# Machine Learning Notes

## From artificial intelligence to physics

Stefano Carrazza

NNPDF/N3PDF meeting,

16-19 September 2018, Palazzo Feltrinelli, Gargnano

European Organization for Nuclear Research (CERN)

Acknowledgement: This project has received funding from HICCUP ERC Consolidator grant (614577) and by the European Unions Horizon 2020 research and innovation programme under grant agreement no. 740006.

N3PDF Machine Learning • PDFs • QCD

CERN

# Why talk about machine learning?

## Why talk about machine learning?

*because*

- it is an essential set of algorithms for building models in science,
- fast development of new tools and algorithms in the past years,
- nowadays it is a requirement in experimental and theoretical physics,
- large interest from the HEP community: *IML, conferences, grants*.

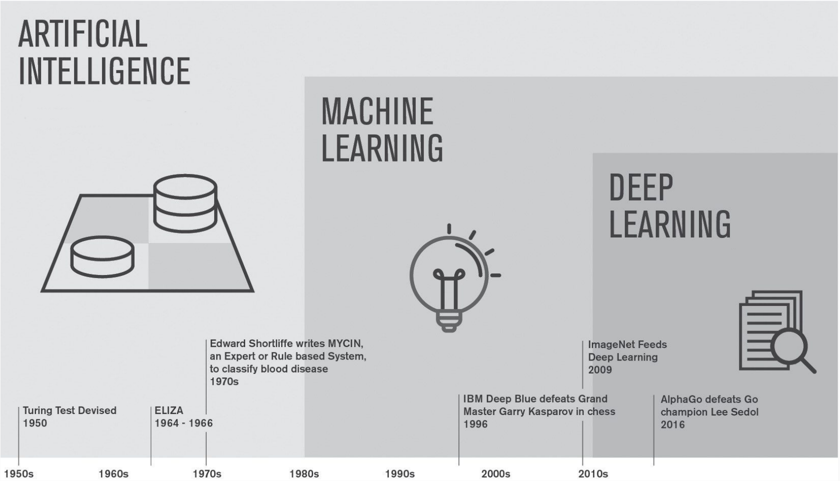
# Outline

**Topics:**

- A.I. and M.L. overview
- Non-linear models
- From physics to ML

# Artificial Intelligence

# Artificial intelligence timeline

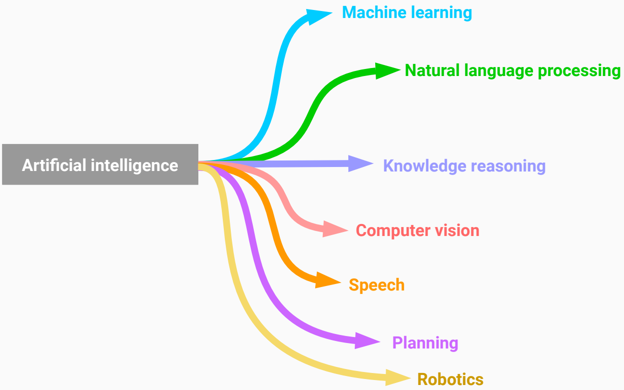


# Defining A.I.

**Artificial intelligence** (A.I.) is *the science and engineering of making intelligent machines*. (John McCarthy '56)

# Defining A.I.

**Artificial intelligence** (A.I.) is *the science and engineering of making intelligent machines*. (John McCarthy '56)

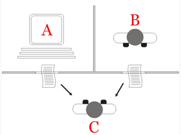



A.I. consist in the development of computer systems to perform tasks commonly associated with intelligence, such as *learning*.

# A.I. and humans

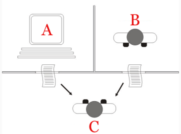
There are two categories of A.I. tasks:

- **abstract and formal:** easy for computers but difficult for humans, *e.g.* play chess (IBM's Deep Blue 1997).
  $\rightarrow$ *Knowledge-based* approach to artificial intelligence.




