

Introduction to Machine Learning

Chap I: Basic concepts

Julien Donini

LPC/Université Clermont Auvergne

Outline of these lectures

Lectures

Chap I: Basic concepts on ML

Chap II: Regression

Chap III: Classification

Practice sessions

Introduction to Machine Learning

Code on Git: 

<https://github.com/judonini/MLcourses2019>

Sections marked with (*) are more advanced



non examinable
for DU students

This week

	S47				
	Lundi 18/11/2019	Mardi 19/11/2019	Mercredi 20/11/2019	Jeudi 21/11/2019	Vendredi 22/11/2019
07h30					
08h00					
08h30					
09h00					
09h30	●	●	●	●	●
10h00	Machine Learning 1/7 09h30 - 11h00 DONINI Julien ST PHYZ 119 M2 UP DU Data Scientist	Machine Learning 4/7 09h30 - 11h00 DONINI Julien ST PPHY 9210 M2 UP DU Data Scientist			
10h30			Machine Learning.. 1/2 09h30 - 12h30 DONINI Julien ST PHYZ 214 M2 UP DU Data Scientist		
11h00					
11h30	●	●			
12h00	Machine Learning 2/7 11h15 - 12h45 Intervenant à préciser ST PHYZ 119 M2 UP DU Data Scientist	Machine Learning 5/7 11h15 - 12h15 DONINI Julien ST PPHY 9210 M2 UP DU Data Scientist			
12h30					
13h00					
13h30		●			
14h00					
14h30					
15h00	●				
15h30	Machine Learning 3/7 14h45 - 16h15 DONINI Julien ST PPHY 9208 M2 UP DU Data Scientist	Machine Learning. 1/2 13h30 - 16h30 DONINI Julien ST LMV info M2 UP DU Data Scientist	Machine Learning 6/7 14h00 - 15h30 DONINI Julien ST PPHY 9208 M2 UP DU Data Scientist		
16h00					
16h30			Machine Learning 7/7 15h45 - 17h15 DONINI Julien ST PPHY 9208 M2 UP DU Data Scientist		
17h00					



Lecture: 4h30

Lecture: 2h30

Practice: 3h

Practice: 4h

Practice: 3h

Lecture: 3h

References (non exhaustive !) & credits

Classical Machine Learning textbooks

- **Elements of statistical learning** (ESL), [Hastie](#) et al., Springer
- **An Introduction to Statistical Learning** (ISLR), Hastie et al. Springer
 - Both books available online: <http://web.stanford.edu/~hastie/pub.htm>
- **Pattern Recognition and Machine Learning**, [Bishop](#), Springer
- **Deep learning book**, I. Goodfellow et al, <http://www.deeplearningbook.org/>

A *lot* of courses, lectures and tutorial on the web

- **Online courses:** DataCamp, Coursera, Andrew Ng (<http://cs229.stanford.edu/>)
- **CERN lectures** (ex: [Kagan](https://indico.cern.ch/event/619370) <https://indico.cern.ch/event/619370>)
- **2 recommended lectures:**
 - [François Fleuret](https://fleuret.org/ee559/) (EPFL) <https://fleuret.org/ee559/>
 - [Gilles Louppe](https://github.com/glouppe/info8010-deep-learning) (University Liège)<https://github.com/glouppe/info8010-deep-learning>
- **ML cheatsheet:** <https://ml-cheatsheet.readthedocs.io/en/latest/index.html>

Python resources

- A **Crash Course** in Python for Scientists :
<http://nbviewer.jupyter.org/gist/rpmuller/5920182>
- Introduction to **scientific computing** with Python:
<http://github.com/jrjohansson/scientific-python-lectures>
- Python **Tutorial**: <https://www.codecademy.com/tracks/python>

Notebooks basics

- **Installation** (recommended): <https://www.anaconda.com/download>
- **Jupyter** Notebook documentation: <https://jupyter-notebook.readthedocs.io/en/stable/>
- **Interactive** notebooks: <https://mybinder.org/>
- Introduction with **video** tutorial: <https://www.youtube.com/watch?v=Duicsycntdo>

Git

- **Git** documentation: <https://book.git-scm.com/>
- **Github**: <https://github.com/>
- **GitLab** (CERN) basics: <https://gitlab.cern.ch/help/gitlab-basics/start-using-git.md>
- **Tutorial** (in FR): <https://github.com/clr-info/tuto-git>
<https://openclassrooms.com/en/courses/1233741-gerez-vos-codes-source-avec-git>

ML software and interfaces

Internal (HEP) tools

ROOT framework for data storage and processing

Multivariate Analysis: [TMVA](#) for mostly BDT and (deep) NN

Specific for Neural Networks: [NeuroBayes](#)

External tools

Data format: text, csv, images, [HDF5](#), ...

ML libraries: [Keras+TensorFlow](#), [Pytorch](#), [scikit-learn](#) (no DL), ...

All kinds of popular algorithms: CNN, GAN, RNN, LSTM, AE, VAE ...

Interfaces and middleware

PyMVA: Interface TMVA and Keras

Several middleware file format conversion solutions:

[arxiv:1807.02876](#)

PyROOT	Python extension module that allows the user to interact with ROOT data/classes.	69
root_numpy	The interface between ROOT and NumPy supported by the Scikit-HEP community.	65
root_pandas	The interface between ROOT and Pandas dataframes supported by the DIANA/HEP project.	70
uproot	A high throughput I/O interface between ROOT and NumPy.	71
c2numpy	Pure C-based code to convert ROOT data into Numpy arrays which can be used in C/C++ frameworks.	72
root4j	The hep.io.root package contains a simple Java interface for reading ROOT files. This tool has been developed based on freehep-rootio.	73
root2npy	The go-hep package contains a reading ROOT files. This tool has been developed based on freehep-rootio.	73
root2hdf5	Converts ROOT files containing TTrees into HDF5 files containing HDF5 tables.	74

Notebooks

Jupyter notebooks: jupyter.org

The screenshot shows the official Jupyter website. At the top is the Jupyter logo, which consists of a stylized orange and yellow sun-like shape with the word "jupyter" written in lowercase. Below the logo is a navigation bar with links for "Install", "About Us", "Community", and "Documentation". The main content area features a large image of a Jupyter notebook interface. The notebook has a title "Exploring the Lorenz System". It contains some mathematical equations and a plot of the Lorenz attractor. On the left side of the notebook, there's a sidebar with various icons representing different programming languages and tools like Python, R, MATLAB, and others.

Install using Anaconda:
www.anaconda.com

Easily install 1,400+ data science packages for Python/R and manage your packages, dependencies, and environments—all with the single click of a button. Free and open source.

The screenshot shows the Anaconda Distribution landing page. At the top is the Anaconda logo, which is a green circle with a white "A". Next to it is the text "ANAconda DISTRIBUTION" and "Most Trusted Distribution for Data Science". Below this is a green header bar with the text "ANAconda Navigator" and "Desktop Portal to Data Science". Underneath is another green bar with the text "ANAconda Project" and "Portable Data Science Encapsulation". The main content area is a blue section titled "DATA SCIENCE LIBRARIES" which lists various tools: Data Science IDEs (jupyter, spyder, jupyterlab, R Studio), Analytics & Scientific Computing (NumPy, SciPy, Numba, pandas, DASK), Visualization (Bokeh, HoloViews, Datashader, matplotlib), and Machine Learning (TensorFlow, Theano, H2O.ai). At the bottom is a green footer bar with the text "CONDA®" and "Data Science Package & Environment Manager".

Why Over 6 Million Users Love Anaconda Distribution

Scikit-learn (scikit-learn.org)



The screenshot shows the main landing page of scikit-learn.org. At the top left is the logo. To its right are navigation links: Home, Installation, Documentation (with a dropdown arrow), Examples, Google Custom Search, and a Search bar. A "Fork me on GitHub" button is in the top right corner. Below the header is a large blue banner featuring a grid of nine small plots illustrating various machine learning models like Nearest Neighbors, Linear SVM, RBF SVM, Gaussian Process, Decision Tree, Random Forest, and Neural Net. To the right of the banner, the text "scikit-learn" and "Machine Learning in Python" is displayed, followed by a bulleted list of features.

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ...

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ...

— Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ...

— Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, non-negative matrix factorization.

— Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics.

— Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: preprocessing, feature extraction.

— Examples

News

On-going development: What's new (Changelog)

Community

About us See authors and contributing

More Machine Learning Find related projects

Who uses scikit-learn?



Deep Learning libraries

www.tensorflow.org



TensorFlow

[Pytorch.org](https://pytorch.org)

PyTorch

FROM
RESEARCH TO
PRODUCTION

An open source deep learning platform that provides a seamless path from research prototyping to production deployment.

Keras.io

Keras: The Python Deep Learning library



Keras

You have just found Keras.

Keras is a high-level neural networks API, written in Python and capable of running on top of [TensorFlow](#), [CNTK](#), or [Theano](#). It was developed with a focus on enabling fast experimentation. *Being able to go from idea to result with the least possible delay is key to doing good research*.

Get Started

Features

extensibility).
binations of the two.

What is Machine Learning

What ML pioneers say

“Giving computer the ability to **learn without training** them” (Arthur Samuel '59)

“**Learning** is any process by which a **system improves performance** from **experience.**”(Herbert Simon)

“ML is the study of **algorithms** that improve their **performances P** at some **task T** by **Experience E.** A well defined learning task is given by **<P,T,E>**” (T. Mitchell '98)

Traditional Programming



Machine Learning



[P. Domingo]

What is Machine Learning

Based on mathematics, statistics and algorithmics + computer power

- Determine complex **models** from data
- **Prediction and inference**

Machine Learning is not recent

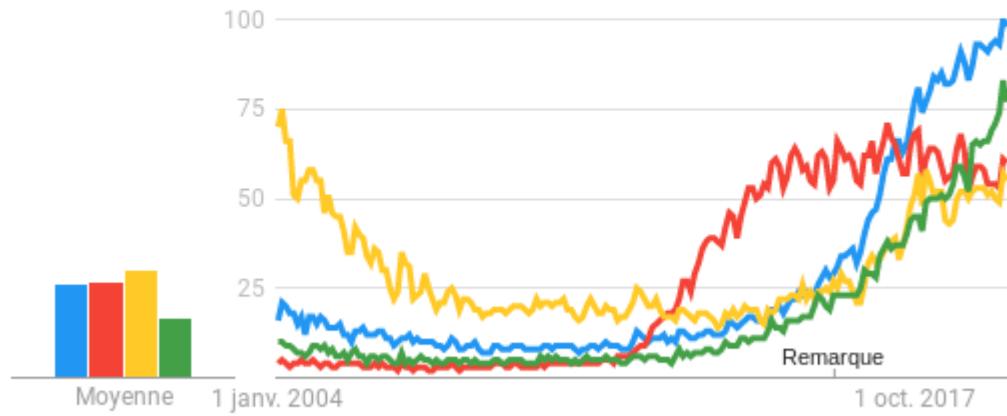
- Artificial **Neural Network** (theory 40's, first functional networks 60's)
- Decision **Trees** (~80's)
- Used in **HEP** since many years – but sometime with scepticism.

Renaissance of the field since ~10 years

- **Deep Learning** (first DNN in HEP [arxiv:2014.4735](https://arxiv.org/abs/1406.2835))
- **Graphics Processing Units** for fast and scalable calculations
- New **recent** algorithms: GAN (2014), Adam minimization (2014), ...

