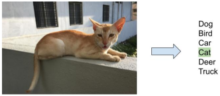
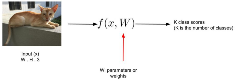
Deep Learning for Computer Vision

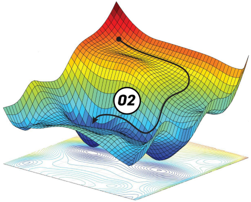
Dr. Konda Reddy Mopuri
Mehta Family School of Data Science and Artificial Intelligence
IIT Guwahati
Aug-Dec 2022

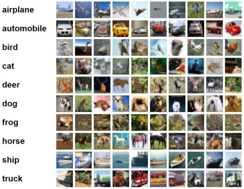
So far in the course...

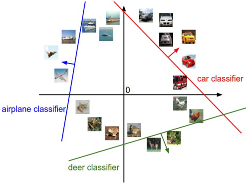
- Image classification: elementary task in CV
- Linear classifier: scoring and loss functions
- Optimization: Gradient descent

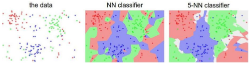












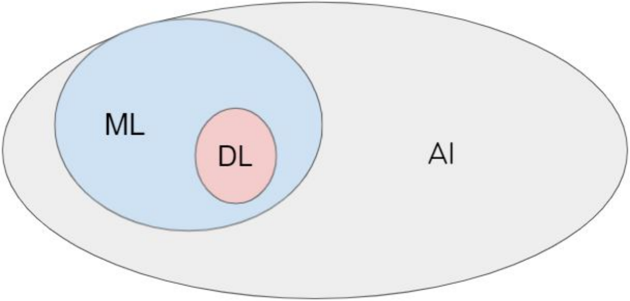
What is DL?

- Subset of ML that is essentially neural networks with more layers

What is DL?

- Subset of ML that is essentially neural networks with more layers
- “Crude” attempt to imitate the human brain in learning

What is DL?



Classical ML vs. DL

- Classical ML: Handcrafted features + learnable model
- Need strong domain expertise

Classical ML vs. DL

- Classical ML: Handcrafted features + learnable model
- Need strong domain expertise

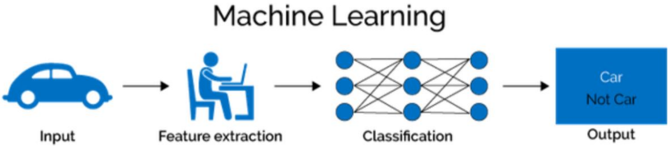


Figure credits: Jay Shaw & Quora

Classical ML vs. DL

- Deep Learning: Deep stack of parameterized processing (NN layers)
- End-to-End learning

Classical ML vs. DL

- Deep Learning: Deep stack of parameterized processing (NN layers)
- End-to-End learning

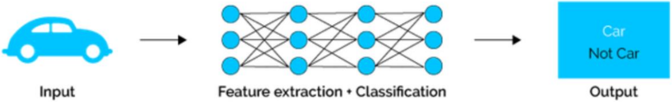


Figure credits: Jay Shaw & Quora

Classical ML vs. DL

- ANNs predate some of the classical ML techniques
- We are now dealing with a new generation ANNs

Neuron

- About 100 billion neurons in human brain

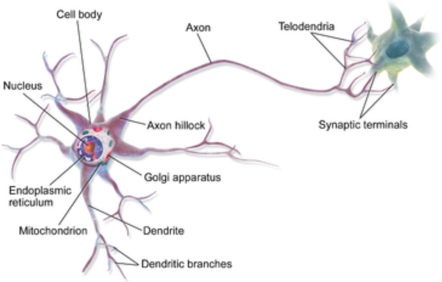


Figure credits: Wikipedia

History of Neural Networks

- McCulloch Pitts neuron (1943) - Threshold Logic Unit

History of Neural Networks

- McCulloch Pitts neuron (1943) - Threshold Logic Unit
- Donald Hebb (1949) - Hebbian Learning Principle

