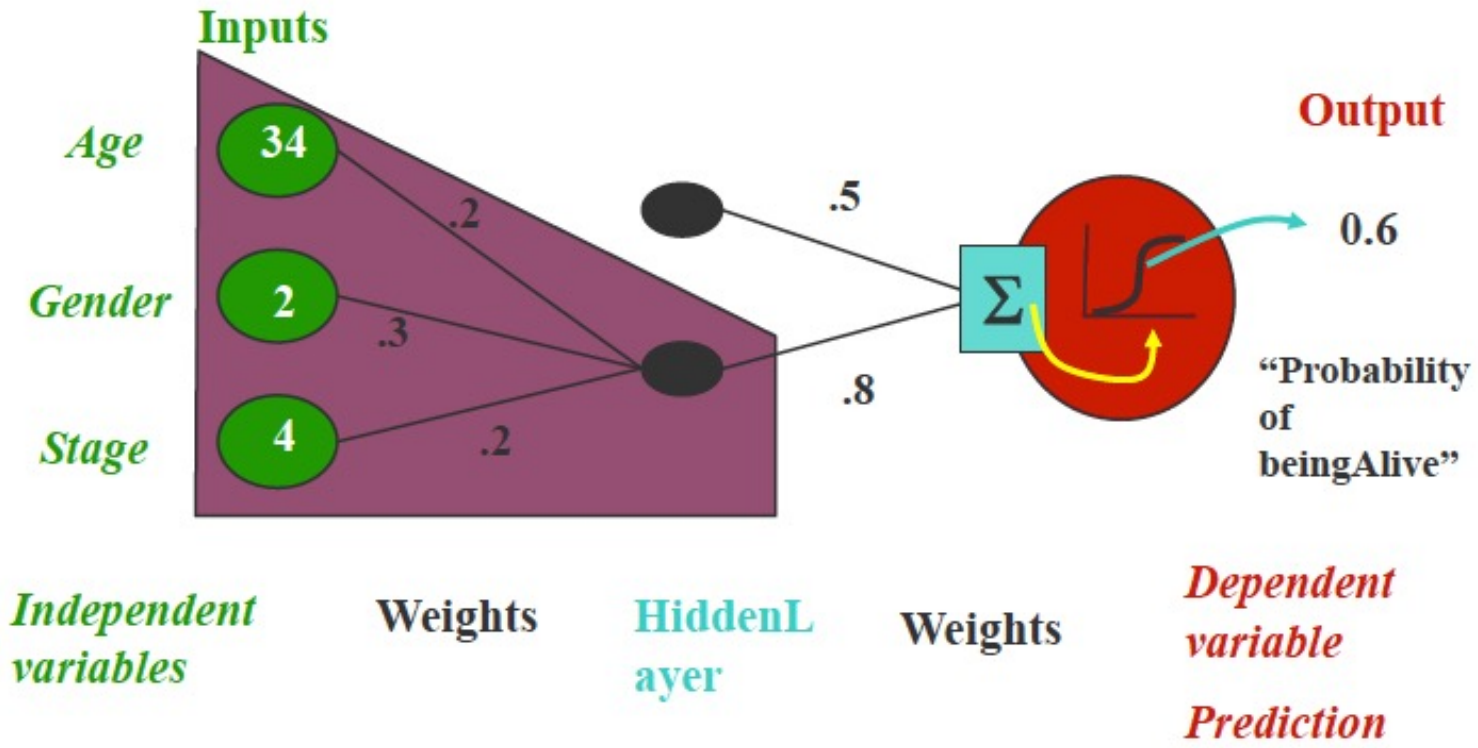
1. Artificial neurons
2. **Multilayer neural networks**
3. Deep Learning
4. The DL Hardware & Software Landscape
5. Current Research Trends

Multi-Layer Perceptrons (MLPs)