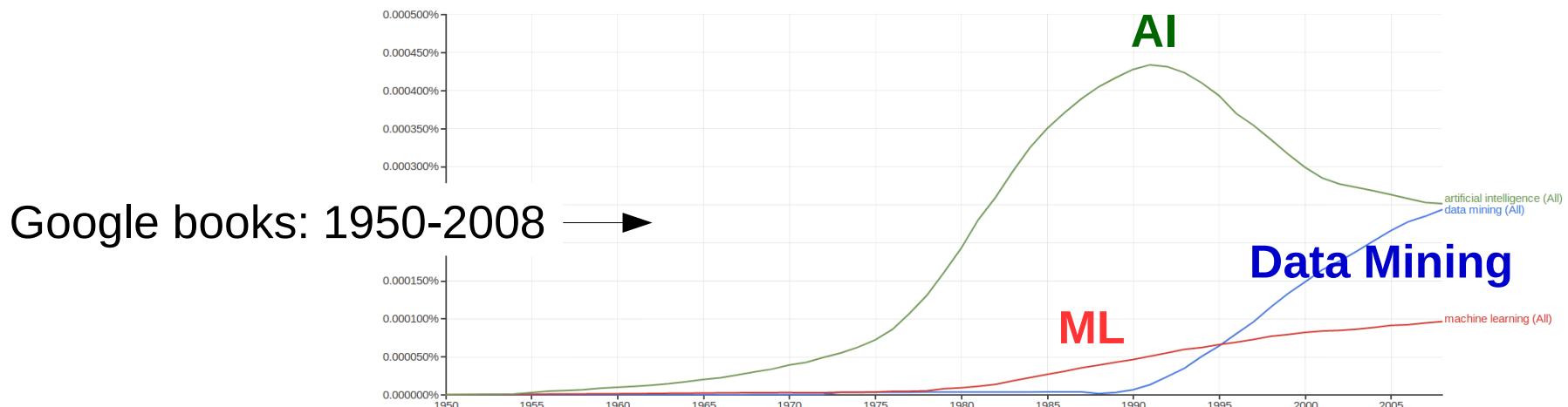
Buzzwords

● machine learning ● big data ● artificial intelligence ● Data science



← Google trends: 2004-2019

Dans tous les pays. 01/01/2004 – 24/10/2019. Recherche sur le Web.



Hype Cycle



gartner.com/SmarterWithGartner

Source: Gartner
© 2019 Gartner, Inc. and/or its affiliates. All rights reserved.

Gartner
Methodology: see [here](#)

Applications

- **Speech and handwriting** recognition
- **Language** processing
- **Image** recognition
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- ...



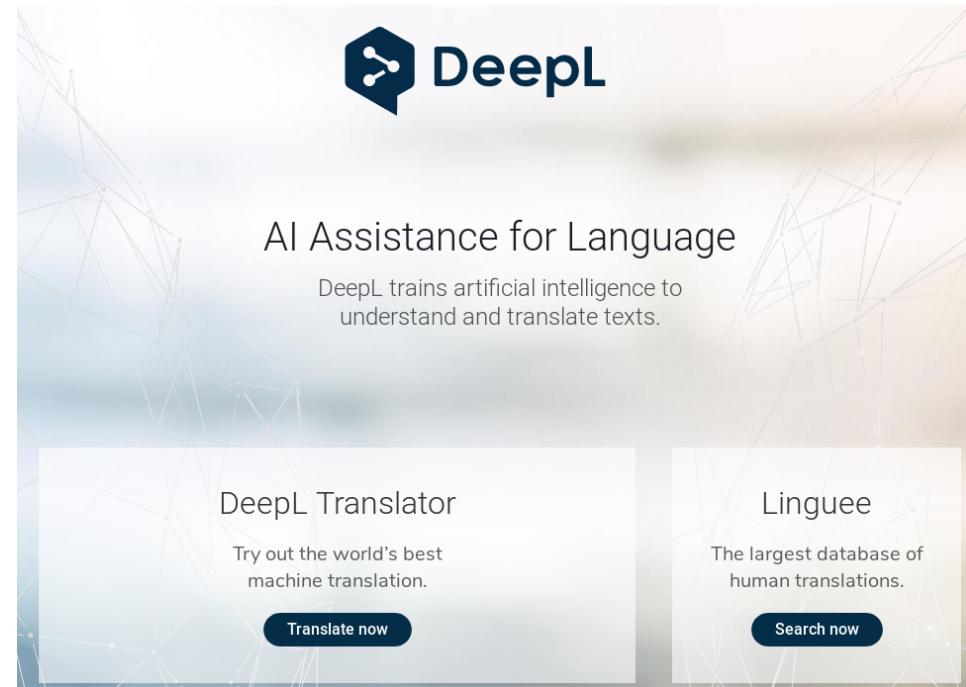
MNIST database
60,000 training images
10,000 testing images.

Human error rate ~ 2%

Best ML error rate: 0.2%

Applications

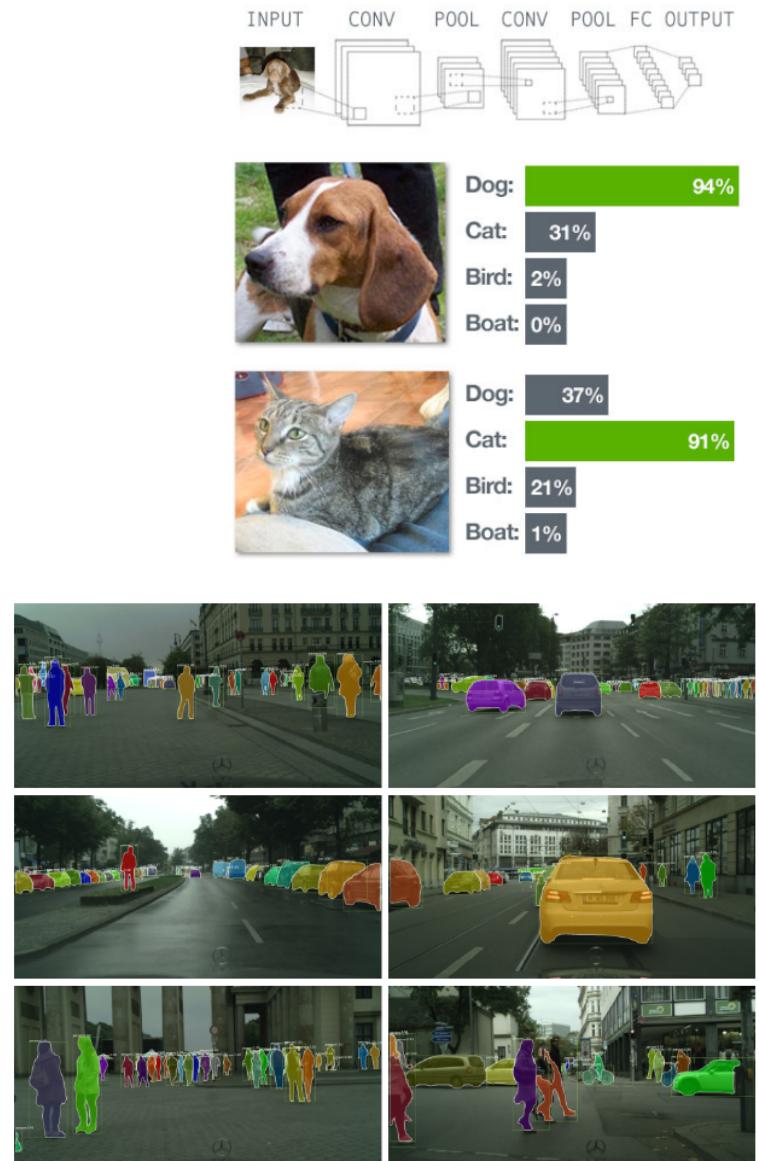
- **Speech and handwriting** recognition
- **Language** processing
- **Image** recognition
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- ...



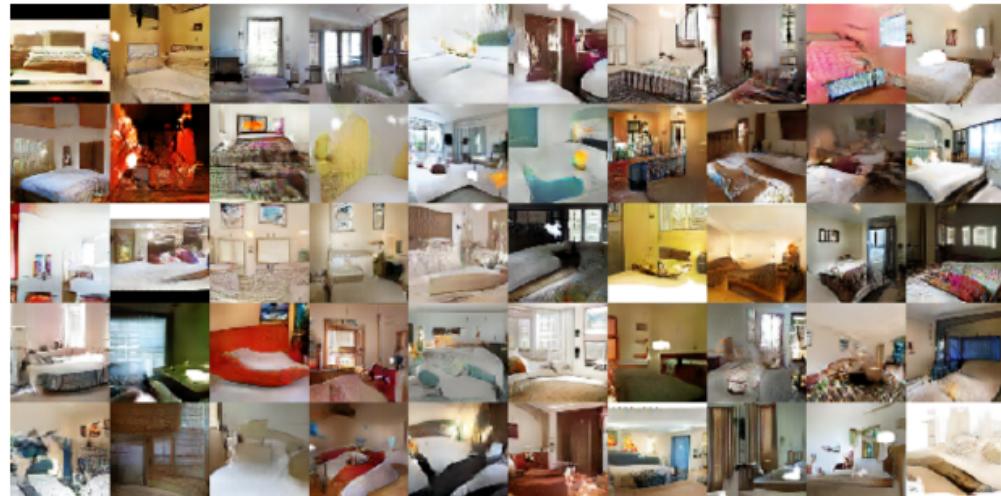
Deep learning translator
<https://www.deepl.com>

Applications

- **Speech and handwriting** recognition
- **Language** processing
- **Image** recognition
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- ...



- **Speech and handwriting** recognition
- **Language** processing
- **Image recognition generation !**
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- ...



(Arjovsky et al, 2017)

Data analysis

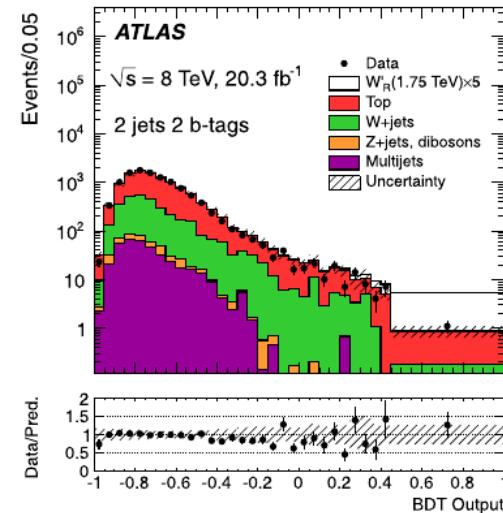
- Precision measurements
- Searches for new physics
- Background rejection
- ...

Performances

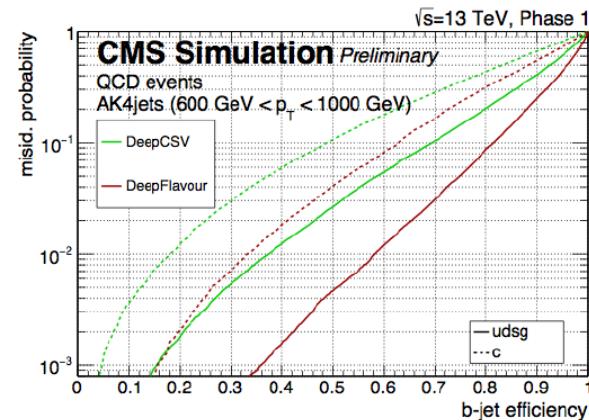
- Trigger and particle identification
- Object reconstruction
- Energy/mass resolution
- Anomaly detection
- ...

