Neural Networks and Deep Learning

Andrea Asperti

DISI - Department of Informatics: Science and Engineering
University of Bologna
Mura Anteo Zamboni 7, 40127, Bologna, ITALY
andrea.asperti@unibo.it

What is Deep Learning?

Q: **What is Deep Learning?**

What is Deep Learning?

Q: **What is Deep Learning?**

A: A branch of Machine Learning

What is Deep Learning?

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

What is Deep Learning?

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

What is Deep Learning?

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Q: **Are there problems where DL particularly excels?**

What is Deep Learning?

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Q: **Are there problems where DL particularly excels?**

A: All problems involving myriads of features, e.g. images/speech/text processing, recommendation systems, . . .

What is Deep Learning?

Q: **What is Deep Learning?**

A: A branch of Machine Learning

Q: **To what kind of problems it can be applied?**

A: To all problems suitable for Machine Learning

Q: **Are there problems where DL particularly excels?**

A: All problems involving myriads of features, e.g. images/speech/text processing, recommendation systems, . . .

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Q: **Why “Deep”?**

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Q: **Why “Deep”?**

A: - because it exploits Deep Neural Networks, composed by many nested layers of neurons

Deep Learning FAQ (2)

Q: **What is the basic technique for DL?**

A: Neural Networks

Q: **Why "Deep"?**

A:
- because it exploits Deep Neural Networks, composed by many nested layers of neurons
- because it exploits deep features of data, that is features extracted from other features

ML recap

What is Machine Learning about?

There are problems that are difficult to address with traditional programming techniques:

- classify a document according to some criteria (e.g. spam, sentiment analysis, ...)
- compute the probability that a credit card transaction is fraudulent
- recognize an object in some image (possibly from an inusual viewpoint, in new lighting conditions, in a cluttered scene)
- ...

Typically the result is a weighted combination of a large number of parameters, each one contributing to the solution in a small degree.

The Machine Learning approach

Suppose to have a set of input-output pairs (training set)

$$\{\langle x_i, y_i \rangle\}$$

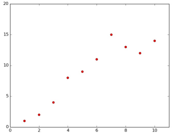
the problem consists in guessing the map $x_i \mapsto y_i$

The M.L. approach:

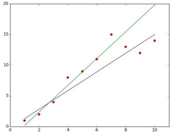
- describe the problem with a **model** depending on some parameters Θ (i.e. choose a parametric class of functions)
- define a **loss function** to compare the results of the model with the expected (experimental) values
- **optimize** (fit) the parameters Θ to reduce the loss to a minimum

Example: a regression problem

You have some points on the plane and you want to fit a line through them

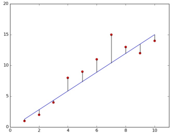


Step 1 Fix a parametric class of models.
For intance linear functions $y = ax + b$;
a and b are the parameters of the model



Step 2 Fix a way to decide when a line is better than another (loss function)
For instance, mean square error (mse)

Step 3 Try to tune the parameters in order to reduce the loss (training).



Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

The point is that the solution to the optimization problem is not given in an analytical form (often there is no closed form solution).

Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

The point is that the solution to the optimization problem is not given in an analytical form (often there is no closed form solution).

So, we use **iterative** techniques (typically, gradient descent) to progressively approximate the result.

Why Learning?

Machine Learning problems are in fact **optimization problems**!
So, why talking about learning?

The point is that the solution to the optimization problem is not given in an analytical form (often there is no closed form solution).

So, we use **iterative** techniques (typically, gradient descent) to progressively approximate the result.

This form of iteration over data can be understood as a way of progressive learning of the objective function based on the experience of past observations.

Using gradients

Goal: minimize a loss function E over (fixed) training samples:

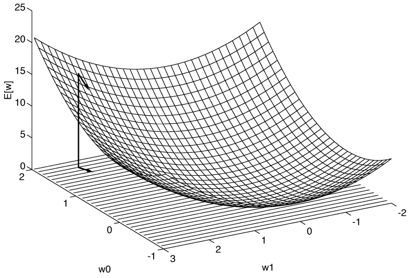
$$\Theta(w) = \sum_i E(o(w, x_i), y_i)$$

See how it changes according to small perturbations $\Delta(w)$ of the parameters w: this is the gradient

$$\nabla_w[\theta] = [\frac{\partial \Theta}{\partial w_1}, \ldots, \frac{\partial \Theta}{\partial w_n}]$$

of Θ w.r.t. w.

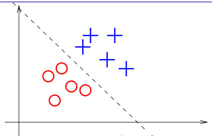
The gradient is a vector pointing in the direction of steepest ascent.