There are two categories of A.I. tasks:

- **abstract and formal:** easy for computers but difficult for humans, *e.g.* play chess (IBM's Deep Blue 1997).
  $\rightarrow$ *Knowledge-based* approach to artificial intelligence.




- **intuitive for humans but hard to describe formally:** *e.g.* recognizing faces in images or spoken words.
  $\rightarrow$ *Concept* capture and generalization


# A.I. technologies

Historically, the *knowledge-based* approach has not led to a major success with intuitive tasks for humans, because:

- requires human *supervision* and hard-coded *logical inference rules*.
- lacks of *representation learning* ability.

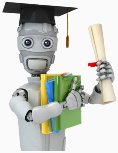
# A.I. technologies

Historically, the *knowledge-based* approach has not led to a major success with intuitive tasks for humans, because:

- requires human *supervision* and hard-coded *logical inference rules*.
- lacks of *representation learning* ability.

**Solution:**

The A.I. system needs to *acquire its own knowledge*.
This capability is known as **machine learning** (ML).
$\rightarrow$ *e.g.* write a program which learns the task.



# Machine Learning

# Machine learning definition

**Definition from A. Samuel in 1959:**
Field of study that gives computers the ability to learn without being explicitly programmed.

# Machine learning definition

**Definition from A. Samuel in 1959:**

Field of study that gives computers the ability to learn without being explicitly programmed.

**Definition from T. Mitchell in 1998:**

A computer program is said to *learn* from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance on $T$, as measured by $P$, improves with experience $E$.

# ML applications in our “day life”

# Machine learning examples

Thanks to work in A.I. and new capability for computers:

- **Database mining:**
    - Search engines
    - Spam filters
    - Medical and biological records

# Machine learning examples

Thanks to work in A.I. and new capability for computers:

- **Database mining:**
  - Search engines
  - Spam filters
  - Medical and biological records
- **Intuitive tasks for humans:**
  - Autonomous driving
  - Natural language processing
  - Robotics (reinforcement learning)
  - Game playing (DQN algorithms)

# Machine learning examples

Thanks to work in A.I. and new capability for computers:

- **Database mining:**
  - Search engines
  - Spam filters
  - Medical and biological records
- **Intuitive tasks for humans:**
  - Autonomous driving
  - Natural language processing
  - Robotics (reinforcement learning)
  - Game playing (DQN algorithms)

# Machine learning examples

Thanks to work in A.I. and new capability for computers:

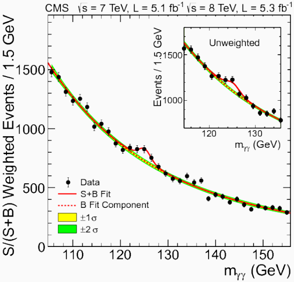
- **Database mining:**
  - Search engines
  - Spam filters
  - Medical and biological records
- **Intuitive tasks for humans:**
  - Autonomous driving
  - Natural language processing
  - Robotics (reinforcement learning)
  - Game playing (DQN algorithms)
- **Human learning:**
  - Concept/human recognition
  - Computer vision
  - Product recommendation

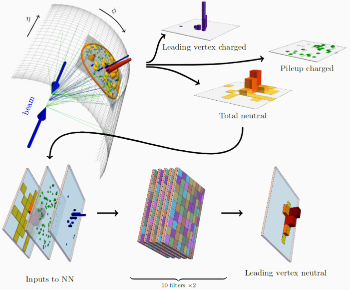
# ML applications in HEP

# ML in experimental HEP

Some remarkable examples are:

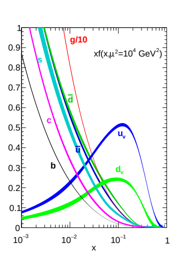
- **Signal-background detection:**
  Decision trees, artificial neural networks, support vector machines.
- **Jet discrimination:**
  Deep learning imaging techniques via convolutional neural networks.
- **HEP detector simulation:**
  Generative adversarial networks, *e.g.* LAGAN and CaloGAN.