Computing

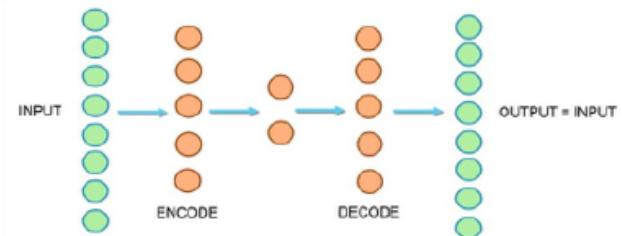
- Best access to popular datasets
- ...



Physics Letters B 743 (2015) 235–255



CMS-DPS-2017-023



Machine Learning Challenge

www.kaggle.com



Higgs Boson Machine Learning Challenge

Use the ATLAS experiment to identify the Higgs boson
Featured · 3 years ago

Homesite
HOME INSURANCE

Homesite Quote Conversion

Which customers will purchase a quoted insurance plan?
Featured · a year ago



Forest Cover Type Prediction

Use cartographic variables to classify forest categories
Playground · 2 years ago

TalkingData

TalkingData Mobile User Demographics

Get to know millions of mobile device users
Featured · 8 months ago

information security
amazon

Amazon.com - Employee Access Challenge

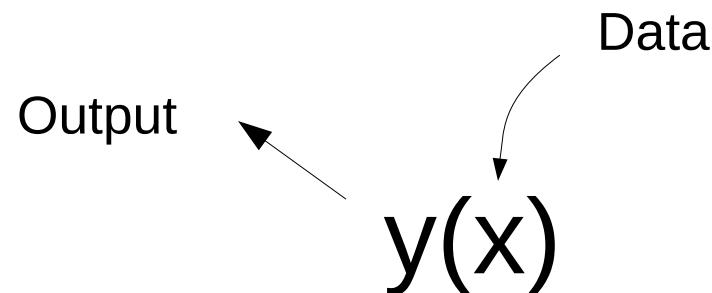
Predict an employee's access needs, given his/her job role
Featured · 4 years ago

Higgs challenge **the HiggsML challenge**
May to September 2014
When **High Energy Physics** meets **Machine Learning**

info to participate and compete : <https://www.kaggle.com/c/higgs-boson>

1800 teams - 2000 participants – 36000 proposed solutions

What is Machine Learning

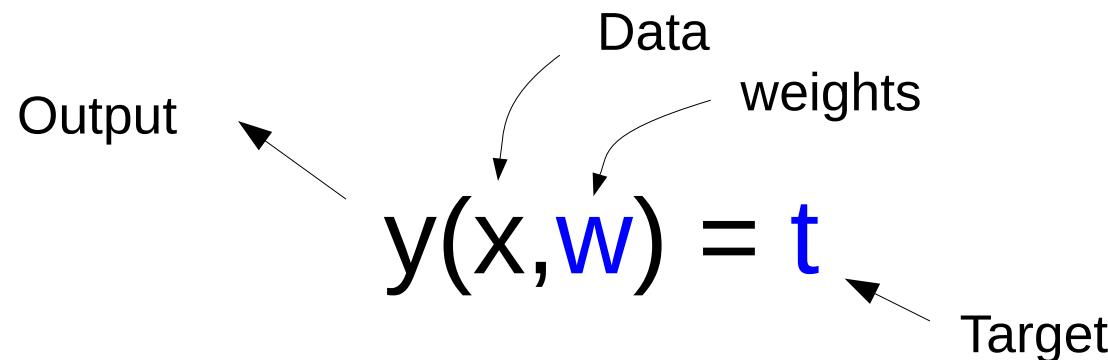


x : input **data** of (multidimensional) variables

$y(x)$: **output** (multidimensional) values

where y is determined by “machine”

What is Machine Learning



Training phase

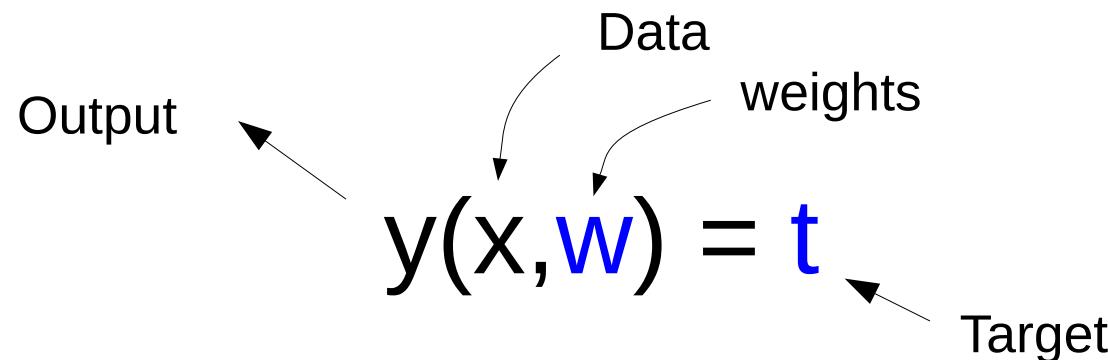
Input data generally consist of a set of:

- x_i : known input **features (or variables)**
 - t_i : known **target values (or label)**
- **Learn $y(x)$ to reproduce t : determine weights w**

Testing

determine y (therefore t) for any new set of x values

What is Machine Learning



Examples

- $X = \{\text{age, year, education, ...}\} \rightarrow \mathbf{t}: \text{income}$
- $X = \{\text{image pixel values}\} \rightarrow \mathbf{t}: \text{face recognition}$
- $X = \{\text{list of words}\} \rightarrow \mathbf{t}: \text{spam detection}$
- $X = \{E, p, \dots\} \rightarrow \mathbf{t}: \text{particle detection}$

...

Supervised learning

- Given: **training data** and **labels** (i.e type of data)
- **Training** and **testing** phases, generalization
 - Regression, classification ...

Unsupervised learning

- Given: **training data** and **no label**
 - Clustering, dimensionality reduction, ...

Semi-supervised learning

- A **mix** of the above, ex: training data + few labels

Variants

- Reinforcement learning (learn by trial and errors)
- Representation learning / Deep learning
- ...

Regression

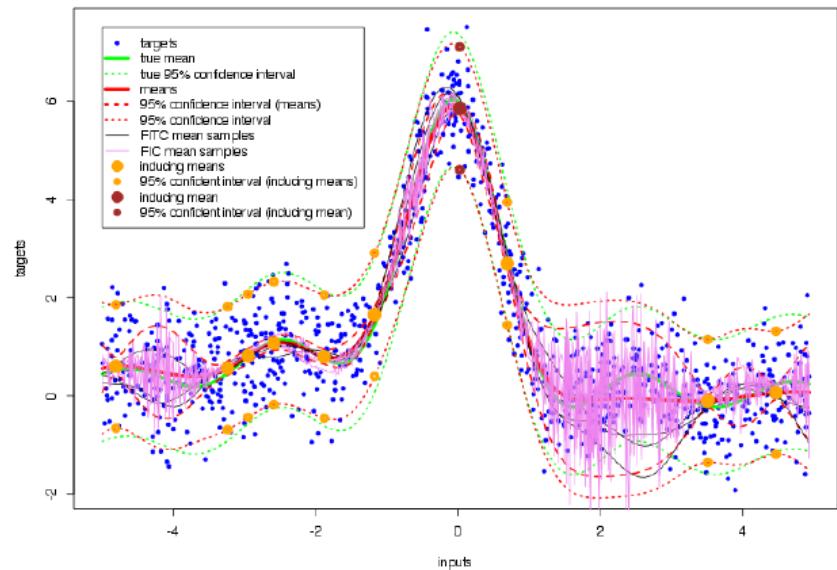
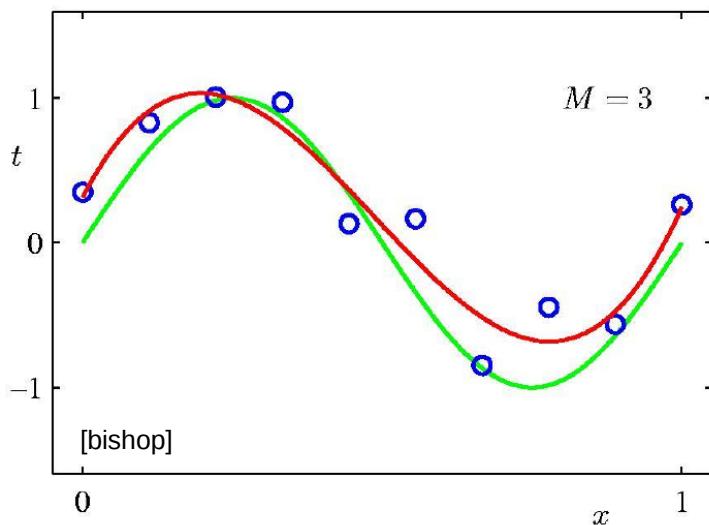
Given **training data** $\{x_i, t_i\}$, learn a function $y(x)$ to predict t given x .

Output y consists of one (or more) **continuous** variables → **Regression**

- Ex: linear regression: data is fit with a **linear** function of **weights**

$$y(x; w_1, \dots, w_M) = \sum_{i=1}^M w_i h_i(x)$$

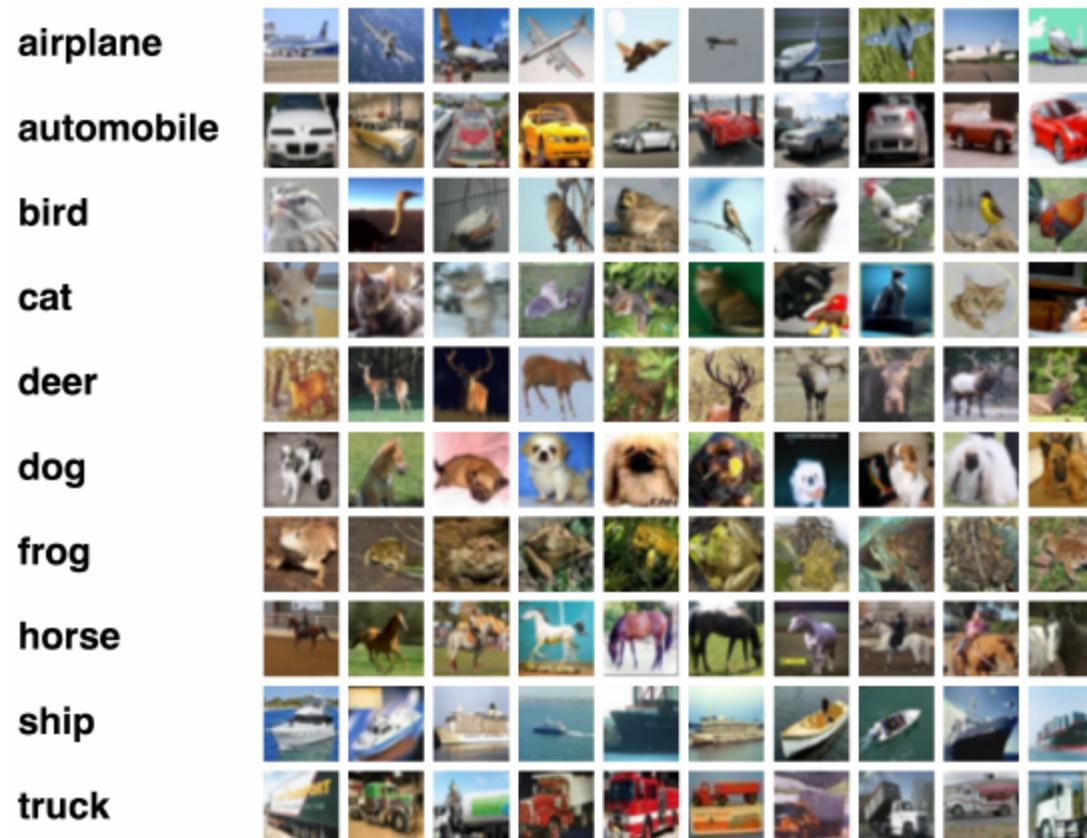
w_i: **weights** (coefficients)
h_i(x): **any** function of x



Classification

Given **training data** $\{x_i, t_i\}$, learn a function $y(x)$ to predict t given x .