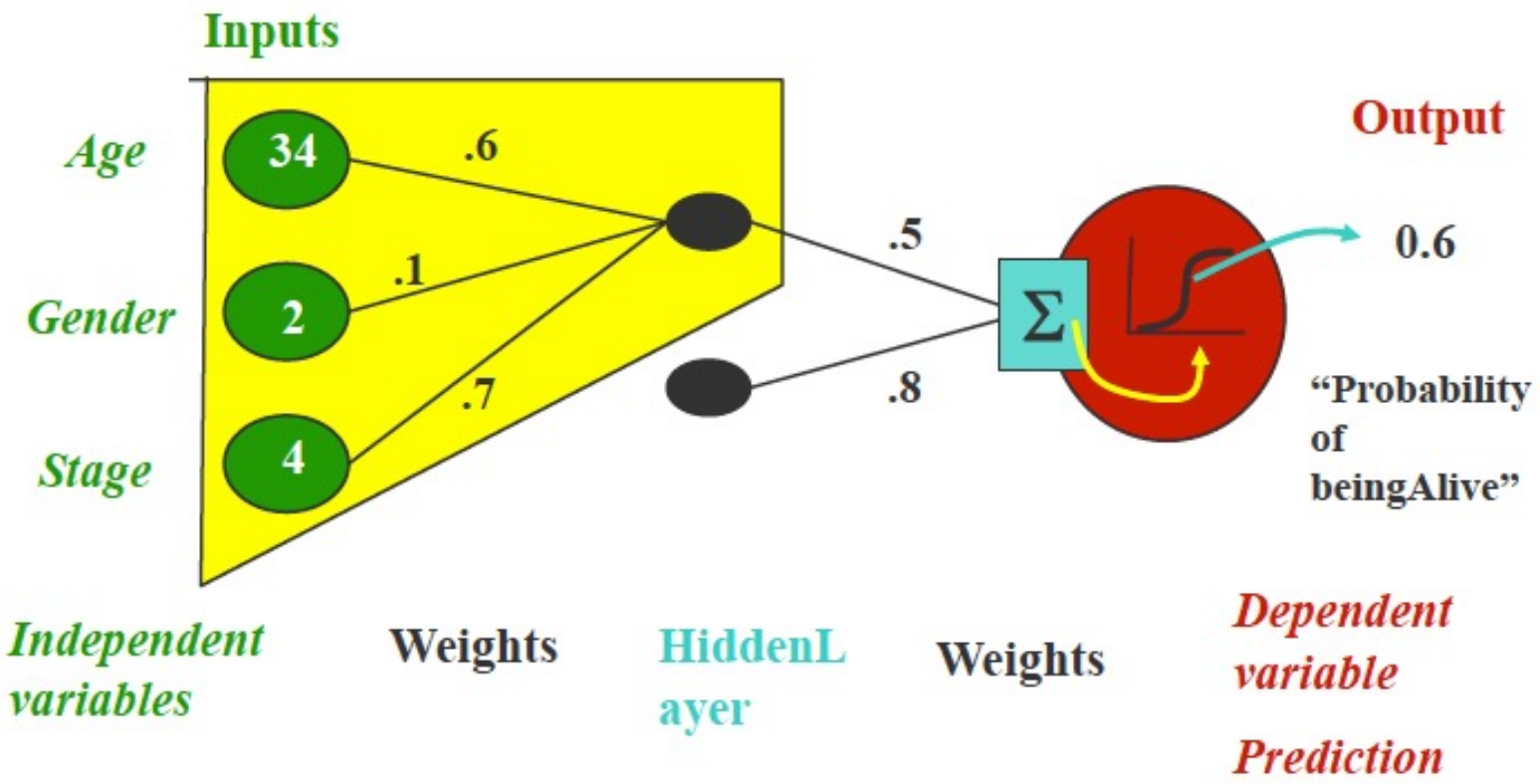
aka Multi-layer Neural Networks



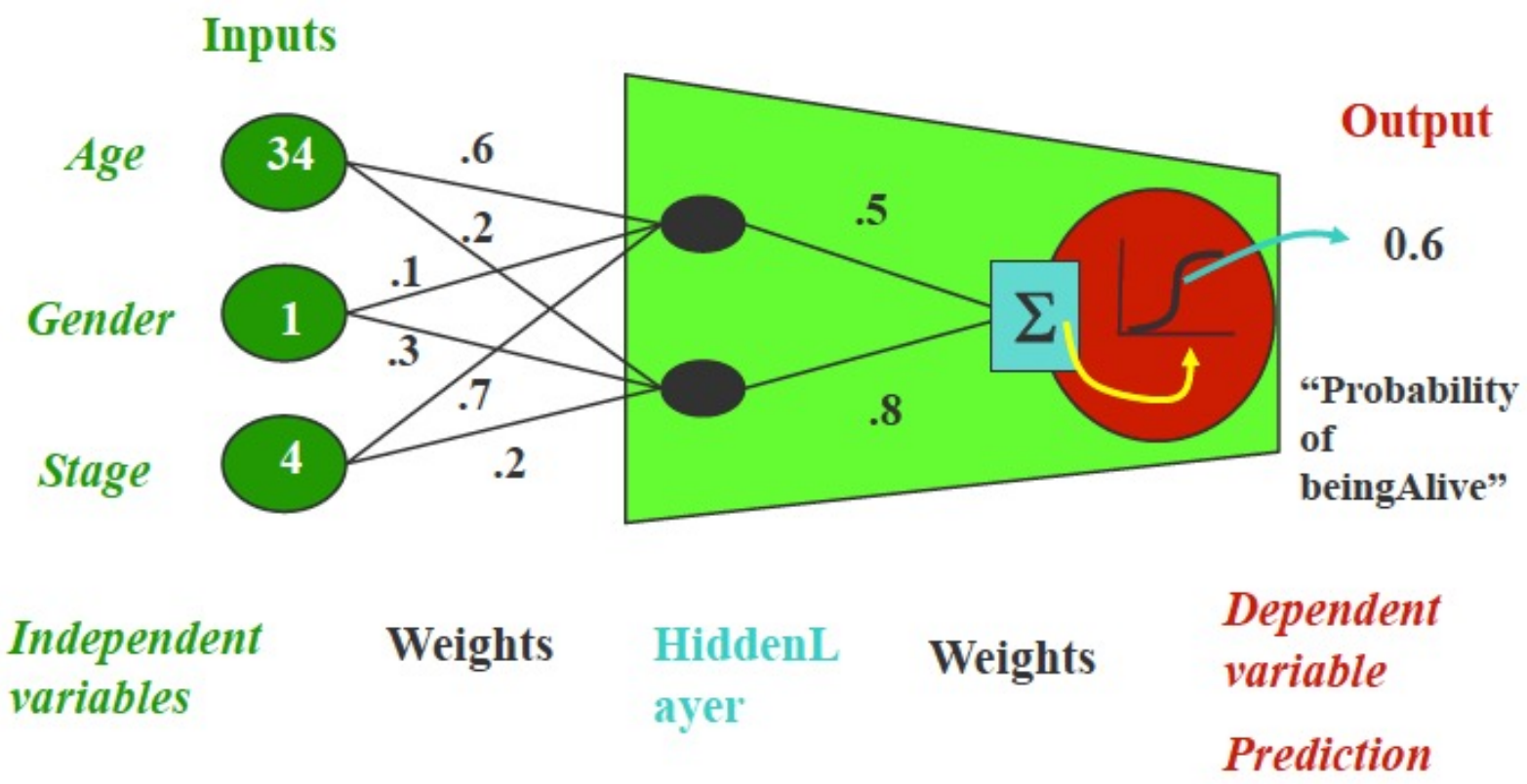
“Combined Logistic Models”