History of Neural Networks

- McCulloch Pitts neuron (1943) - Threshold Logic Unit
- Donald Hebb (1949) - Hebbian Learning Principle
- Marvin Minsky (1951) - created the first ANN (Hebbian Learning, 40 neurons)

History of Neural Networks

- McCulloch Pitts neuron (1943) - Threshold Logic Unit
- Donald Hebb (1949) - Hebbian Learning Principle
- Marvin Minsky (1951) - created the first ANN (Hebbian Learning, 40 neurons)
- Frank Rosenblatt (1958) - created perceptron to classify 20X20 images

History of Neural Networks

- McCulloch Pitts neuron (1943) - Threshold Logic Unit
- Donald Hebb (1949) - Hebbian Learning Principle
- Marvin Minsky (1951) - created the first ANN (Hebbian Learning, 40 neurons)
- Frank Rosenblatt (1958) - created perceptron to classify 20X20 images
- David H Hubel and Torsten Wiesel (1959) demonstrated orientation selectivity and columnar organization in cat's visual cortex

History of Neural Networks

- McCulloch Pitts neuron (1943) - Threshold Logic Unit
- Donald Hebb (1949) - Hebbian Learning Principle
- Marvin Minsky (1951) - created the first ANN (Hebbian Learning, 40 neurons)
- Frank Rosenblatt (1958) - created perceptron to classify 20X20 images
- David H Hubel and Torsten Wiesel (1959) demonstrated orientation selectivity and columnar organization in cat's visual cortex
- Paul Werbos (1982) proposed back-propagation for ANNs

Deep Learning

- Natural generalization to ANNs - Doesn't differ much from the 90s NNs

Deep Learning

- Natural generalization to ANNs - Doesn't differ much from the 90s NNs
- Computational graph of tensor operations that take advantage of
  - Chain rule (back-propagation)
  - SGD
  - GPUs
  - Huge datasets
  - Convolutions, etc.

ILSVRC Error

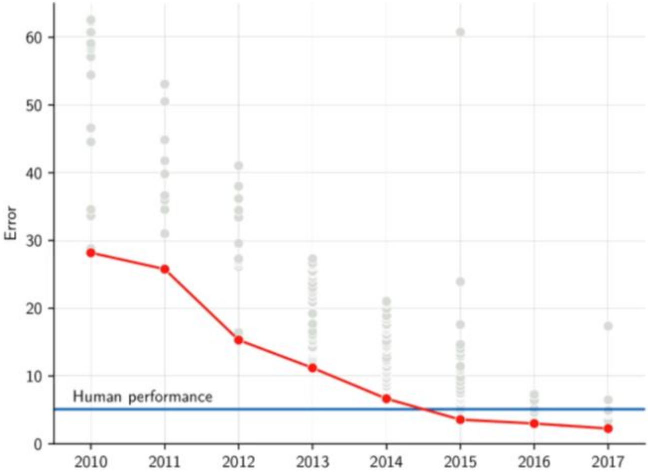


Figure credits: Gershgorn, 2017

What makes it work now?

- Huge research and progress in ML

What makes it work now?

- Huge research and progress in ML
- Hardware developments - CPUs/GPUs/Storage technologies

What makes it work now?

- Huge research and progress in ML
- Hardware developments - CPUs/GPUs/Storage technologies
- Piles of data over the Internet

What makes it work now?

- Huge research and progress in ML
- Hardware developments - CPUs/GPUs/Storage technologies
- Piles of data over the Internet
- Collaborative development (open source tools and forums for sharing/discussions, etc)

What makes it work now?

- Huge research and progress in ML
- Hardware developments - CPUs/GPUs/Storage technologies
- Piles of data over the Internet
- Collaborative development (open source tools and forums for sharing/discussions, etc)
- Collective efforts from large institutions/corporations
- . . .