Output y consists of one (or more) **categories** → **Classification**



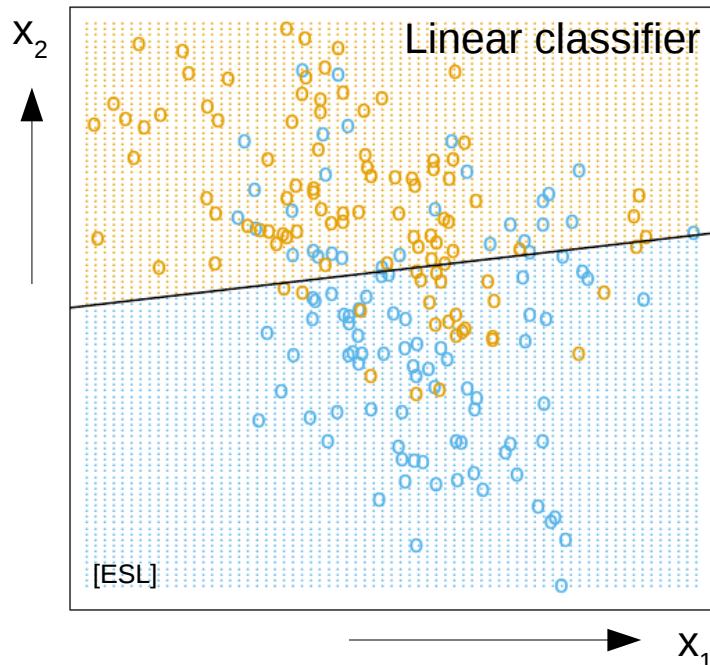
Classification

Given **training data** $\{x_i, t_i\}$, **learn** a function $y(x)$ to predict t given x .

Output y consists of one (or more) **categories** → **Classification**

Example (2D):

Data coded as a binary variable (**BLUE** = 0, **ORANGE** = 1), and then fit by a function.



$$\begin{cases} y(x_1, x_2) > 0.5 \rightarrow \text{ORANGE} \\ y(x_1, x_2) < 0.5 \rightarrow \text{BLUE} \end{cases}$$

Boundary: $y(x_1, x_2) = 0.5$

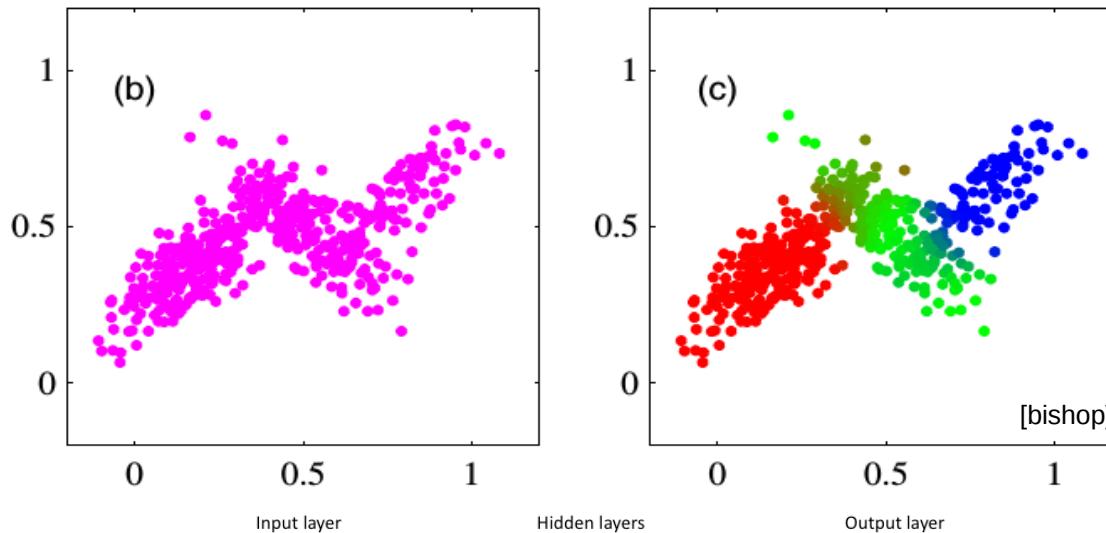
Orange shaded region: space classified as **ORANGE**

Blue region: space classified as **BLUE**.

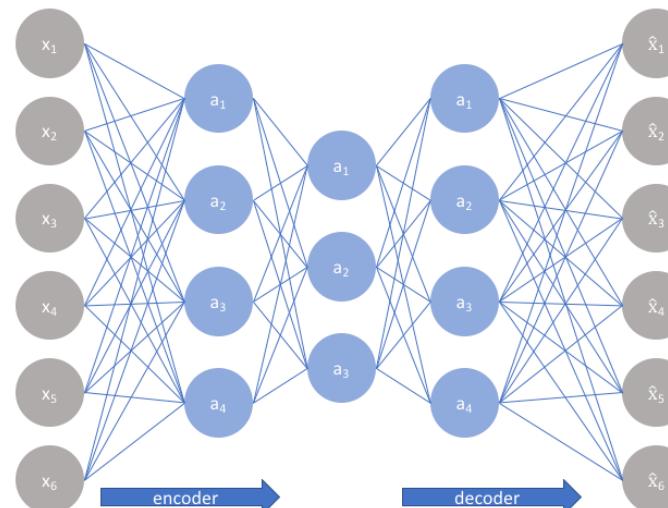
Unsupervised learning

Unsupervised learning = no labels

Clustering

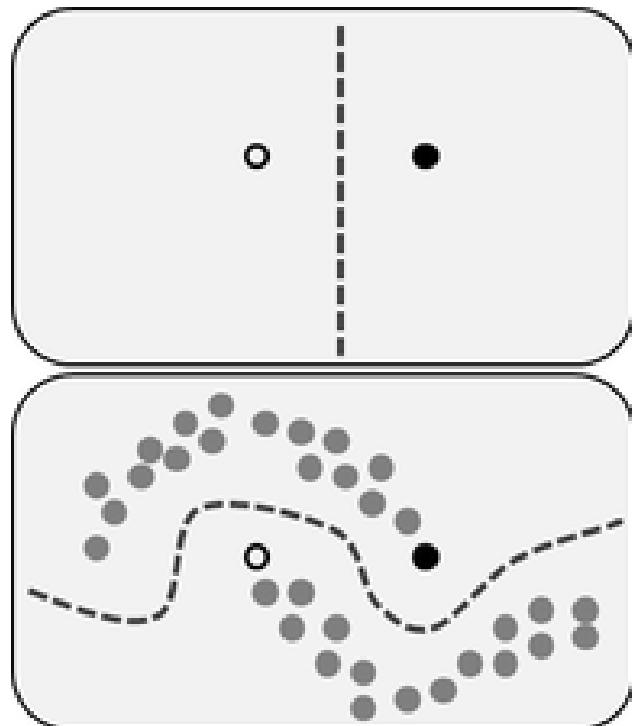


Dimensionality reduction



Semi-supervised learning

Semi-supervised learning = unlabelled data + few labels



Example of the influence of unlabelled data in semi-supervised learning.

The unlabelled data (grey dots) influence the separation of the two classes (decision surface)

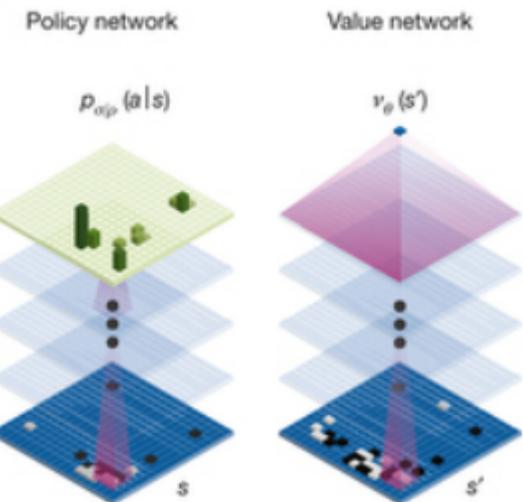
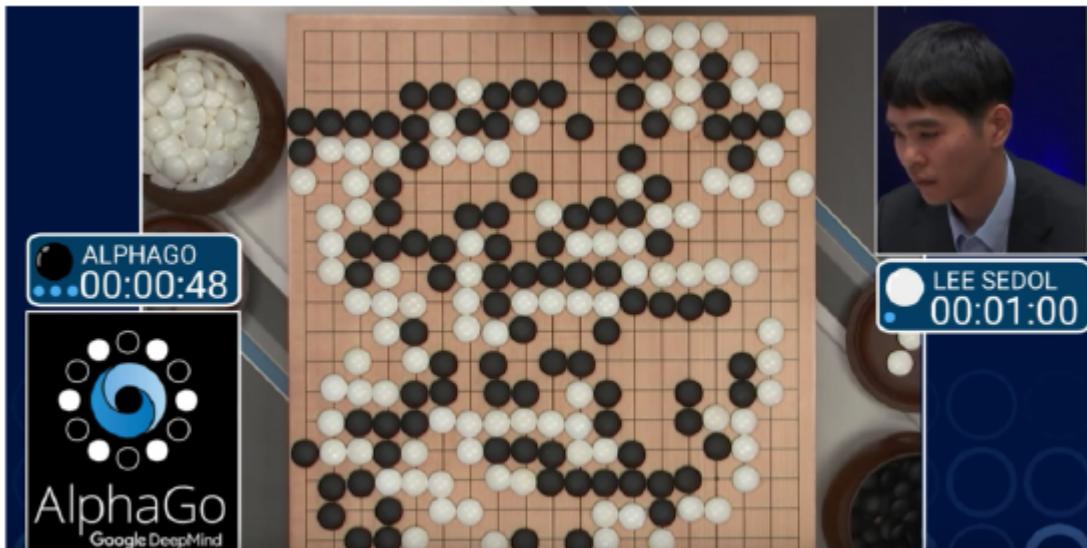
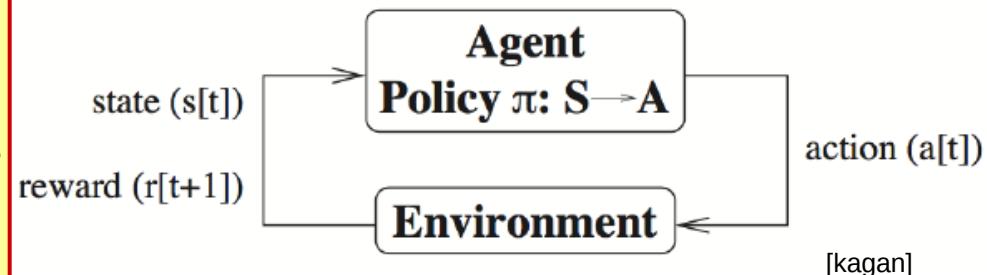
[wikipedia]

Reinforcement learning

Model for **agents** that take **actions** depending on current **state**

Action imply **reward** and modify state

Learn to make best sequences of decisions.