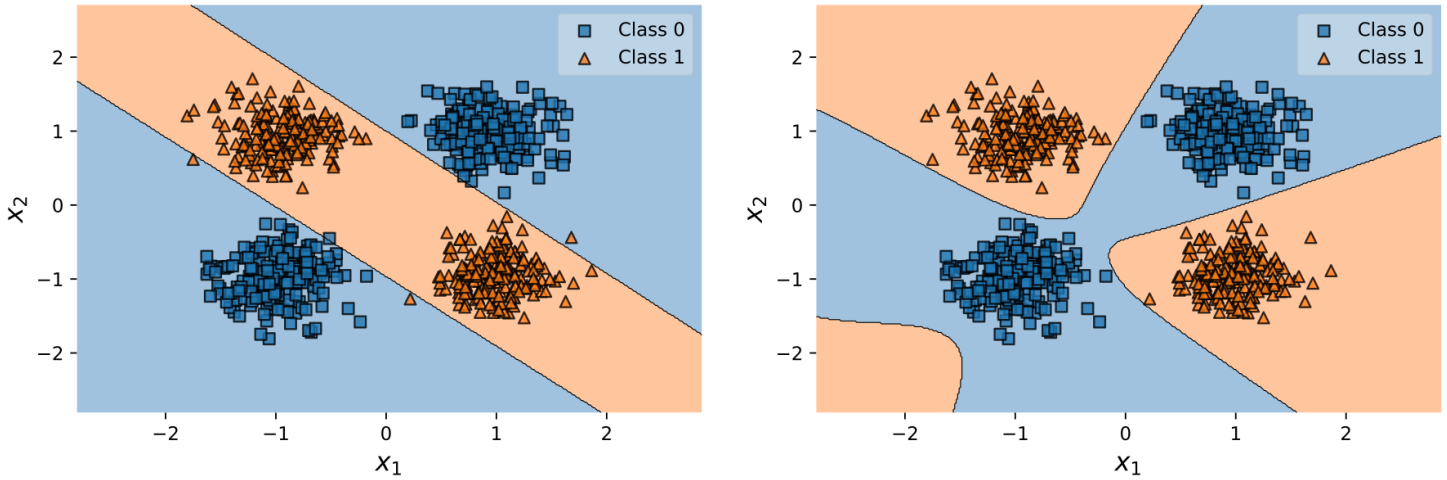
"Combined Logistic Models"



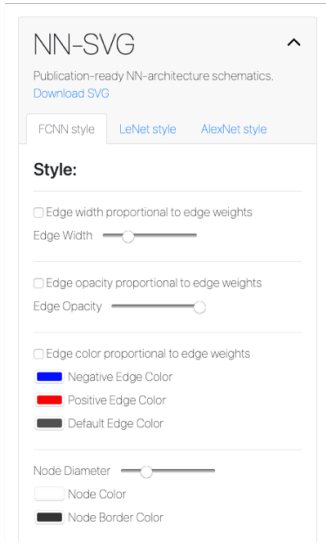
“Combined Logistic Models”

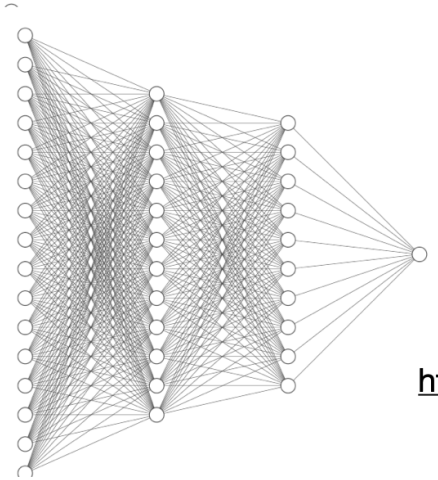


Multilayer Neural Networks Can Solve XOR



Decision boundaries of two different multilayer perceptrons on simulated data solving the XOR problem





https://alexlenail.me/NN-SVG/index.html

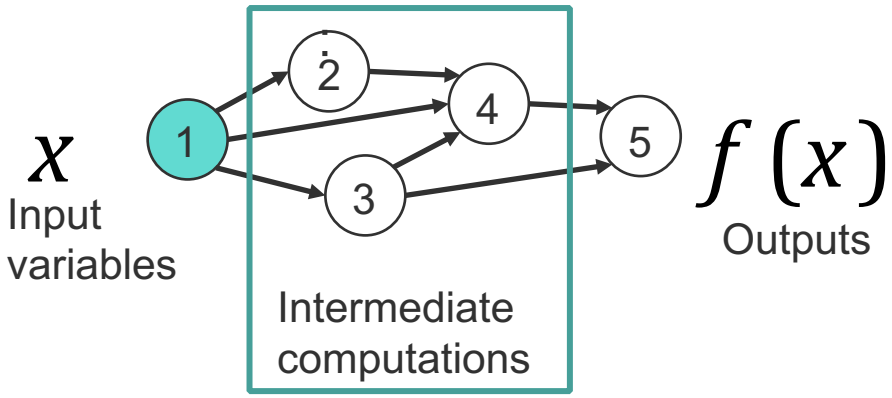
A new problem: Training

- How can we train a multilayer model?
  - No targets / ground truth for the hidden nodes
- Solution: Backpropagation
  - Independently formulated many times
    - http://people.idsia.ch/~juergen/who-invented-backpropagation.html
  - Rumelhart and Hinton (1986) showed that it really works
    - Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning representations by back-propagating errors.* Nature, 323(6088), 533.

> In late 1985, I actually had a deal with Dave Rumelhart that I would write a short paper about backpropagation, which was his idea, and he would write a short paper about autoencoders, which was my idea. It was always better to have someone who didn't come up with the idea write the paper because he could say more clearly what was important.
>
> So I wrote the short paper about backpropagation, which was the *Nature* paper that came out in 1986, but Dave still hasn't written the short paper about autoencoders. I'm still waiting.
>
> What he did do was tell Dave Zipser about the idea of autoencoders and
>
> – Geoffrey Hinton in Talking Nets - An Oral History of Neural Networks, pg. 380

Backpropagation

- Neural networks are function compositions that can be represented as computation graphs:



$$\frac{\partial f_n}{\partial x} =$$

- By applying the chain rule, and working in reverse order, we get:

$$\frac{\partial f_n}{\partial x} = \sum_{i_1 \in \pi(n)} \frac{\partial f_n}{\partial f_{i_1}} \frac{\partial f_{i_1}}{\partial x} = \sum_{i_1 \in \pi(n)} \frac{\partial f_n}{\partial f_{i_1}} \sum_{i_2 \in \pi(i_1)} \frac{\partial f_{i_1}}{\partial f_{i_2}} \frac{\partial f_{i_1}}{\partial x} = \dots$$

Backpropagation: More than just gradient descent?

- Engineering innovations

current gradient and earlier gradients to the weight change.

To break symmetry we start with small random weights. Variants on the learning procedure have been discovered independently by David Parker (personal communication) and by Yann Le Cun[3].

Leads to Dropout?

Backpropagation: More than just gradient descent?