What makes it work now?

- We have been doing a lot of ML already
  - Taxonomy of ML concepts: Classification, regression, generative models, clustering, etc.
  - Rich statistical formalizations: Bayesian estimation, PAC, etc.
  - Understood fundamentals: Bias-Variance, VC dimension, etc.
  - Good understanding of optimization
  - Efficient large-scale algorithms

Deep Learning - practical perspective

- Doesn't require a deep mathematical grasp(!?)

Deep Learning - practical perspective

- Doesn't require a deep mathematical grasp(!?)
- Makes the design of large models a system/software development task

Deep Learning - practical perspective

- Doesn't require a deep mathematical grasp(!?)
- Makes the design of large models a system/software development task
- Leverages modern hardware

Deep Learning - practical perspective

- Doesn't require a deep mathematical grasp(!?)
- Makes the design of large models a system/software development task
- Leverages modern hardware
- Doesn't seem to plateau with more data

Deep Learning - practical perspective

- Doesn't require a deep mathematical grasp(!?)
- Makes the design of large models a system/software development task
- Leverages modern hardware
- Doesn't seem to plateau with more data
- Makes the trained models a commodity

Compute getting cheaper

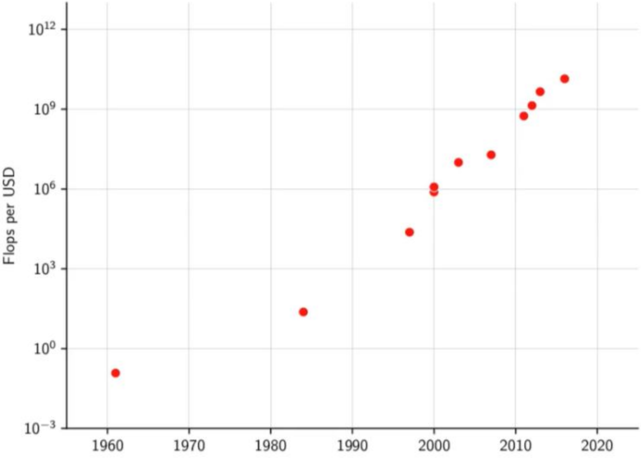


Figure Credits: Wikipedia

Storage getting cheaper

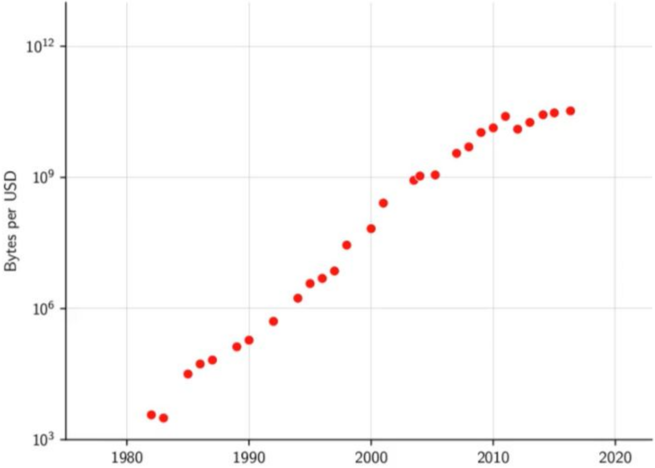


Figure Credits: John C Mccallum

AlexNet to AlphaGo: 300000X increase in compute

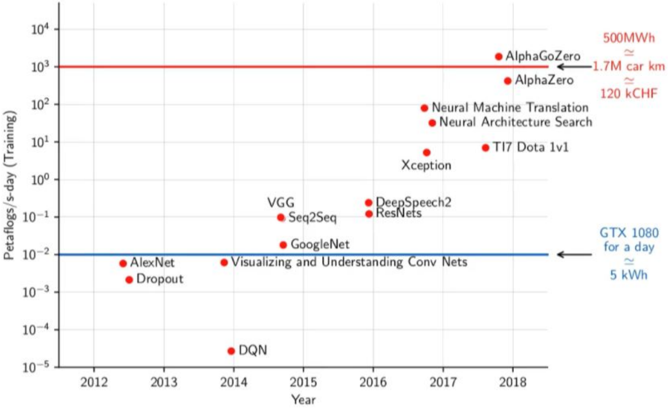


Figure Credits: Radford, 2018. 1 petaflop/s-day ≈ 100 GTX 1080 GPUs for a day, ≈ 500kwh