Nature 529, 484–489 (28 January 2016)

Representation learning

Representation is how we present the information (data) to solve a problem

Example: what is the best number representation to perform this division ?

Decimal 121 : 11

Roman CXXI / XI

Binary 1111001 : 1011

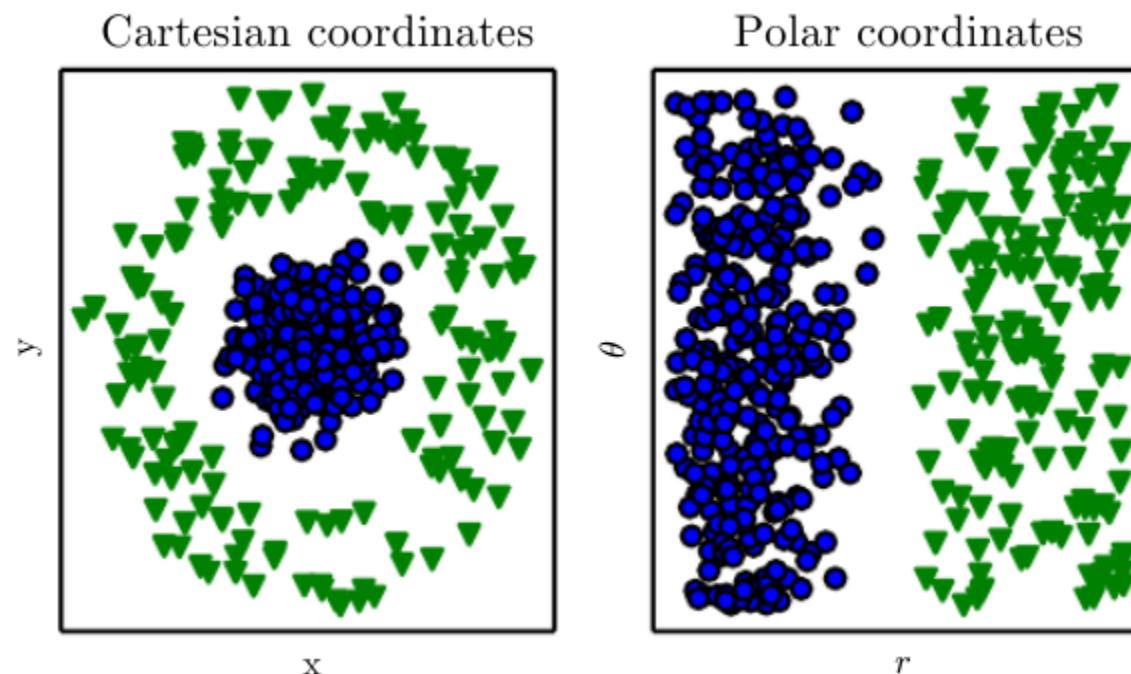
Hex 79 : B

ASCII y : VT

Representation learning

Representation is how we present the information (data) to solve a problem

Example: what are the best coordinates to separate this data ?



[deeplearningbook]

Representation learning

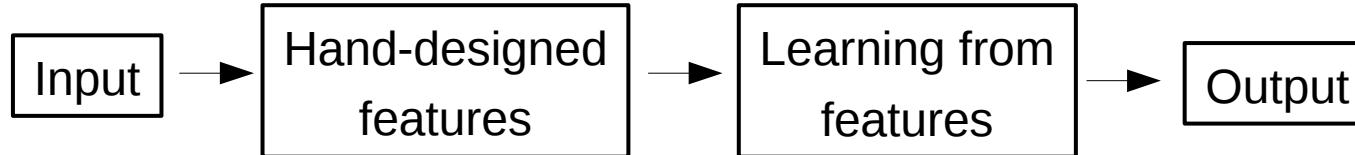
Representation is how we present the information (data) to solve a problem

For ML a good representation is one that makes the **learning task easier**

ML algorithms can also learn best representation: **representation learning**

→ Central to **deep learning** : learn complex representation from simpler ones

Classic ML

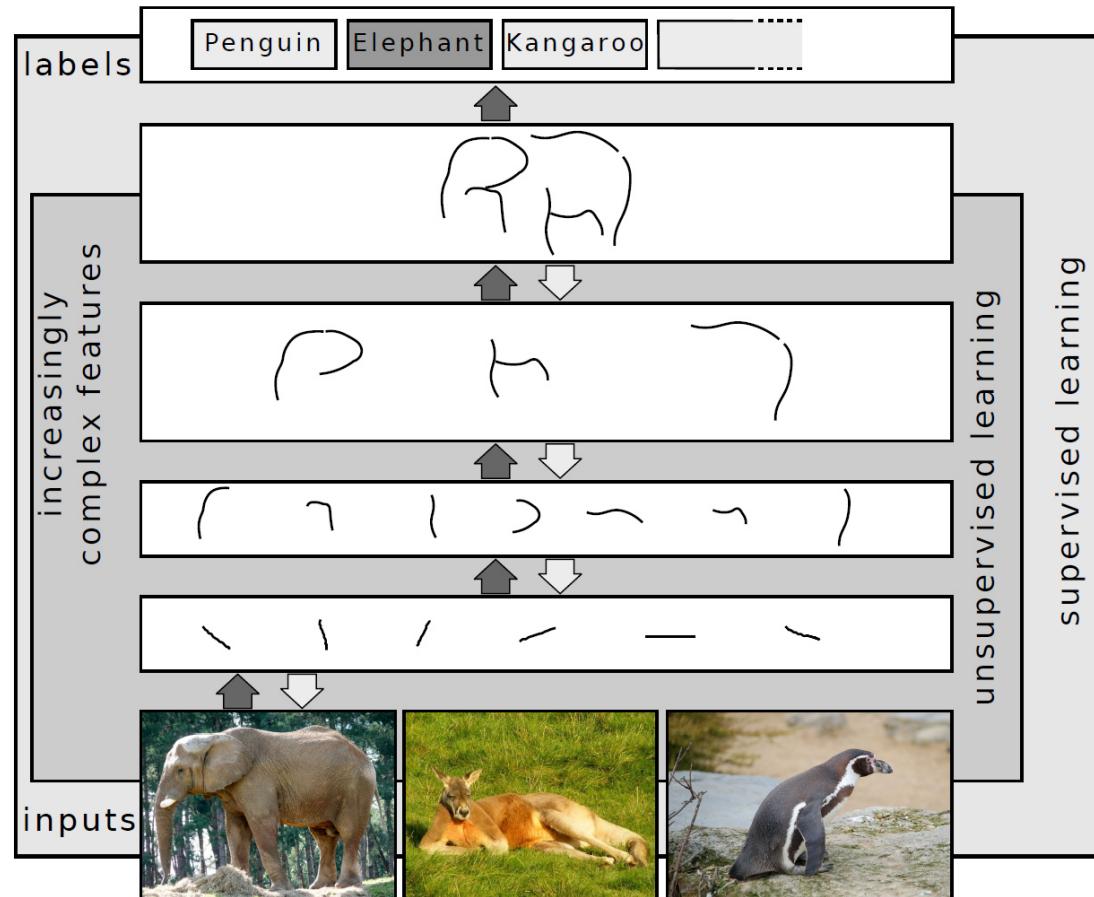


Deep learning



Deep learning

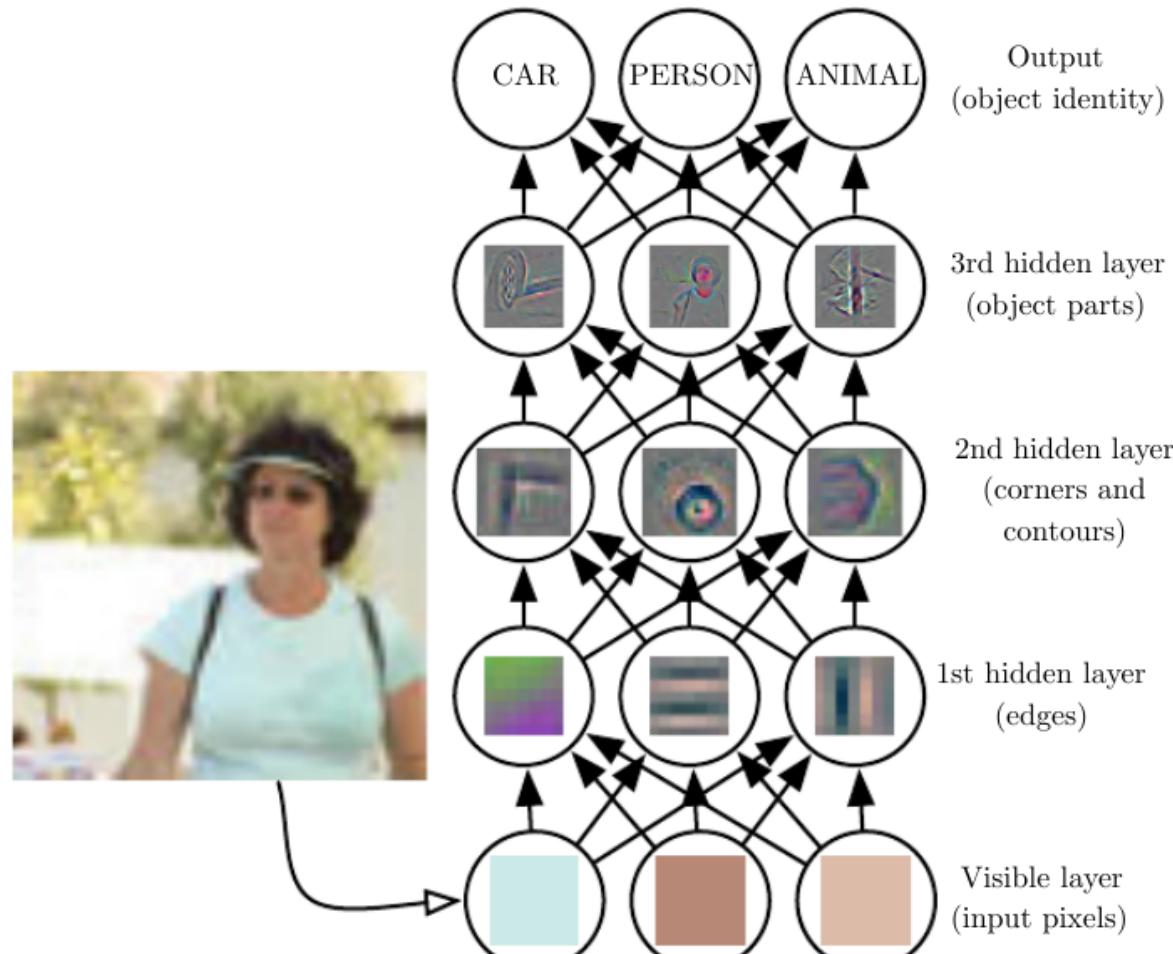
Class of **ML algorithms** (in general artificial neural networks) that use **multiple layer** to **extract higher level features** from raw data