- Engineering innovations

The most obvious drawback of the learning procedure is that the error-surface may contain local minima so that gradient descent is not guaranteed to find a global minimum. However, experience with many tasks shows that the network very rarely gets stuck in poor local minima that are significantly worse than the global minimum. We have only encountered this undesirable behaviour in networks that have just enough connections to perform the task. Adding a few more connections creates extra dimensions in weight-space and these dimensions provide paths around the barriers that create poor local minima in the lower dimensional subspaces.

Leads to Over-parameterized models?

Questions about Multilayer networks?

Today: A Brief History of DL

1. Artificial neurons
2. Multilayer neural networks
3. **Deep Learning**
4. The DL Hardware & Software Landscape
5. Current Research Trends

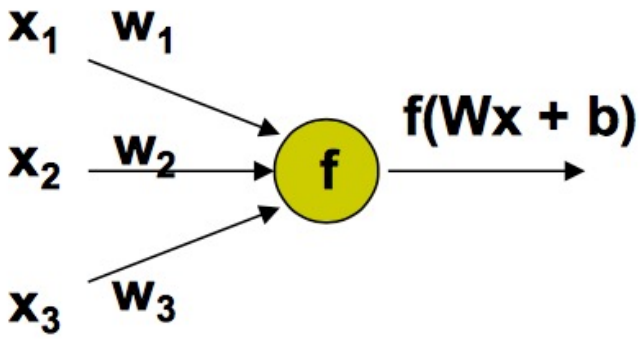
About the term “Deep Learning”

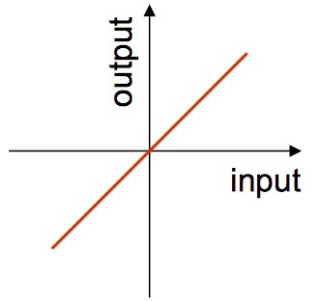
“Representation learning is a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification. Deep learning methods are representation-learning methods with multiple levels of representation [...]”

-- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436.

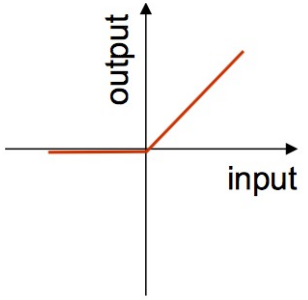
DL building blocks

- Activation Functions
  - Linear and ReLU
  - Sigmoid and tanh
  - etc.

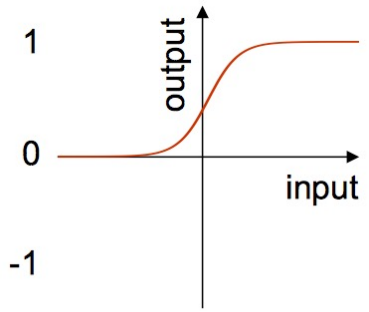




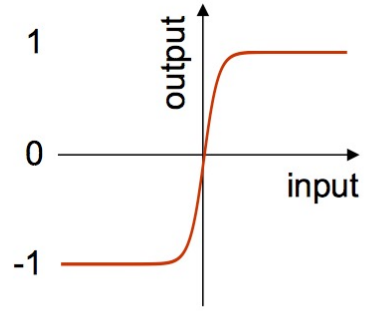
Linear



Rectified linear



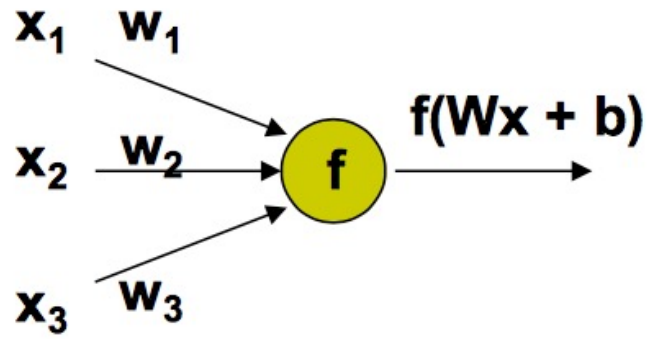
Sigmoid

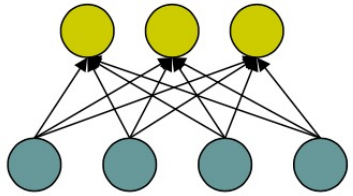


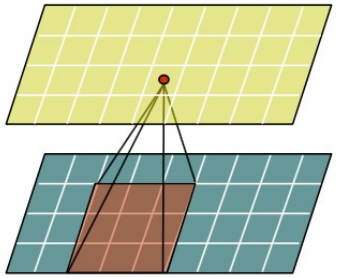
Hyperbolic tangent

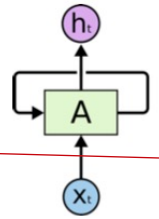
DL building blocks

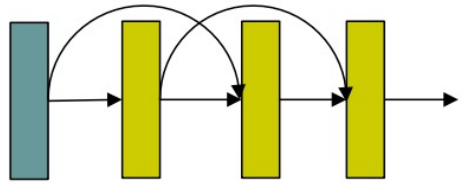
- Activation Functions
  - Linear and ReLU
  - Sigmoid and tanh
  - etc.
- Layers
  - Fully-connected
  - Convolutional & pooling
  - Recurrent
  - Residual (ResNets)
  - Transformers
  - etc.