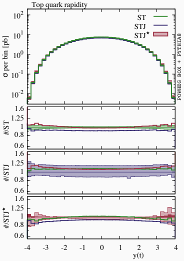


# ML in theoretical HEP

- **Supervised learning:**
  - The structure of the proton at the LHC
    - parton distribution functions
  - Theoretical prediction and combination
  - Monte Carlo reweighting techniques
    - neural network Sudakov
  - BSM searches and exclusion limits
- **Unsupervised learning:**
  - Clustering and compression
    - PDF4LHC15 recommendation
  - Density estimation and anomaly detection
    - Monte Carlo sampling



## Social experiment:
## ML applied to NNPDF meetings (JI & SC)

# ML for Gargnano 2018

**Goal:**

- Extract research topics from participants
- Classify and cluster people
- Propose matches

**How:**

- Unsupervised learning and natural language processing

# ML for Gargnano 2018

**Goal:**

- Extract research topics from participants
- Classify and cluster people
- Propose matches

**How:**

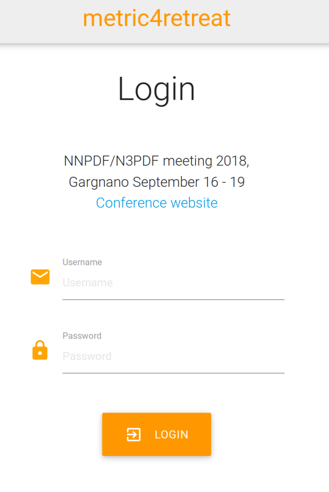
- Unsupervised learning and natural language processing

**Disclaimer:** This is a proof of concept, made in 5 min yesterday...

## ML for Gargnano 2018

You are invited to visit the results:

`http://apfel.mi.infn.it:7000`




- **username:** nnpdf
- **password:** 2018

**Create bag of words:**

- For each participant in the indico page (22 people):
  1. **Download** all **papers** from **arXiv** (total 1514 papers)
  2. Extract **title** and **abstract** of each paper
  3. Perform **tokenization**
  4. Remove **stop words**
  5. Perform **stemming**
- Generated **dictionary** with **4051** unique tokens

# http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

**Create bag of words:**

- For each participant in the indico page (22 people):
    1. **Download** all **papers** from **arXiv** (total 1514 papers)
    2. Extract **title** and **abstract** of each paper
    3. Perform **tokenization**
    4. Remove **stop words**
    5. Perform **stemming**
- Generated **dictionary** with **4051** unique tokens

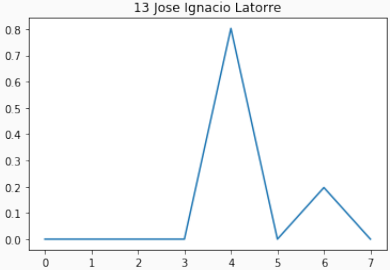
**Topic determination:**

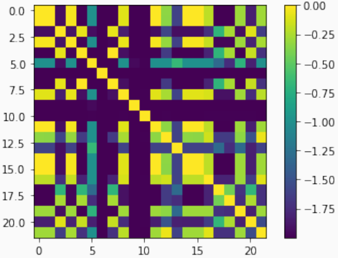
Apply Latent Dirichlet Allocation (LDA) to the bag of words:

topic 2: u'0.020*"higg" + 0.016*"boson" + 0.012*"qcd"'...
topic 4: u'0.016*"theori" + 0.012*"lattic" + 0.012*"quantum"...
topic 6: u'0.018*"pdf" + 0.014*"parton" + 0.012*"determin"...
Found a total of 8 topics!

## http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

- For each participant compute the **probability** for each of the 8 topics, e.g. JI is ∼80% topic 4 ⇒ quantum, theori, ...