[wikipedia]

Deep learning

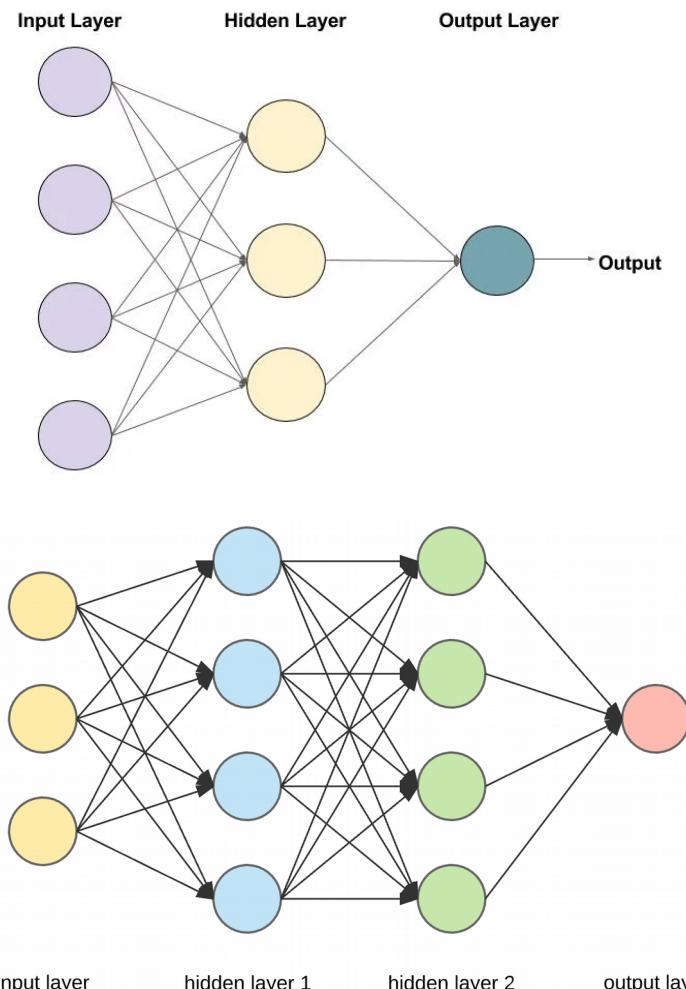
Class of **ML algorithms** (in general artificial neural networks) that use **multiple layer** to **extract higher level features** from raw data



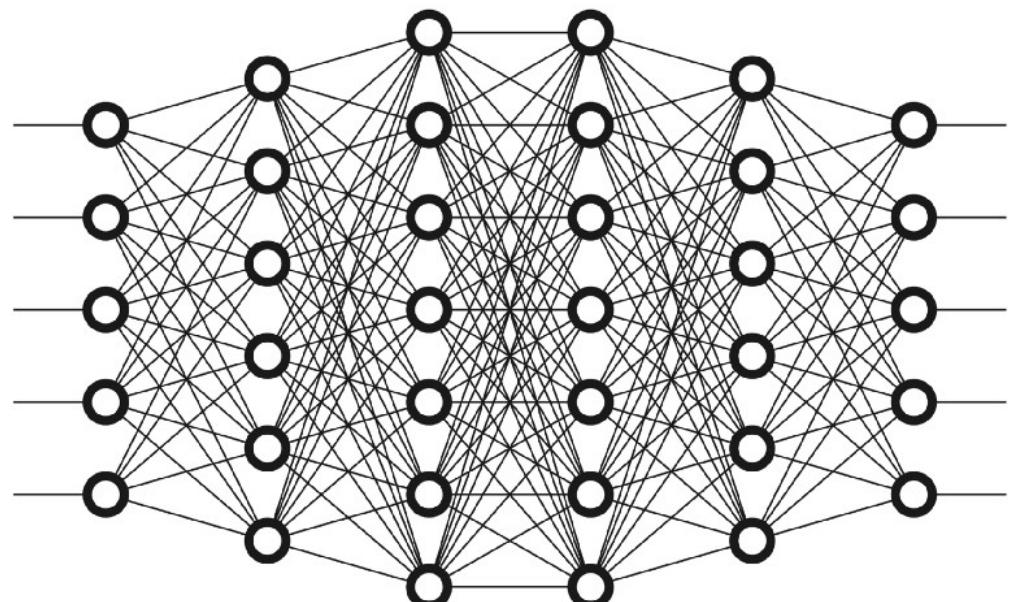
[deeplearningbook]

Shallow Neural Networks

(1 or 2 hidden layers)

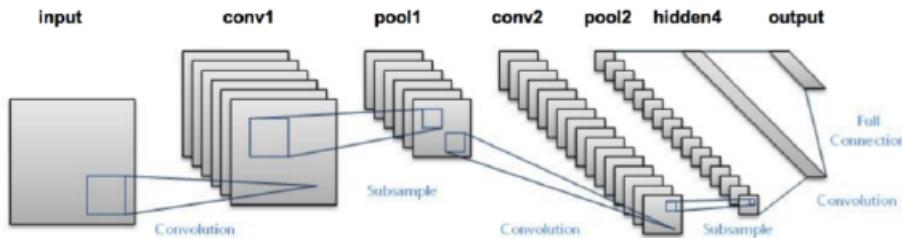


Deep Neural Network
(more layers)

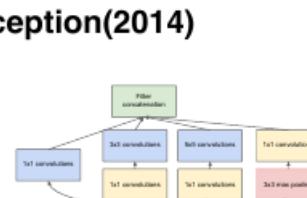


Deep Learning

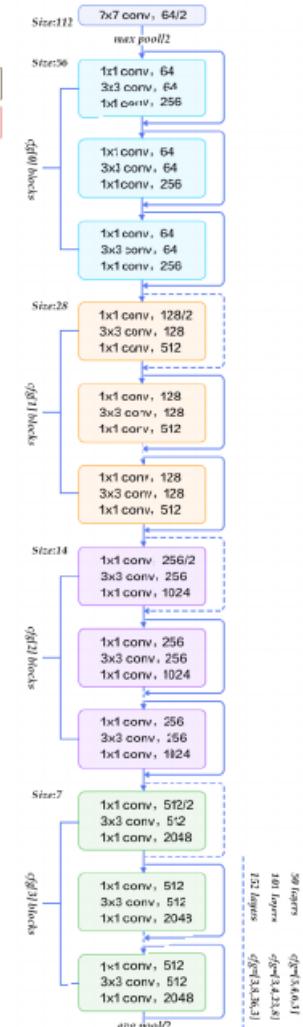
LeNet-5 (1998)



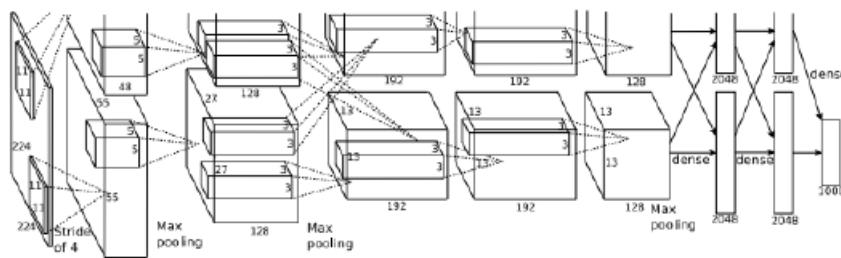
GoogleNet/Inception(2014)



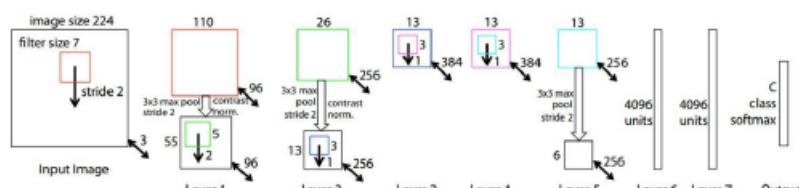
ResNet(2015)



AlexNet (2012)

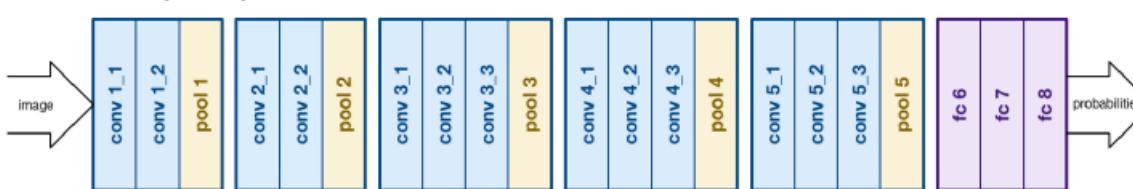


ZFNet(2013)



22 Layers

VGGNet (2014)



152 Layers

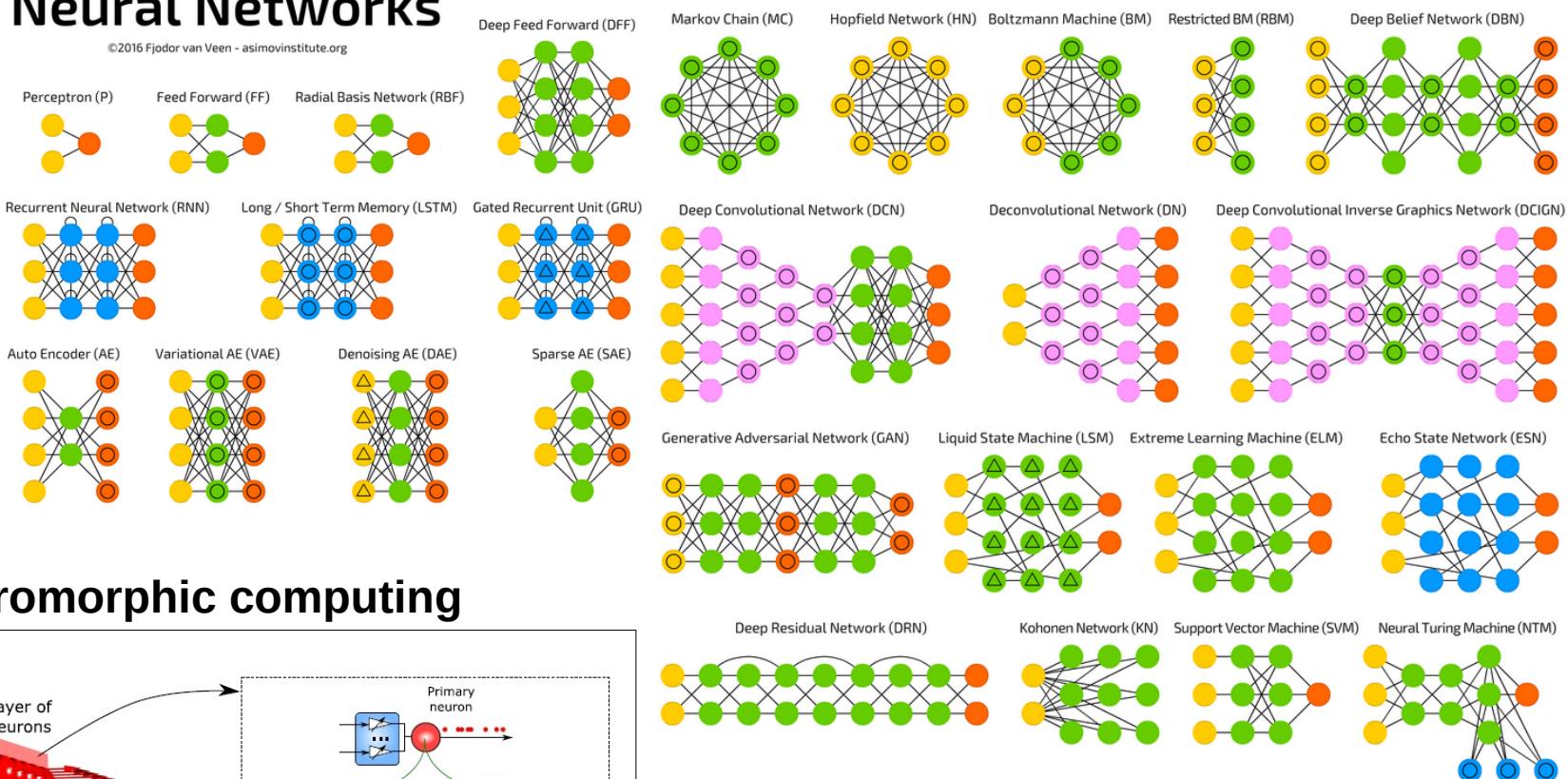
<https://medium.com/@sidereal/cnns-architectures-leenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5>