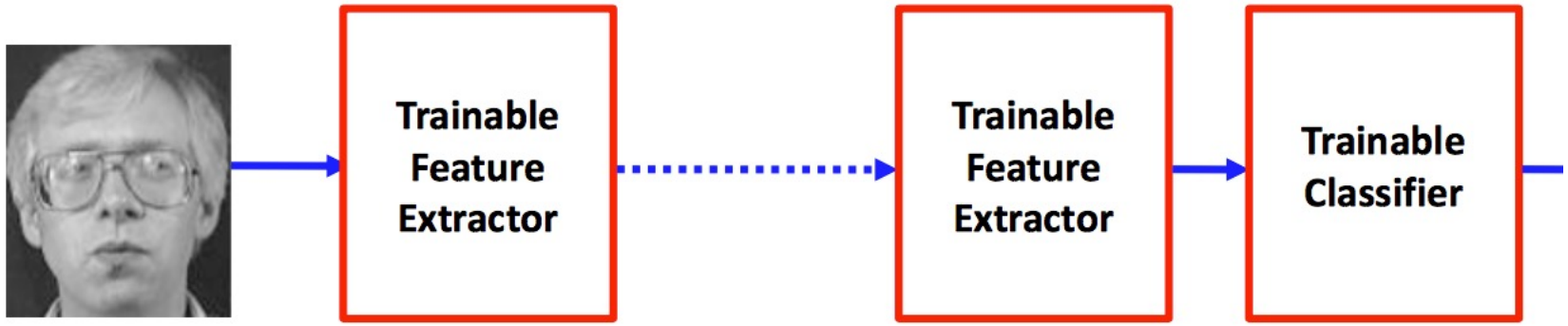




DL building blocks

- Activation Functions
  - Linear and ReLU
  - Sigmoid and tanh
  - etc.
- Layers
  - Fully-connected
  - Convolutional & pooling
  - Recurrent
  - Residual (ResNets)
  - Transformers
  - etc.
- Loss functions
  - Cross-entropy, MSE, etc.

DL learns hierarchical representations



- In Language Models: hierarchy in syntax and semantics
  - Words → Parts of Speech → Sentences → Stories
- Is Vision: hierarchy in composition
  - Pixels → Edges → Textons → Parts → Objects → Scenes

When did DL Become Really Popular?

That was the view of people in computer vision until 2012. Most people in computer vision thought this stuff was crazy, even though Yann LeCun sometimes got systems working better than the best computer vision systems, they still thought this stuff was crazy, it wasn't the right way to do vision. They even rejected papers by Yann, even though they worked better than the best computer vision systems on particular problems, because the referees thought it was the wrong way to do things. That's a lovely example of scientists saying, "We've already decided what the answer has to look like, and anything that doesn't look like the answer we believe in is of no interest."

In the end, science won out, and two of my students won a big public competition, and they won it dramatically. They got almost half the error rate of the best computer vision systems, and they were using mainly techniques developed in Yann LeCun's lab but mixed in with a few of our own techniques as well.

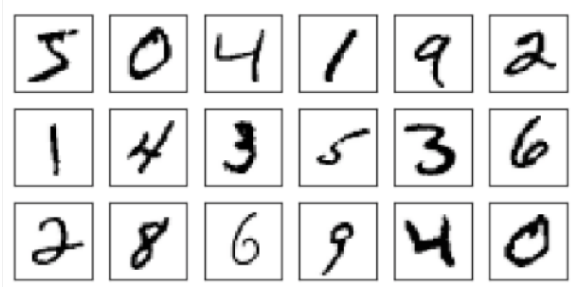
MARTIN FORD: This was the ImageNet competition?

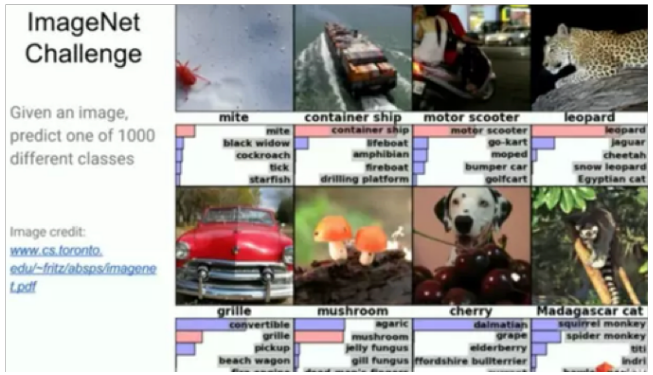
GEOFFREY HINTON: Yes, and what happened then was what should happen in science. One method that people used to think of as complete nonsense had now worked much better than the method they believed in, and within two years, they all switched. So, for things like object classification, nobody would dream of trying to do it without using a neural network now.

(Excerpt from "Architects of Intelligence")

- AlexNet achieved 15.4% error on top-5 in 2012
  - 2nd best was not even close: 26.2%
  - (nowadays ~3% error on ImageNet)

DL: Driven by Benchmark Datasets





MNIST (1998)

- 60,000 examples, 10 classes
- features: 28x28x1
- http://yann.lecun.com/exdb/mnist/

CIFAR-10/CIFAR-100 (2009)

- 60,000 examples, 10 or 100 classes
- features: 32x32x3,
- https://www.cs.toronto.edu/~kriz/cifar.html

ImageNet (~2010)

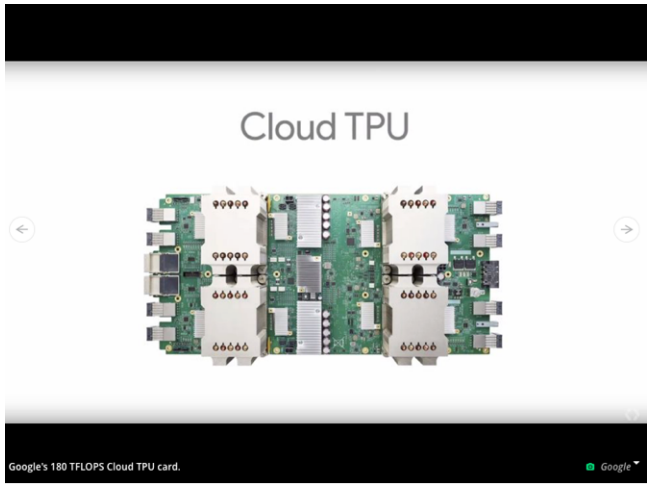
- ~14 million images
- features: full resolution
- http://www.image-net.org

Questions about DL History?