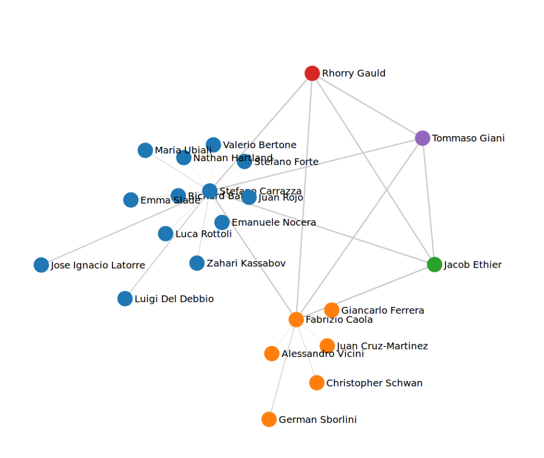


- Repeat this procedure for all participants and compute the euclidean distance between participants.

# http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

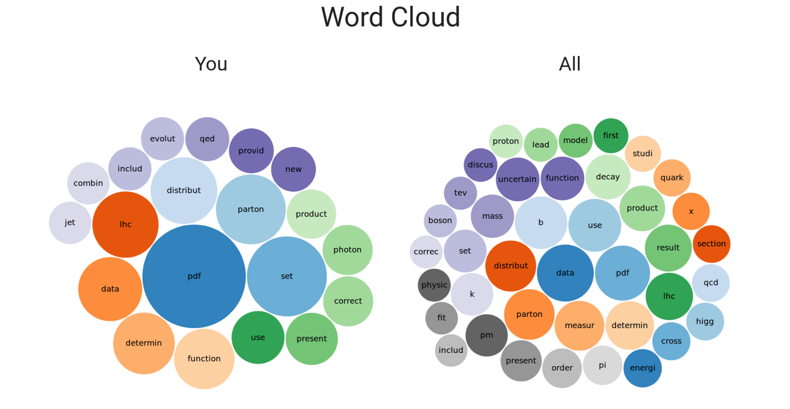
- Cluster with **affinity propagation algorithm**:




- Quite good without human intervention!

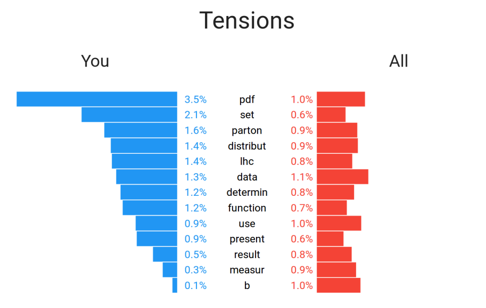
# http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

We can complete the topic analysis with standard semantic metrics:



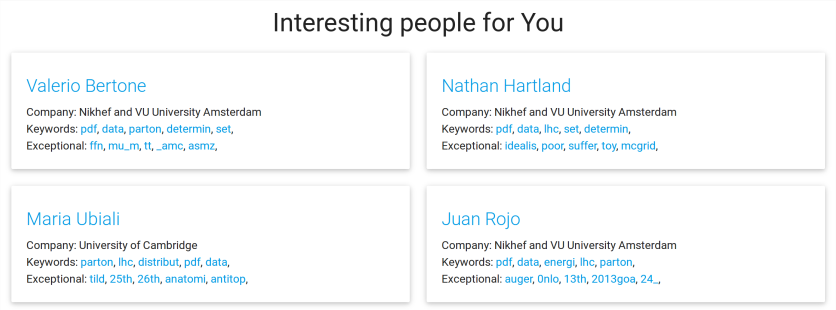
# http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

We can complete the topic analysis with standard semantic metrics:




# http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

We can propose matches based on affinity:

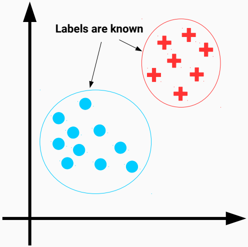


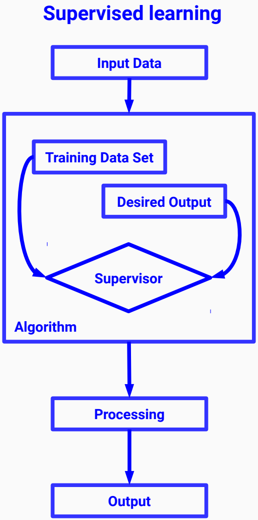
## http://apfel.mi.infn.it:7000, usr=nnpdf, psw=2018

We can even create a “micro google” for the event:

## Machine learning algorithms:

- Supervised learning:
  regression, classification, ...