Neural Networks today

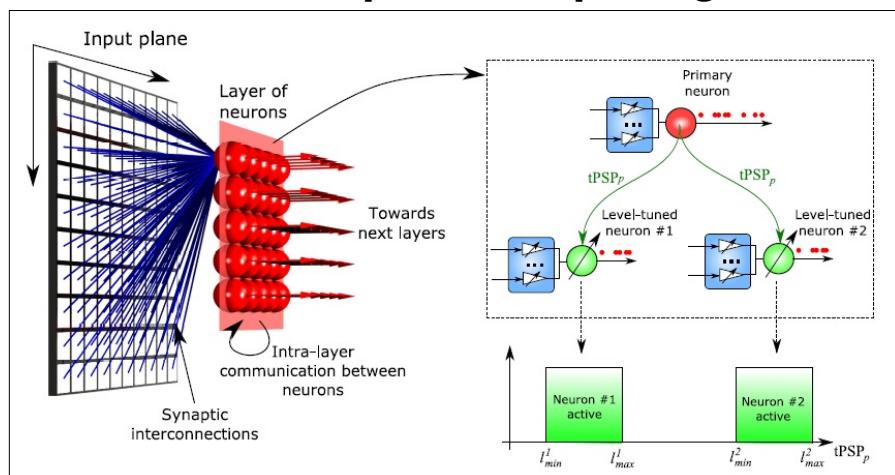
A mostly complete chart of Neural Networks

- Backfed Input Cell
- Input Cell
- △ Noisy Input Cell
- Hidden Cell
- Probabilistic Hidden Cell
- △ Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- △ Different Memory Cell
- Kernel
- Convolution or Pool

©2016 Fjodor van Veen - asimovinstitute.org

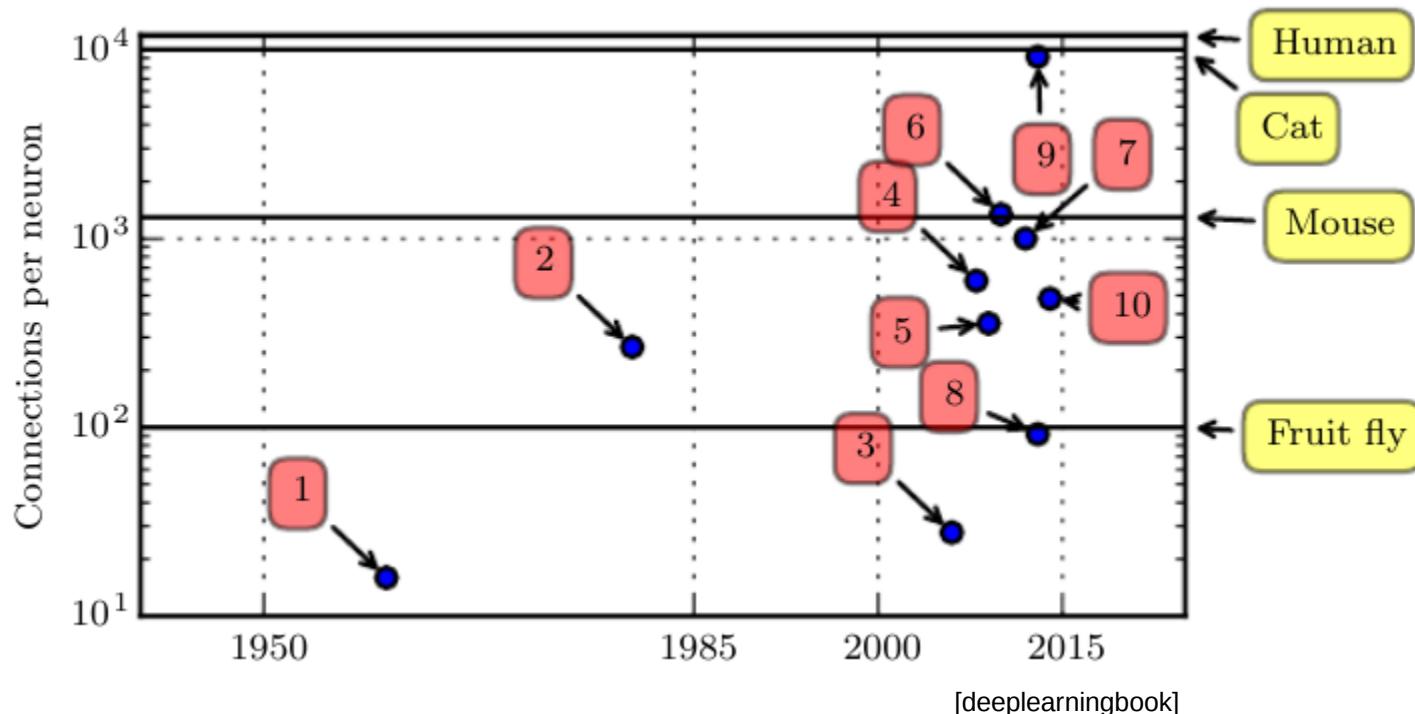


Neuromorphic computing



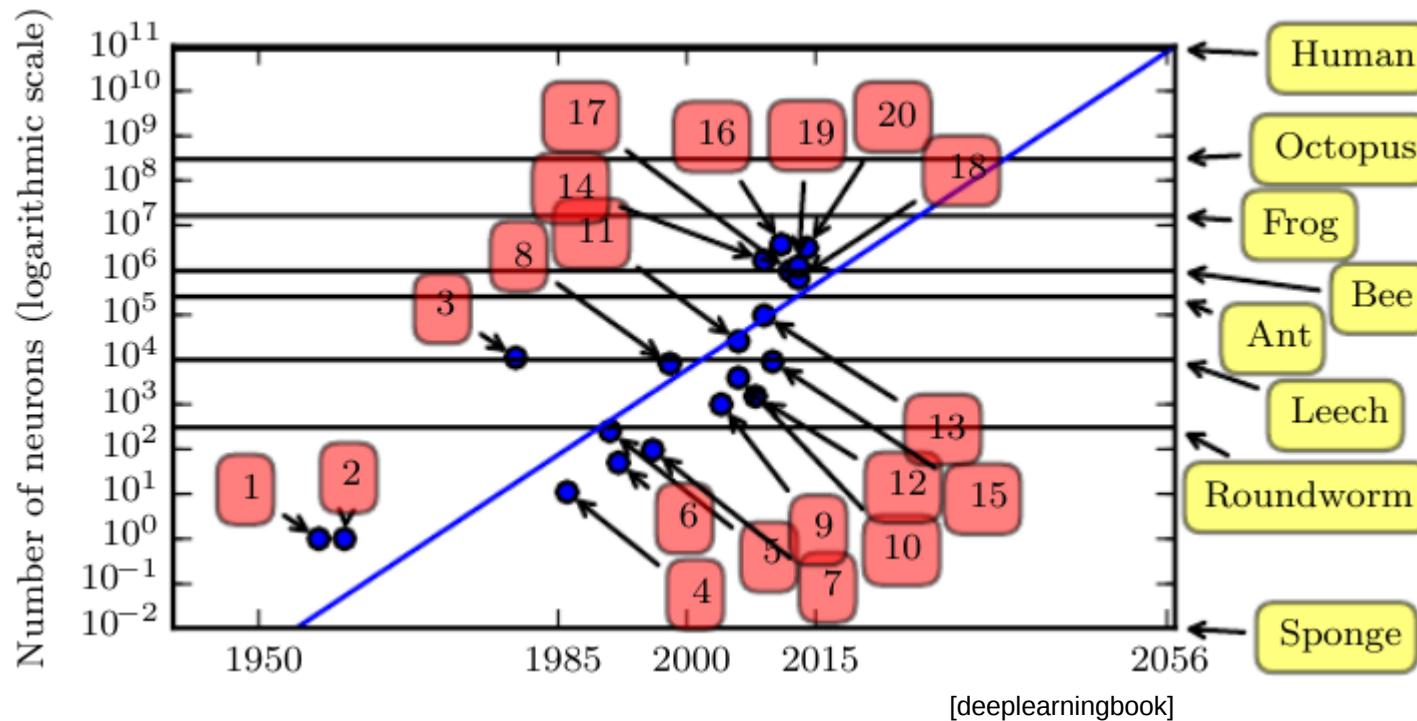
<http://www.asimovinstitute.org/neural-network-zoo/>

Connections per neuron



1. Adaptive linear element (Widrow and Hoff, 1960)
2. Neocognitron (Fukushima, 1980)
3. GPU-accelerated convolutional network (Chellapilla et al., 2006)
4. Deep Boltzmann machine (Salakhutdinov and Hinton, 2009a)
5. Unsupervised convolutional network (Jarrett et al., 2009)
6. GPU-accelerated multilayer perceptron (Ciresan et al., 2010)
7. Distributed autoencoder (Le et al., 2012)
8. Multi-GPU convolutional network (Krizhevsky et al., 2012)
9. COTS HPC unsupervised convolutional network (Coates et al., 2013)
10. GoogLeNet (Szegedy et al., 2014a)