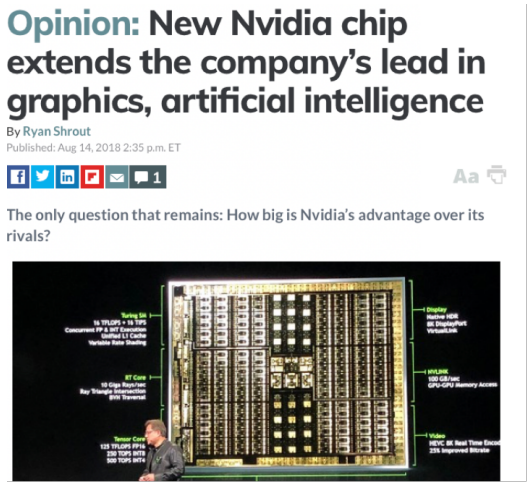
Today: A Brief History of DL

1. Artificial neurons
2. Multilayer neural networks
3. Deep Learning
4. **The DL Hardware & Software Landscape**
5. Current Research Trends

Developing Specialized Hardware



https://arstechnica.com/gadgets/2018/07/the-ai-revolution-has-spawned-a-new-chips-arms-race/



Opinion: New Nvidia chip extends the company's lead in graphics, artificial intelligence

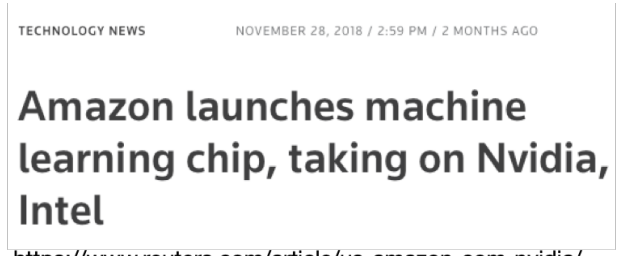
By Ryan Shrout

Published: Aug 14, 2018 2:35 p.m. ET

The only question that remains: How big is Nvidia's advantage over its rivals?

https://www.marketwatch.com/story/new-nvidia-chip-extends-the-companys-lead-in-graphics-artificial-intelligence-2018-08-14

https://developer.arm.com/products/processors/machine-learning/arm-ml-processor

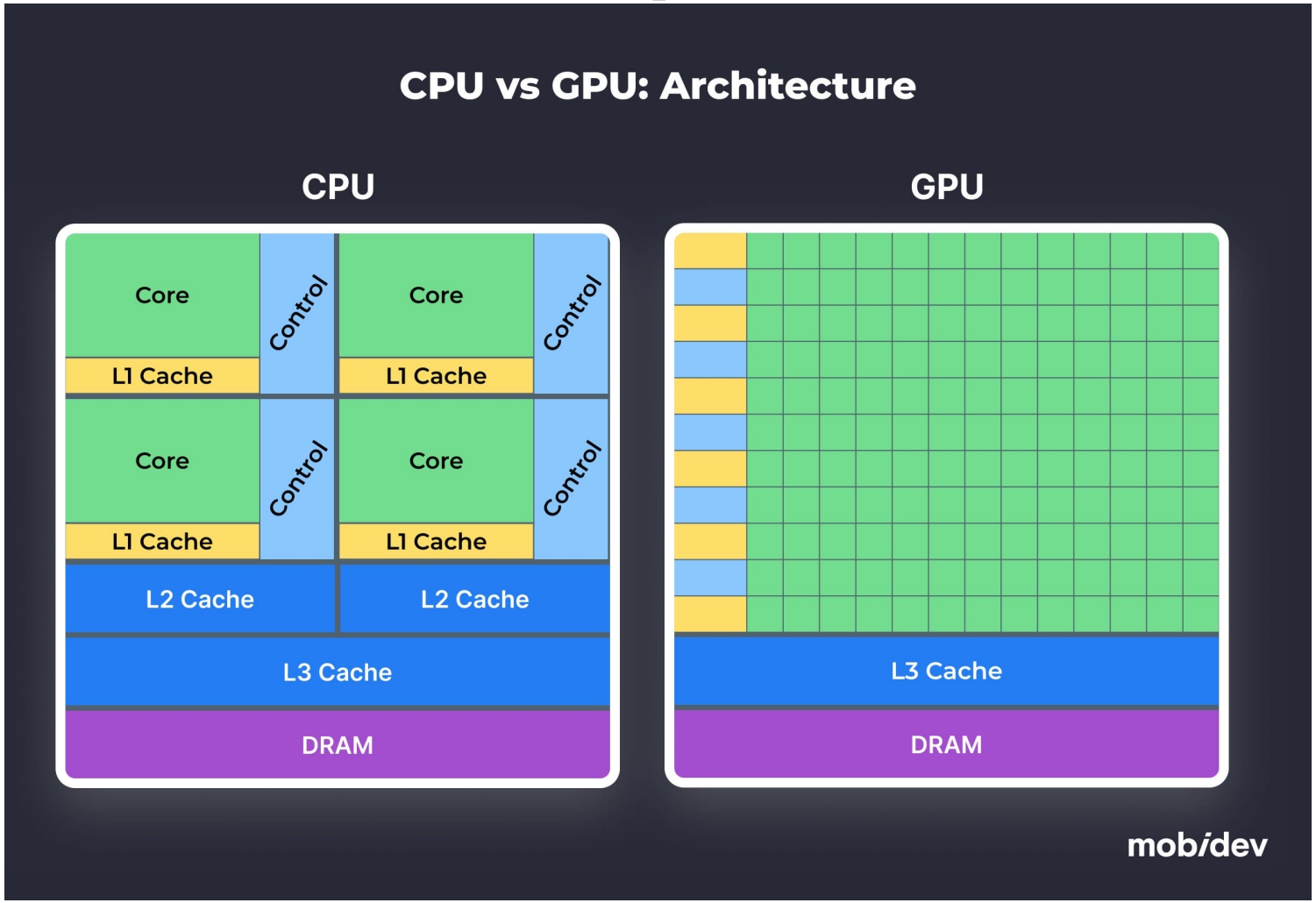


TECHNOLOGY NEWS NOVEMBER 28, 2018 / 2:59 PM / 2 MONTHS AGO

Amazon launches machine learning chip, taking on Nvidia, Intel

https://www.reuters.com/article/us-amazon-com-nvidia/amazon-launches-machine-learning-chip-taking-on-nvidia-intel-idUSKCN1NX2PY

Hardware for Matrix Multiplications



https://mobidev.biz/blog/gpu-machine-learning-on-premises-vs-cloud

Today: A Brief History of DL

1. Artificial neurons
2. Multilayer neural networks
3. Deep Learning
4. The DL Hardware & Software Landscape
5. **Current Research Trends**