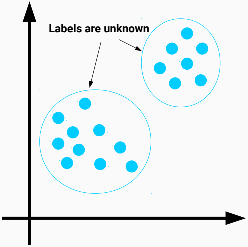


# Machine learning algorithms

**Machine learning algorithms:**

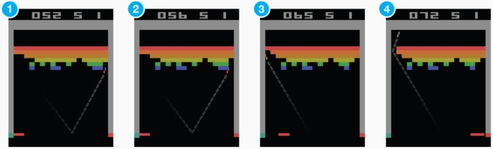
- Supervised learning:
  regression, classification, ...
- Unsupervised learning:
  clustering, dim-reduction, ...




**Unsupervised learning**

**Machine learning algorithms:**

- Supervised learning:
  regression, classification, ...
- Unsupervised learning:
  clustering, dim-reduction, ...
- Reinforcement learning:
  real-time decisions, ...

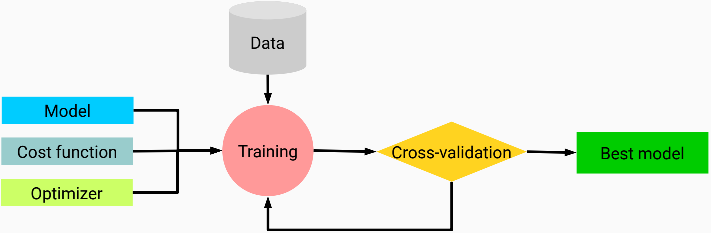



# Machine learning algorithms


More than 60 algorithms.

# Workflow in machine learning

The operative workflow in ML is summarized by the following steps:



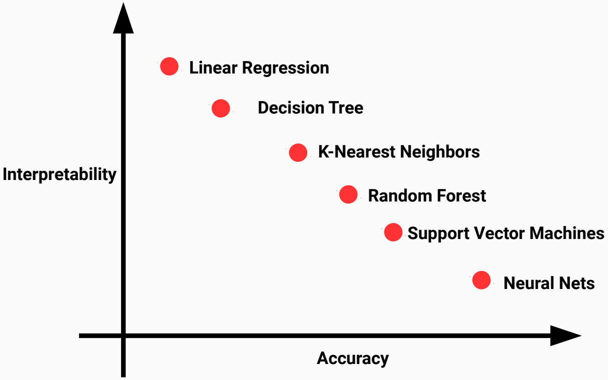

The best model is then used to:

- supervised learning: make predictions for new observed data.
- unsupervised learning: extract features from the input data.

# Model representation trade-offs

However, the selection of the appropriate model comes with trade-offs:

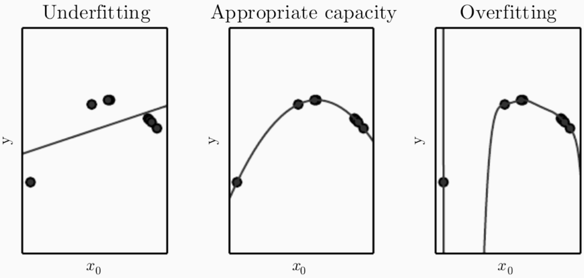
- **Prediction accuracy vs interpretability**:
  → *e.g.* linear model vs splines or neural networks.