Datasets

Data-set		Year	Nb. images	Size
MNIST	(classification)	1998	60K	12Mb
Caltech 101	(classification)	2003	9.1K	130Mb
Caltech 256	(classification)	2007	30K	1.2Gb
CIFAR10	(classification)	2009	60K	160Mb
ImageNet	(classification)	2012	1.2M	150Gb
MS-COCO	(segmentation)	2015	200K	32Gb
Cityscape	(segmentation)	2016	25K	60Gb

Data-set		Year	Size
SST2	(sentiment analysis)	2013	20Mb
WMT-18	(translation)	2018	7Gb
OSCAR	(language model)	2020	6Tb

Figure Credits: François Fleuret

Implementation

	Language(s)	License	Main backer
PyTorch	**Python**, C++	BSD	Facebook
TensorFlow	Python, C++	Apache	Google
JAX	Python	Apache	Google
MXNet	Python, C++, R, Scala	Apache	Amazon
CNTK	Python, C++	MIT	Microsoft
Torch	Lua	BSD	Facebook
Theano	Python	BSD	U. of Montreal
Caffe	C++	BSD 2 clauses	U. of CA, Berkeley

Figure Credits: François Fleuret

We use PyTroch for this course

PyTorch

http://pytorch.org

Threshold Logic Unit

- First Mathematical Model for a neuron

Threshold Logic Unit

- First Mathematical Model for a neuron
- McCulloch and Pitts, $1943 \rightarrow$ MP neuron

Threshold Logic Unit

- First Mathematical Model for a neuron
- McCulloch and Pitts, $1943 \to$ MP neuron
- Boolean inputs and output

$$f(x) = \mathbb{1}(w \sum_i x_i + b \geq 0) \tag{1}$$

Threshold Logic Unit

- let's implement simple functions

Threshold Logic Unit

- let's implement simple functions
- NOT

Threshold Logic Unit

- let's implement simple functions
- NOT
  - $\mathsf{NOT}(x) = \mathbb{1}(-x + 0.5 \geq 0)$

Threshold Logic Unit

- let's implement simple functions
- NOT
  - NOT$(x) = \mathbb{1}(-x + 0.5 \geq 0)$
- OR

Threshold Logic Unit

- let's implement simple functions
- NOT
  - NOT$(x) = \mathbb{1}(-x + 0.5 \geq 0)$
- OR
  - OR$(x, y) = \mathbb{1}(x + y - 0.5 \geq 0)$

Threshold Logic Unit

- let's implement simple functions
- NOT
  - $\text{NOT}(x) = \mathbb{1}(-x + 0.5 \geq 0)$
- OR
  - $\text{OR}(x, y) = \mathbb{1}(x + y - 0.5 \geq 0)$
- AND

Threshold Logic Unit

- let's implement simple functions
- NOT
  - NOT$(x) = \mathbb{1}(-x + 0.5 \geq 0)$
- OR
  - OR$(x, y) = \mathbb{1}(x + y - 0.5 \geq 0)$
- AND
  - AND$(x, y) = \mathbb{1}(x + y - 1.5 \geq 0)$

Threshold Logic Unit

- Can realize any Boolean function using TLUs

Threshold Logic Unit

- Can realize any Boolean function using TLUs
- What one unit does? - Learn linear separation

Threshold Logic Unit

- Can realize any Boolean function using TLUs
- What one unit does? - Learn linear separation
- No learning; heuristics approach