A bit of taxonomy

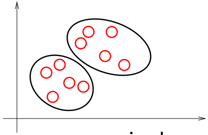
Different types of Learning Tasks

- **supervised learning**:
  inputs + outputs (labels)
  - classification
  - regression



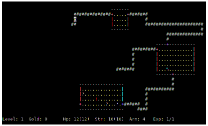
supervised

- **unsupervised learning**:
  just inputs
  - clustering
  - component analysis
  - anomaly detection
  - generative techniques



unsupervised

- **reinforcement learning**
  actions and rewards
  - learning long-term gains
  - planning



reinforcement

Discriminative vs. Generative AI

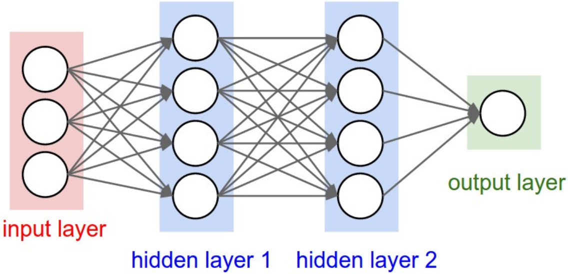
A different dichotomy:

- **Discriminative models** learn decision boundaries over data.
- **Generative models** learn the data distribution.

Discriminative models typically require supervised data, but generative models can be trained in both supervised and unsupervised settings (e.g. for conditional generation).

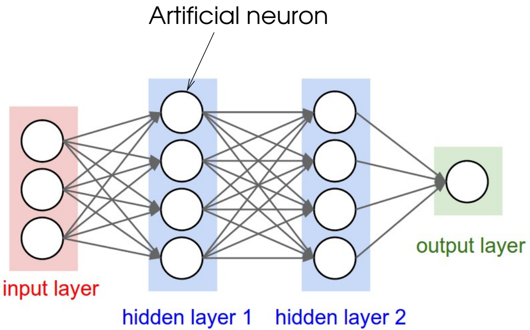
Next argument

Neural Networks



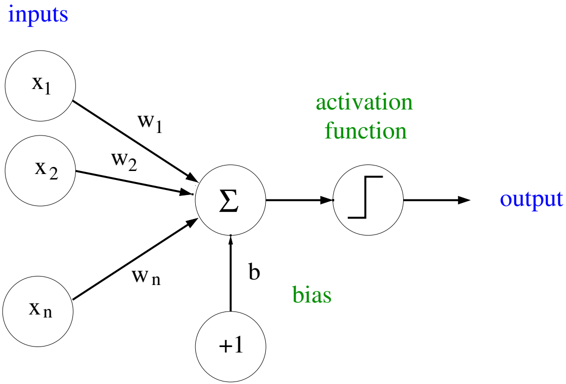
Neural Network

A network of (artificial) neurons



Each neuron takes multiple inputs and produces a single output (that can be passed as input to many other neurons).

The artificial neuron

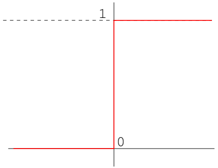


Each neuron (!) implements a logistic regressor

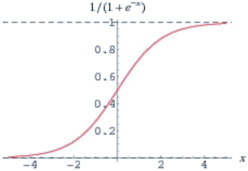
$$\sigma(wx + b)$$

Different activation functions

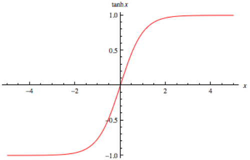
The activation function is responsible for threshold triggering.



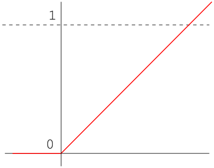
threshold: if $x > 0$ then 1 else 0



logistic function: $\frac{1}{1+e^{-x}}$

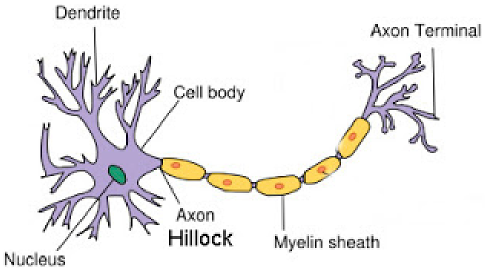


hyperbolic tangent: $\frac{e^x - e^{-x}}{e^x + e^{-x}}$



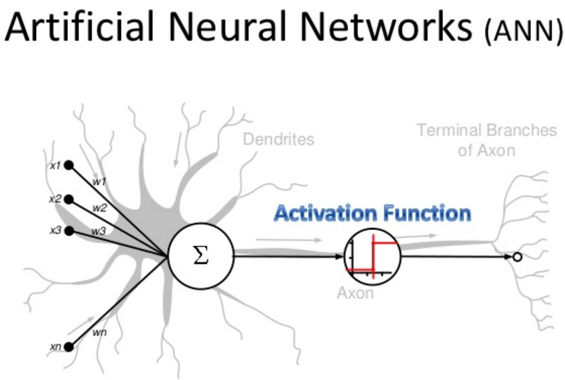
rectified linear (RELU): if $x > 0$ then x else 0