# Model representation trade-offs

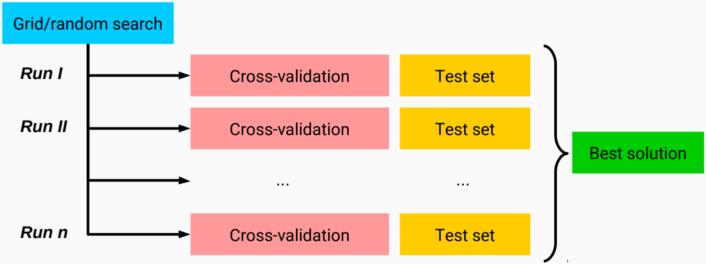
However, the selection of the appropriate model comes with trade-offs:

- **Prediction accuracy vs interpretability**:
  $\rightarrow$ *e.g.* linear model vs splines or neural networks.
- **Optimal capacity/flexibility:** number of parameters, architecture
  $\rightarrow$ deal with overfitting, and underfitting situations



# ML in practice

Perform hyperparameter tune coupled to cross-validation:




Easy parallelization at search and cross-validation stages.

# Most popular public ML frameworks

**For experimental HEP:**

- TMVA: ROOT's builtin machine learning package.

**For ML applications:**

- Keras: a Python deep learning library.
- Theano: a Python library for optimization.
- PyTorch: a DL framework for fast, flexible experimentation.
- Caffe: speed oriented deep learning framework.
- MXNet: deep learning frameowrk for neural networks.
- CNTK: Microsoft Cognitive Toolkit.
- Theta: the RTBM implementation library.

**For ML and beyond:**

- TensorFlow: libray for numerical computation with data flow graphs.
- scikit-learn: general machine learning package.

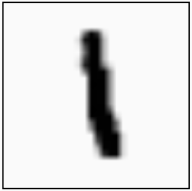
# Artificial neural networks

# Limitations of linear models

Why not linear models everywhere?

Why not linear models everywhere?

**Example:** consider 1 image from the MNIST database:



$$\simeq \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&.6&.8&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.5&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.7&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&.9&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&.3&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

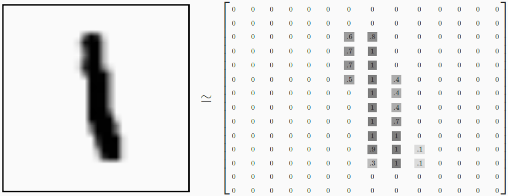
Each image has 28x28 pixels = 785 features (x3 if including RGB colors).

If consider quadratic function $\mathcal{O}(n^2)$ so linear models are impractical.

# Limitations of linear models

Why not linear models everywhere?

**Example:** consider 1 image from the MNIST database:

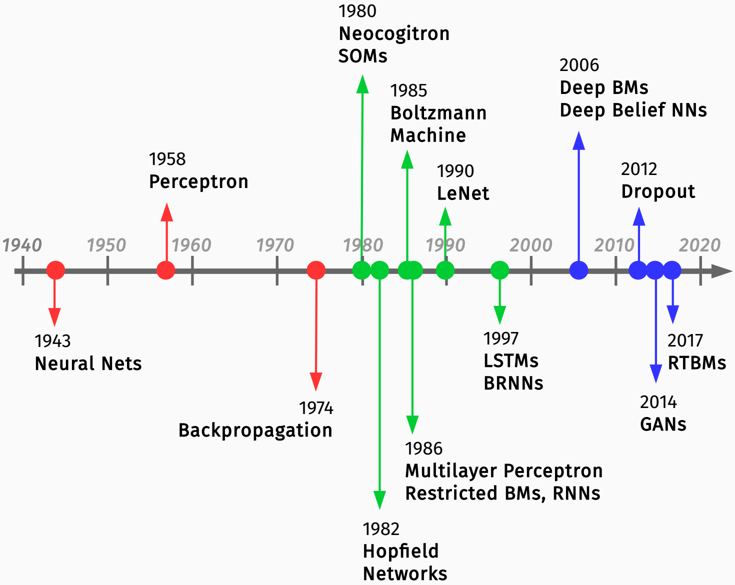


Each image has 28x28 pixels = 785 features (x3 if including RGB colors).

If consider quadratic function $\mathcal{O}(n^2)$ so linear models are impractical.