Self-supervised learning / pretext tasks

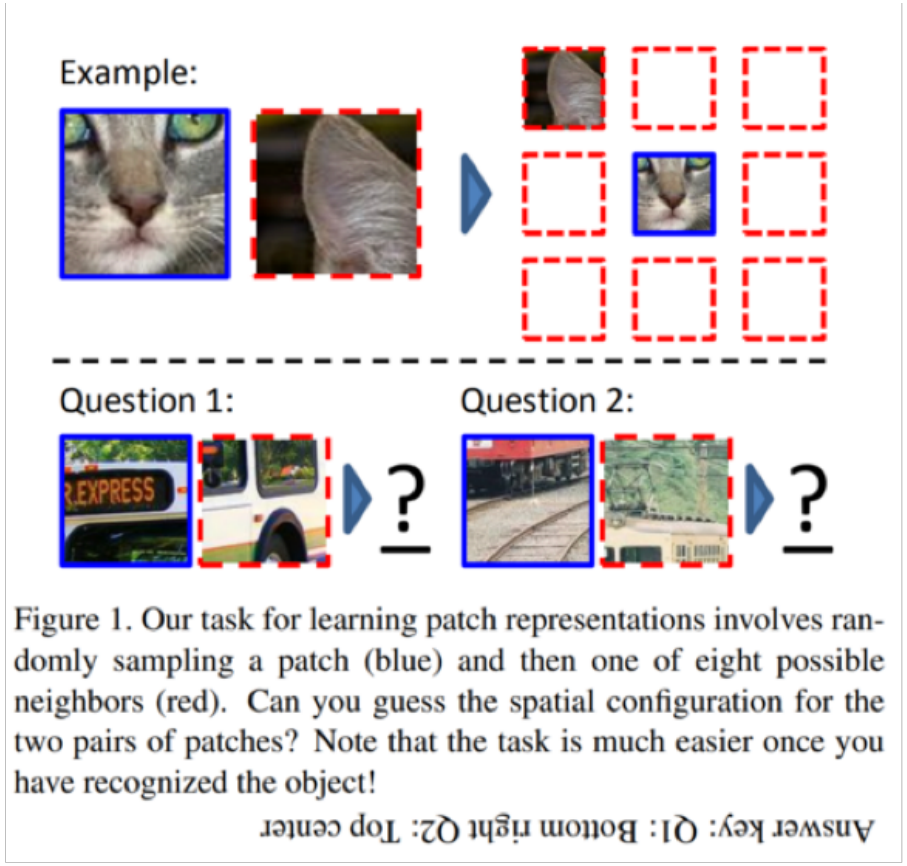
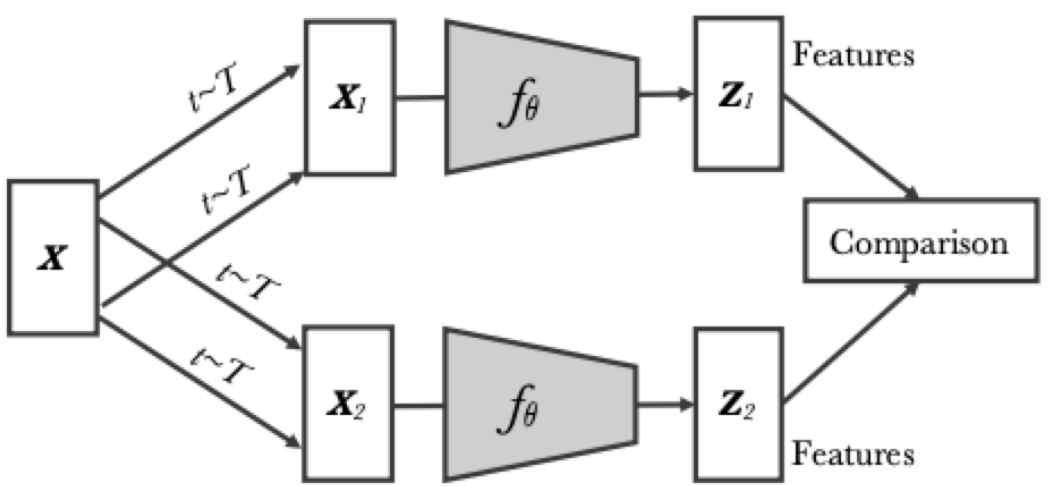


Figure 1. Our task for learning patch representations involves randomly sampling a patch (blue) and then one of eight possible neighbors (red). Can you guess the spatial configuration for the two pairs of patches? Note that the task is much easier once you have recognized the object!

Answer key: Q1: Bottom right Q2: Top center

Lee, H. Y., Huang, J. B., Singh, M., & Yang, M. H. (2017). Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 667-676).