The cortical neuron



- the dendritic tree of the cell collects inputs from other neurons, that get summed together
- when a triggering threshold is exceeded, the Axon Hillock generate an impulse that get transmitted through the axon to other neurons.

A comparison with the cortical neuron



Slide credit : Andrew L. Nelson

Some figures for human brains

- number of neurons: $\sim 2 \cdot 10^{10}$
- switching time for neuron: $1 - 5$ ms. (slow!)
- synapses (connections) per neuron: $\sim 10^{4-5}$
- time to process an image: 100 ms.

not too deep (< 100)

very high parallelism

Motivations behind neural computation

- to understand, via simulation, how the brain works
- to investigate a different paradigm of computation

  very far from traditional programming languages
- **to solve practical problems difficult to address with algorithmic techniques**

  useful even if the brain works in a different way

Next argument

A closer look at Networks

Feed-forward and recurrent networks

If the network is acyclic, it is called a feed-forward network.

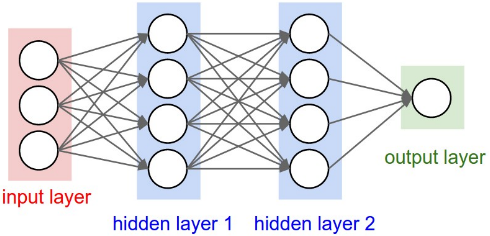
If it has cycles it is called recurrent.

The large majority of modern NNs are feed-forward.

The main class of recurrent networks are the so called Long Short-Term Memory models (LSTM) and variants.

Layers

In a feed-forward network, neurons are usually organized in layers.



If there is more than one hidden layer the network is deep, otherwise it is called a shallow network.

Layers and Tensors

Each Layer takes in input a Tensor and returns a new Tensor.

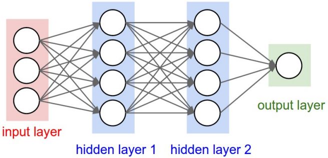
A Tensor is simply a multi-dimensional array.
A Layer computes an algebraic mainupaltion of the inpout tensor.

There are many different kind of layers, serving different purposes:

- **Processing layers**: dense (linear), convolutional, attention, ...
- **Shaping layers**: Upsampling, Dowsampling, Pooling, Reshape, ...
- **Structural layers**: concatenation (shortcuts), sum (residuality), ...
- **Normalization and regularization layers**: BatchNormalization, Dropout, ...

An example: the dense layer

Dense layer: each neuron at layer $k-1$ is connected to **each each** neuron at layer k.



A single neuron:

$$I^n \cdot W^n + B^1 = O^1$$

the operation can be vectorized to pruduce m outputs in parallel:

$$I^n \cdot W^{n \times m} + B^m = O^m$$

- dense layers usually work on flat (unstructured) inputs
- the order of elements in input is irrelevant

Parameters and hyper-parameters

The weights W_k are the parameters of the model: they are **learned** during the training phase.

The number of neurons and the way they are connected together are hyper-parameters: they are chosen by the user and **fixed** before training may start.

Other important hyper-parameters govern training such as learning rate, batch-size, number of ephocs an many others.

A first example

[demo]

Understanding Deep Learning

- understand the **different layers**, and their purpose
- understand how layers can be organized in **relevant architectures**
- understand the different possible **applications** of DL, and their specific solutions
- understand the main **issues, problems** and **costs**

Frameworks for DL

- PyTorch, Meta, OpenAI
- TensorFlow/Keras, Google Brain
- JAX, Google Research
- MindSpore, Huawey

We shall mostly make examples in Keras, for its simplicitly.

Historical remarks - Legacy

Legacy

1958	perceptron
1975	backpropagation
1980	convolutional layers
1992	Max-pooling
1997	LSTM
...	...