Perceptron

- Frank Rosenblatt 1957 (American Psychologist)

Perceptron

- Frank Rosenblatt 1957 (American Psychologist)
- Very crude biological model

Perceptron

- Frank Rosenblatt 1957 (American Psychologist)
- Very crude biological model
- Similar to MP neuron - Performs linear classification

Perceptron

- Frank Rosenblatt 1957 (American Psychologist)
- Very crude biological model
- Similar to MP neuron - Performs linear classification
- Inputs can be real, weights can be different for different i/p components

Perceptron

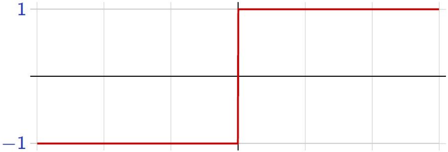
- Frank Rosenblatt 1957 (American Psychologist)
- Very crude biological model
- Similar to MP neuron - Performs linear classification
- Inputs can be real, weights can be different for different i/p components
-

$$f(x) = \begin{cases} 1 & \text{when } \sum_i w_i x_i + b \geq 0 \\ 0 & \text{else} \end{cases}$$

Perceptron

- For simplicity we consider +1 and -1 responses

$$\sigma(x) = \begin{cases} 1 & \text{when } x \geq 0 \\ -1 & \text{else} \end{cases}$$

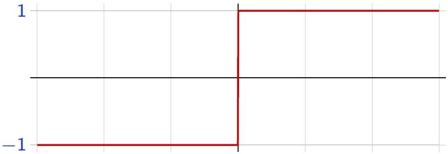


$$f(\mathbf{x}) = \sigma(\mathbf{w^T} \cdot \mathbf{x} + \mathbf{b})$$

Perceptron

- For simplicity we consider +1 and -1 responses

$$\sigma(x) = \begin{cases} 1 & \text{when } x \geq 0 \\ -1 & \text{else} \end{cases}$$



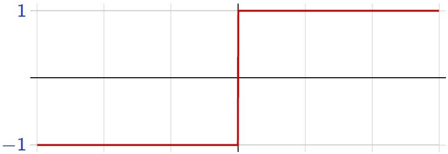
$$f(\mathbf{x}) = \sigma(\mathbf{w^T} \cdot \mathbf{x} + \mathbf{b})$$

- In general, $\sigma(\cdot)$ that follows a linear operation is called an activation function

Perceptron

- For simplicity we consider +1 and -1 responses

$$\sigma(x) = \begin{cases} 1 & \text{when } x \geq 0 \\ -1 & \text{else} \end{cases}$$



$$f(\mathbf{x}) = \sigma(\mathbf{w}^{\mathbf{T}} \cdot \mathbf{x} + \mathbf{b})$$

- In general, $\sigma(\cdot)$ that follows a linear operation is called an activation function
- $\mathbf{w}$ are referred to as weights and b as the bias

Perceptron vs. MP neuron

- Perceptron is more general computational model

Perceptron vs. MP neuron

- Perceptron is more general computational model
- Inputs can be real

Perceptron vs. MP neuron

- Perceptron is more general computational model
- Inputs can be real
- Weights are different on the input components

Perceptron vs. MP neuron

- Perceptron is more general computational model
- Inputs can be real
- Weights are different on the input components
- Mechanism for learning weights!

Weights and Bias

- Why are the weights important?

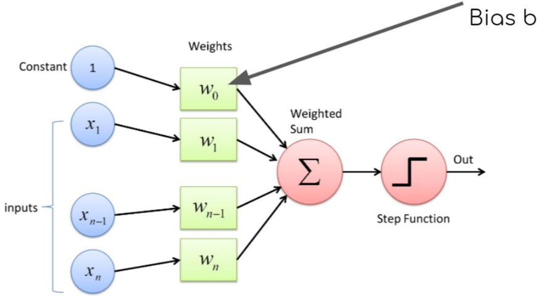


Figure credits: DeepAI

Weights and Bias

- Why are the weights important?
- Why is it called ‘bias’? What does it capture?