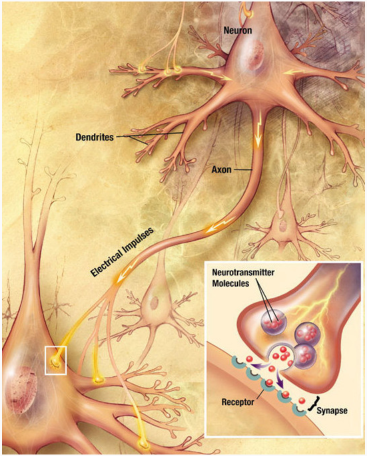
**Solution:** use non-linear models.

# Non-linear models timeline



# Neural networks

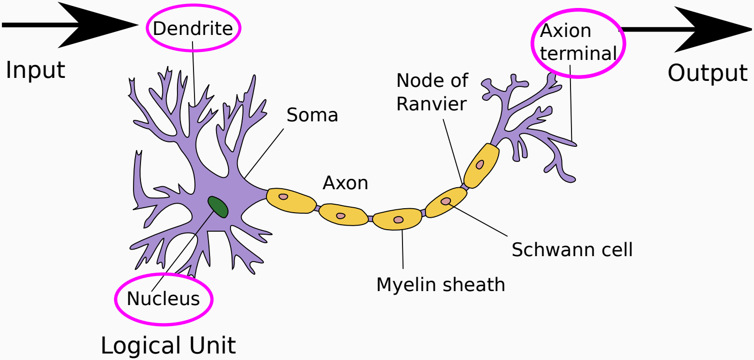
Artificial neural networks are computer systems inspired by the biological neural networks in the brain.




Currently the state-of-the-art technique for several ML applications.

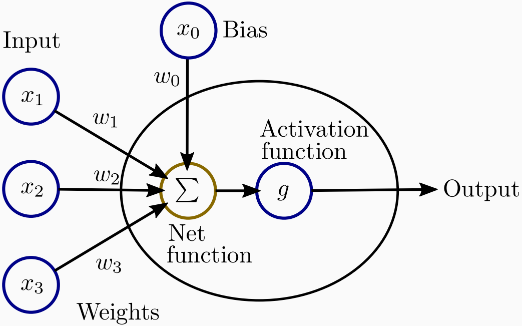
# Neuron model

We can imagine the following data communication pattern:



# Neuron model

Schematically:




where

- each **node** has an associate weights and bias $w$ and inputs $x$,
- the output is modulated by an **activation function**, $g$.

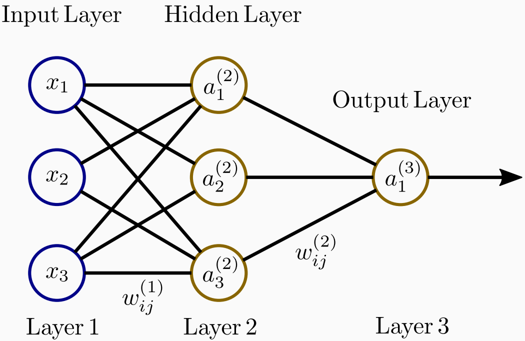
Some examples of activation functions: sigmoid, tanh, linear, ...

$$g_w(x) = \frac{1}{1+e^{-w^T x}}, \quad \tanh(w^T x), \quad x.$$

# Neural networks

In practice, we simplify the bias term with $x_0 = 1$.

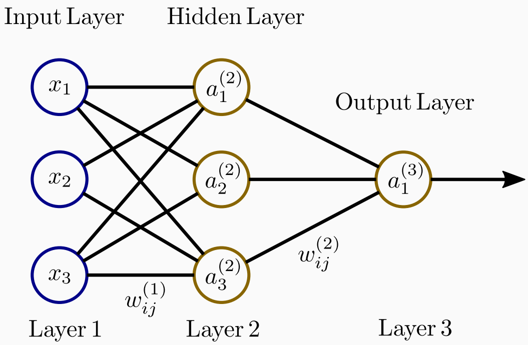
Neural network $\rightarrow$ connecting multiple units together.




where

- $a_i^{(l)}$ is the activation of unit $i$ in layer $l$,
- $w_{ij}^{(l)}$ is the weight between nodes $i, j$ from layers $l, l+1$ respectively.