Extremely slow progress

Until 2010 Neural Networks played a completely marginal role in AI and Computer Science

The Deep Learning revolution

2011	Google Brain foundation	2017	PPO
2012	ReLU and Dropout	2018	JAX release
2012	ImageNet Competition	2018	Transformers
2013	DQN	2018	BERT/GPT
2014	GANs	2018	Soft Actor Critic
2014	Attention	2019	AlphaStar
2014	Inception v1	2020	Vision Transformes
2015	Tensorflow release	2020	OpenAI Jukebox
2015	Keras release	2021	CLIP, Dall·E
2015	AlphaGo	2022	Dall·E 2, Imagen
2015	Batchnormalization	2022	ChatGPT
2015	YOLO v1	2023	Gemini
2015	OpenAI foundation	2023	LLaMA-2
2016	ResNet - residuality	2023	ChatGPT-4
2017	PyTorch release	2024	OpenAI's o1
2017	Mask-RCNN	2025	DeepSeek's R1

Just to mention a few milestones ...

The Deep Learning revolution

Frameworks and Libraries

2011	Google Brain foundation	2017	PPO
2012	ReLU and Dropout	2018	**JAX release**
2012	ImageNet Competition	2018	Transformers
2013	DQN	2018	BERT/GPT
2014	GANs	2018	Soft Actor Critic
2014	Attention	2019	AlphaStar
2014	Inception v1	2020	Vision Transformes
2015	**Tensorflow release**	2020	OpenAI Jukebox
2015	**Keras release**	2021	CLIP, Dall·E
2015	AlphaGo	2022	Dall·E 2, Imagen
2015	Batchnormalization	2022	ChatGPT
2015	YOLO v1	2023	Gemini
2015	OpenAI foundation	2023	LLaMA-2
2016	ResNet - residuality	2023	ChatGPT-4
2017	**PyTorch release**	2024	OpenAI's o1
2017	Mask-RCNN	2025	DeepSeek's R1

The Deep Learning revolution

Notable technical improvements

2011	Google Brain foundation
2012	**ReLU and Dropout**
2012	ImageNet Competition
2013	DQN
2014	GANs
2014	**Attention**
2014	Inception v1
2015	Tensorflow release
2015	Keras release
2015	AlphaGo
2015	**Batchnormalization**
2015	YOLO v1
2015	OpenAI foundation
2016	ResNet - **residuality**
2017	PyTorch release
2017	Mask-RCNN
2017	PPO
2018	JAX release
2018	Transformers
2018	BERT/GPT
2018	Soft Actor Critic
2019	AlphaStar
2020	Vision Transformes
2020	OpenAI Jukebox
2021	CLIP, Dall·E
2022	Dall·E 2, Imagen
2022	ChatGPT
2023	Gemini
2023	LLaMA-2
2023	ChatGPT-4
2024	OpenAI's o1
2025	DeepSeek's R1

The Deep Learning revolution

Important Architectures

2011	Google Brain foundation	2017	PPO
2012	ReLU and Dropout	2018	JAX release
2012	ImageNet Competition	2018	**Transformers**
2013	DQN	2018	BERT/GPT
2014	**GANs**	2018	Soft Actor Critic
2014	Attention	2019	AlphaStar
2014	**Inception v1**	2020	**Vision Transformes**
2015	Tensorflow release	2020	OpenAI Jukebox
2015	Keras release	2021	CLIP, Dall·E
2015	AlphaGo	2022	Dall·E 2, Imagen
2015	Batchnormalization	2022	ChatGPT
2015	**YOLO v1**	2023	Gemini
2015	OpenAI foundation	2023	LLaMA-2
2016	**ResNet** - residuality	2023	ChatGPT-4
2017	PyTorch release	2024	OpenAI's o1
2017	**Mask-RCNN**	2025	DeepSeek's R1

The Deep Learning revolution

Deep Reinforcement Learning

2011	Google Brain foundation
2012	ReLU and Dropout
2012	ImageNet Competition
2013	**DQN**
2014	GANs
2014	Attention
2014	Inception v1
2015	Tensorflow release
2015	Keras release
2015	**AlphaGo**
2015	Batchnormalization
2015	YOLO v1
2015	OpenAI foundation
2016	ResNet - residuality
2017	PyTorch release
2017	Mask-RCNN
2017	**PPO**
2018	JAX release
2018	Transformers
2018	BERT/GPT
2018	**Soft Actor Critic**
2019	AlphaStar
2020	Vision Transformes
2020	OpenAI Jukebox
2021	CLIP, Dall·E
2022	Dall·E 2, Imagen
2022	ChatGPT
2023	Gemini
2023	LLaMA-2
2023	ChatGPT-4
2024	OpenAI's o1
2025	DeepSeek's R1