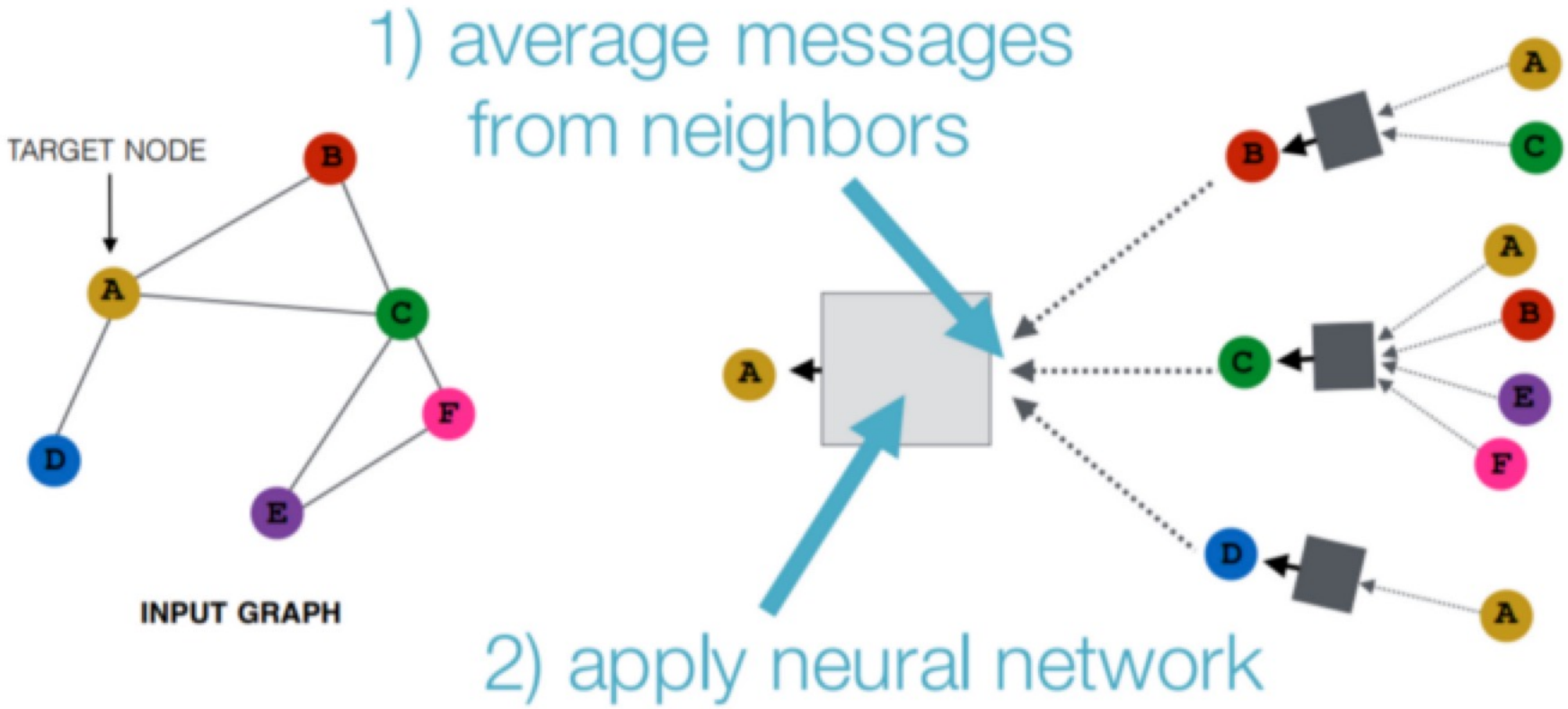
contrastive learning gained popularity in self-supervised representation learning

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., & Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*.

Instance discrimination compares features from different transformations of the same images to each other

DL on all kinds of structured data (graphs, etc.)

A gentle introduction to graph neural networks:
https://heartbeat.fritz.ai/introduction-to-graph-neural-networks-c5a9f4aa9e99



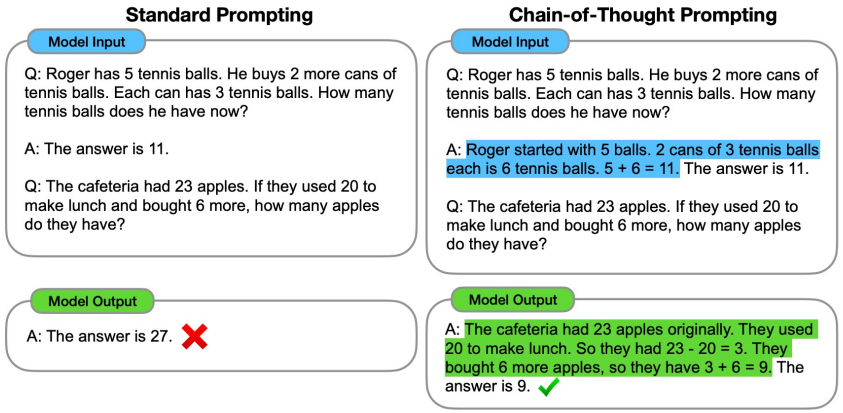
Massive unsupervised learning

From GPT-1 (2018) to GPT-4:

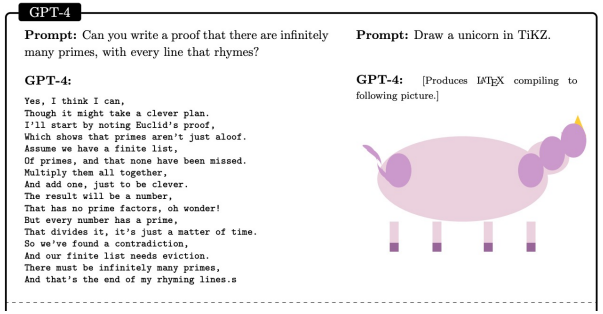
- **Architecture:**
  - **Scale:** Variety of options, with biggest (1.5B params → >1T params):
    - Block size (max context): 512 → 128k
    - Layers: 12 → >96
    - Attention Heads: 12 → >96
    - Embedding Dim: 768 → >12,288
    - Vocab: 40k → >50k tokens
  - Tokenizer: Includes image patches for multimodal
  - Mixture-of-Experts
- **Training:**
  - Dataset: BookCorpus (5GB) → Private 13T tokens (~50TB)
  - Reinforcement learning for alignment

Alignment of large systems with human values

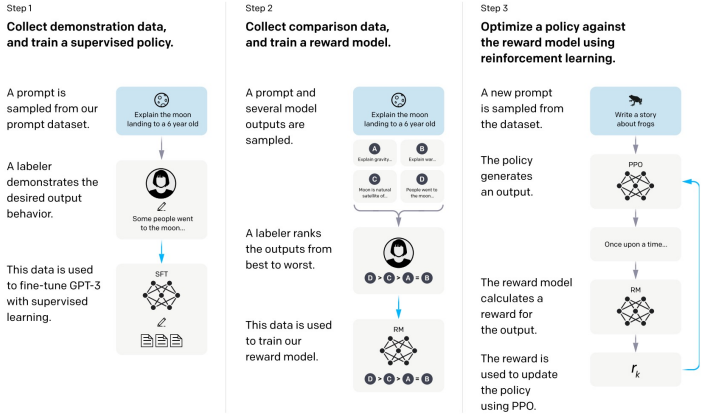
Chain-of-Thought Prompting Elicits Reasoning in Large Language Models



Experiments with Large Language models Sparks of Artificial General Intelligence: Early experiments with GPT-4



Training language models to follow instructions with human feedback (RLHF)



Many directions open…

- Verifiable Rewards
  - Code generation, math calculations, etc.
- Vision-Language models and learning with multimodal data
- Large-scale reinforcement learning
- Model uncertainty, hallucinations
- Model editing, interpretability
- Model synthesis, connections, communication

Next Lecture

Stats / linear algebra / calculus review

Questions?