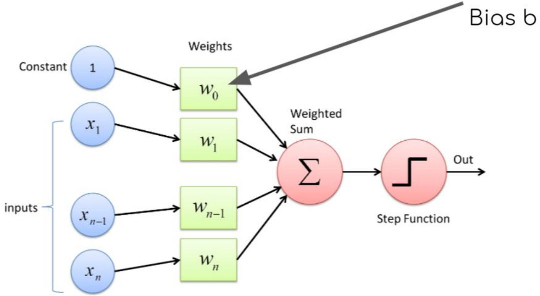


Figure credits: DeepAI

Perceptron

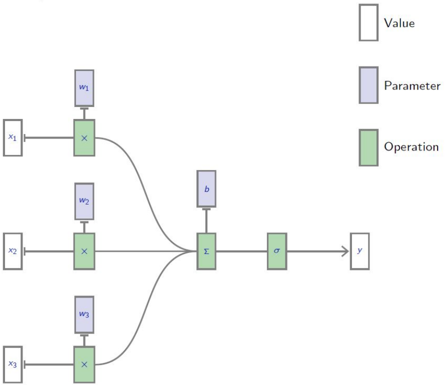


Figure credits: François Fleuret

Perceptron

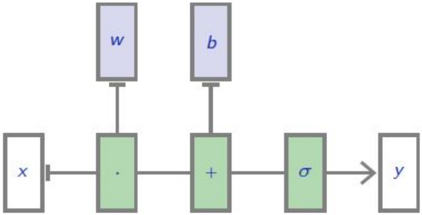


Figure credits: François Fleuret

Perceptron Learning algorithm

- Training data $(x_n, y_n) \in \mathcal{R}^D \times -1, 1, n = 1, \ldots, N$

Perceptron Learning algorithm

- Training data $(x_n, y_n) \in \mathcal{R}^D \times -1, 1, n = 1, \ldots, N$
- Start with $\mathbf{w} = \mathbf{0}$

Perceptron Learning algorithm

- Training data $(x_n, y_n) \in \mathcal{R}^D \times -1, 1, n = 1, \ldots, N$
- Start with $\mathbf{w} = \mathbf{0}$
- While $\exists n_k$ such that $y_{nk}(\mathbf{w_k^T} \cdot \mathbf{x_{nk}}) \leq \mathbf{0}$, update $\mathbf{w_{k+1}} = \mathbf{w_k} + \mathbf{y_{nk}} \cdot \mathbf{x_{nk}}$

Perceptron Learning algorithm

- Training data $(x_n, y_n) \in \mathcal{R}^D \times -1, 1, n = 1, \ldots, N$
- Start with $\mathbf{w} = \mathbf{0}$
- While $\exists n_k$ such that $y_{nk}(\mathbf{w_k^T} \cdot \mathbf{x_{nk}}) \leq \mathbf{0}$, update $\mathbf{w_{k+1}} = \mathbf{w_k} + \mathbf{y_{nk}} \cdot \mathbf{x_{nk}}$
- Note that the bias b is absorbed as a component of $\mathbf{w}$ and $\mathbf{x}$ is appended with 1 suitably

Perceptron Learning Algorithm

▸ Colab Notebook: Perceptron

Perceptron Learning Algorithm

- Convergence result: Can show that after some iterations, no training sample gets misclassified

Perceptron Learning Algorithm

- Convergence result: Can show that after some iterations, no training sample gets misclassified
- Stops as soon as it finds a separating boundary

Perceptron Learning Algorithm

- Convergence result: Can show that after some iterations, no training sample gets misclassified
- Stops as soon as it finds a separating boundary
- Other algorithms maximize the margin from boundary to the samples

Next lecture..

- More on NNs: MLP, . . .