Number of neurons



1. Perceptron (Rosenblatt, 1958, 1962)

2. Adaptive linear element (Widrow and Hoff, 1960)

3. Neocognitron (Fukushima, 1980)

4. Early back-propagation network (Rumelhart et al., 1986b)

5. Recurrent neural network for speech recognition (Robinson and Fallside, 1991)

6. Multilayer perceptron for speech recognition (Bengio et al., 1991)

7. Mean field sigmoid belief network (Saul et al., 1996)

8. LeNet-5 (LeCun et al., 1998b)

9. Echo state network (Jaeger and Haas, 2004)

10. Deep belief network (Hinton et al., 2006)

11. GPU-accelerated convolutional network (Chellapilla et al., 2006)

12. Deep Boltzmann machine (Salakhutdinov and Hinton, 2009a)

13. GPU-accelerated deep belief network (Raina et al., 2009)

14. Unsupervised convolutional network (Jarrett et al., 2009)

15. GPU-accelerated multilayer perceptron (Ciresan et al., 2010)

16. OMP-1 network (Coates and Ng, 2011)

17. Distributed autoencoder (Le et al., 2012)

18. Multi-GPU convolutional network (Krizhevsky et al., 2012)

19. COTS HPC unsupervised convolutional network (Coates et al., 2013)

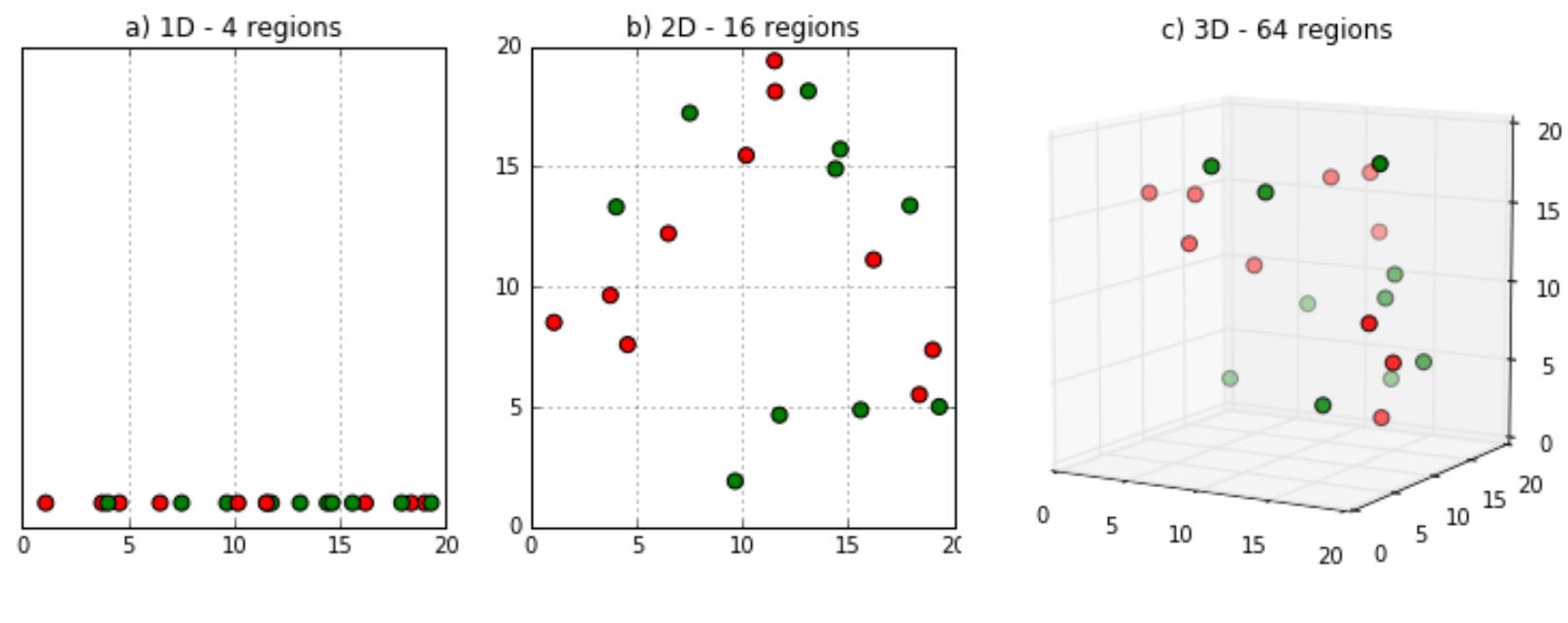
20. GoogLeNet (Szegedy et al., 2014a)

Limitations and difficulties

Deep learning algorithms are very efficient in many domains (in particular image recognition), but they require **a lot of data to train** because of large number of **dimensions** and high number of **hyperparameters**.

Illustration: the “curse of dimensionality”

Simple example: classify each region depending on majority of labels



As dimensions grows, dimensions space increases exponentially.
Classification ok for ~2 variables, but fails at higher dimensions !

Limitations and difficulties

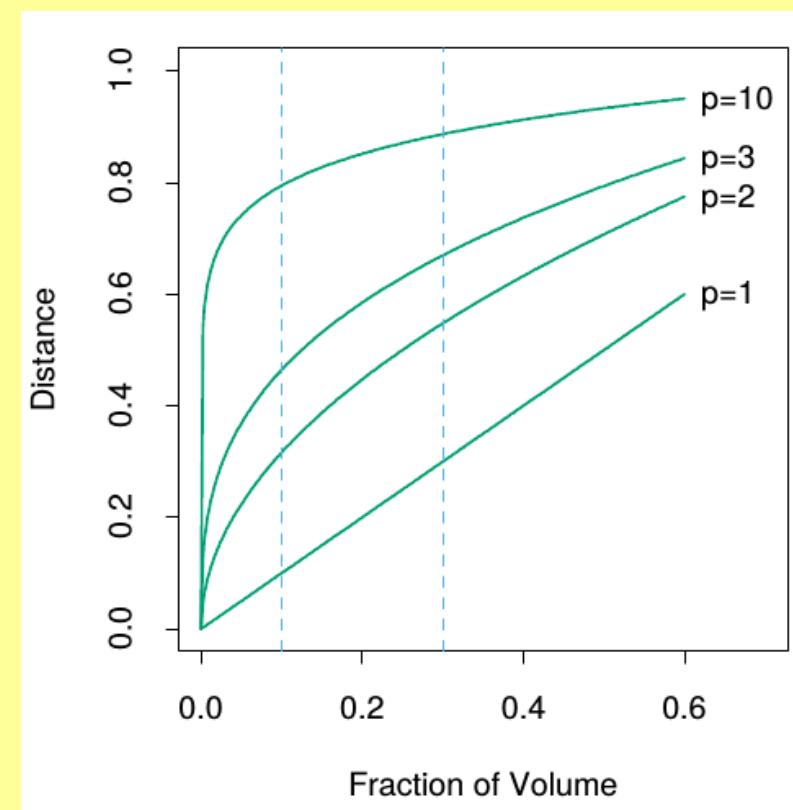
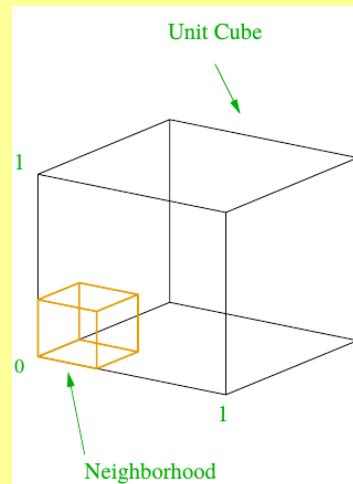
Deep learning algorithms are very efficient in many domains (in particular image recognition), but they require **a lot of data to train** because of large number of **dimensions** and high number of **hyperparameters**.

Illustration: the “curse of dimensionality”

In a hypercube of length 1 and dimension p a fraction r of the volume corresponds to an edge length $e_p(r) = r^{1/p}$

- $p=10$ and $r=1\%$: $e_{10}(0.01)=0.63$
- $p=10$ and $r=10\%$: $e_{10}(0.1)=0.80$

To contain 1% or 10% of the volume we must cover 63% or 80%, respectively, of the range of each input variable !

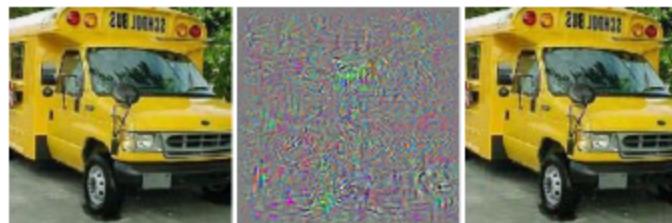


Limitations and difficulties

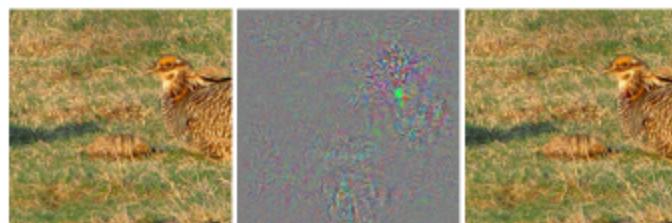
Algorithms get so **complex** that it is difficult to **interpret** what they really do !

Also their complexity can become a **weakness/threat**:

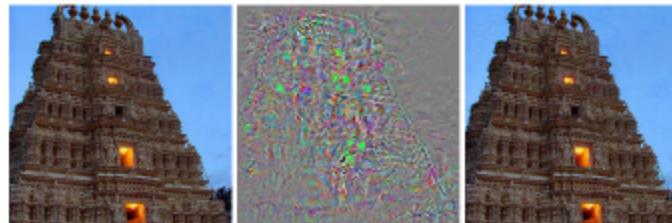
School bus



Bird
(partridge)



Temple



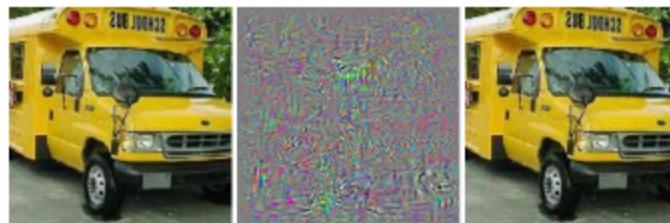
[arxiv:1312.6199]

Limitations and difficulties

Algorithms get so **complex** that it is difficult to **interpret** what they really do !

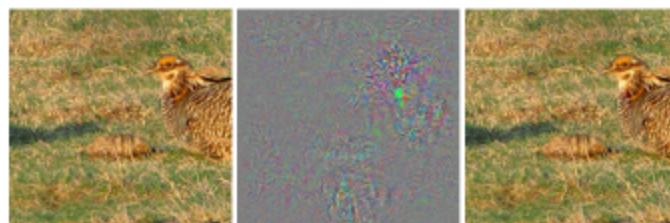
Also their complexity can become a **weakness/threat**:

School bus



Ostrich !

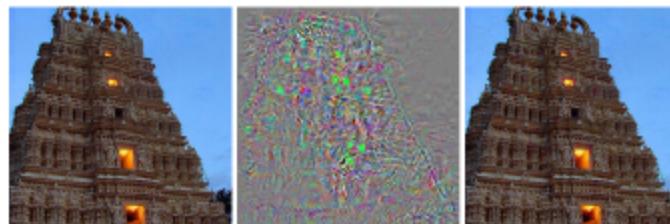
Bird
(partridge)



Ostrich !

?!


Temple



Ostrich !

Perturbation


[arxiv:1312.6199]

Limitations and difficulties

Algorithms get so **complex** that it is difficult to **interpret** what they really do !

Also their complexity can become a **weakness/threat**:



x
“panda”
57.7% confidence

+ .007 ×



$\text{sign}(\nabla_x J(\theta, x, y))$
“nematode”
8.2% confidence

=



$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$
“gibbon”
99.3 % confidence

[arxiv:1412.6572]

Machine Learning is based on statistics and computing (and lots of data !)

Not new ('40s) but field in **exponential evolution** since 10 years

Today: huge amount of **resources and tools** on the web

However real **understanding** of ML requires some efforts

Worth it because ML is becoming evermore present in our life's