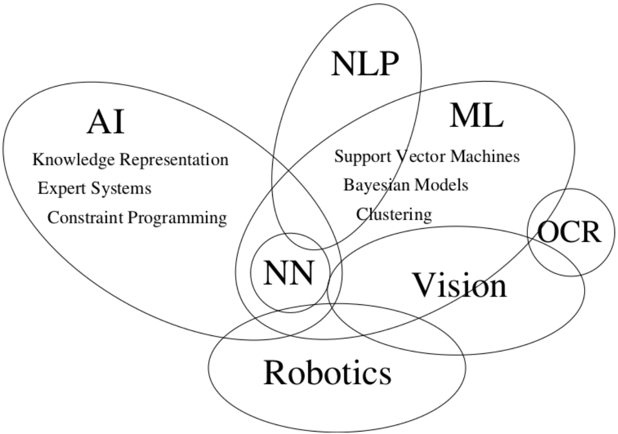
The New Revolution

Generative AI and Large Language Models

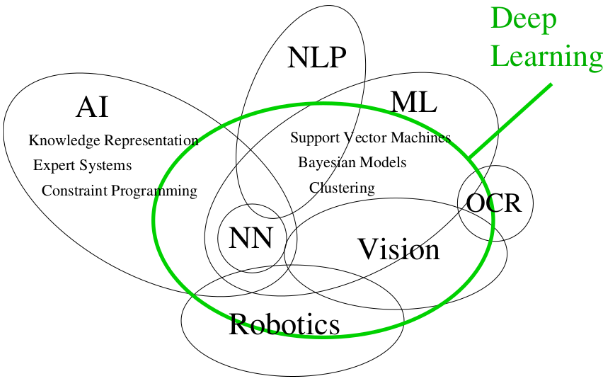
2011	Google Brain foundation
2012	ReLU and Dropout
2012	ImageNet Competition
2013	DQN
2014	GANs
2014	**Attention**
2014	Inception v1
2015	Tensorflow release
2015	Keras release
2015	AlphaGo
2015	Batchnormalization
2015	YOLO v1
2015	OpenAI foundation
2016	ResNet - residuality
2017	PyTorch release
2017	Mask-RCNN

2017	PPO
2018	JAX release
2018	**Transformers**
2018	**BERT/GPT**
2018	Soft Actor Critic
2019	AlphaStar
2020	**Vision Transformes**
2020	**OpenAI Jukebox**
2021	**CLIP, Dall·E**
2022	**Dall·E 2, Imagen**
2022	**ChatGPT**
2023	**Gemini**
2023	**LLaMA-2**
2023	**ChatGPT-4**
2024	**OpenAI's o1**
2025	**DeepSeek's R1**

The situation at the beginning of the century



The deep learnig era



See my blog for a short historical perspective.

Didactic Information

- structure of the course
- books, tutorials and blogs
- software
- exam
- office hours

Structure of the course

Frontal lessons intermixed with demos + labs

- Domains of application of Deep Learning
- Expressiveness
- Backpropagation
- Computational Layers
- Other layers
- Major Architectures
- Interpretability
- Object Detection and Segmentation
- Generative Models for imaging
- Recurrent Networks and LSTM
- Introduction to LLMs
- Reinforcement Learning (if time permits)

Text Books

Suggested reading:

- Dive into deep learning (D2L)

Very good online material (fast updating):

- Tensorflow tutorials
- Towards data science
- Keras blog. By F.Chollet.
- Machine learning tutorial with Python
- Deep Learning Tutorial. LISA lab. University of Montreal.
- a lot of interesting lessons and seminars on youtube
- a lot of material on github
- . . .

Code and Datasets

Important code repositories:

- Hugging Face
- The State of the Art site! (papers with code)

Data sets and more...

- Tensor flow dataset
- Kaggle Datasets
- Coco, . . .
- CelebA, Labeled Faces in the Wild, . . .
- Biomedical challenges
- Amazon Datasets
- ...

Exam modalities:

At each exam session you will receive a project assignement that you are supposed to complete in **7 days**.

You are supposed to deliver:

- the code source (Keras/TensorFlow) in the form of a **single**, **commented** pyhton notebook

The work will be evaluated according to

1. 80% : comparative evaluation of results (measured in an objective way according to given metrics);
2. 20% : descriptive quality of the notebook

You may possibly integrate the grade with an oral examination.

Office hours

Prof.Andrea Asperti
andrea.asperti@unibo.it
Via Zanolini 41 (ex-stazione Veneta)

If you need to talk to me, just send me an email and we shall fix a meeting, either in presence or on line, as you prefer.