# Neural networks

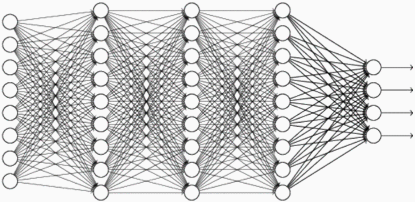



- $a_1^{(2)} = g(w_{10}^{(1)} + w_{11}^{(1)} x_1 + w_{12}^{(1)} x_2 + w_{13}^{(1)} x_3)$
- $a_2^{(2)} = g(w_{20}^{(1)} + w_{21}^{(1)} x_1 + w_{22}^{(1)} x_2 + w_{23}^{(1)} x_3)$
- $a_3^{(2)} = g(w_{30}^{(1)} + w_{31}^{(1)} x_1 + w_{32}^{(1)} x_2 + w_{33}^{(1)} x_3)$
- **Output** $\rightarrow a_1^{(3)} = g(w_{10}^{(2)} + w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)} + w_{13}^{(2)} a_3^{(2)})$

# Neural networks

Some useful names:

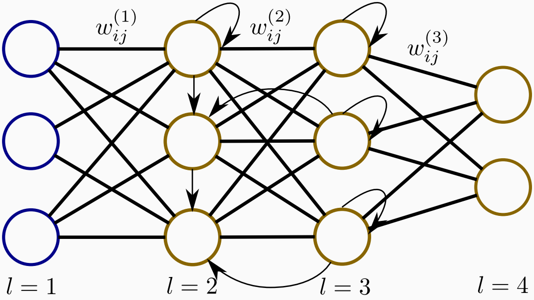
- Feedforward neural network: no cyclic connections between nodes from the same layer (previous example).
- Multilayer perceptron (MLP): is a feedforward neural network with at least 3 layers.
- Deep neural networks: term referring to neural networks with more than one hidden layer.



# Artificial neural networks architectures

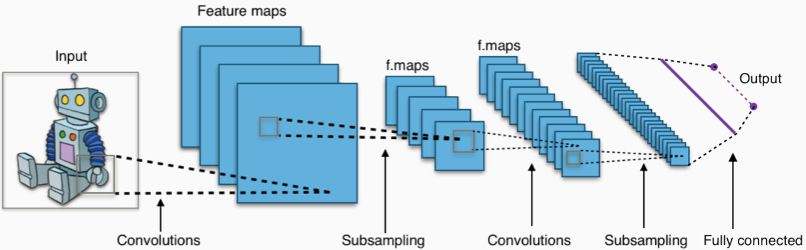
Some examples of neural network popular architectures:

- Recurrent neural networks: neural networks where connections between nodes form a directed cycle.
  - built-in internal state memory
  - built-in notion of time ordering for a time sequence



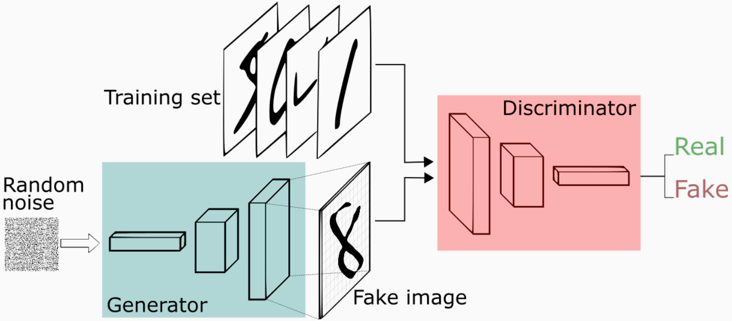
# Artificial neural networks architectures

- Convolutional neural networks: multilayer perceptron designed to require minimal preprocessing, *i.e.* space invariant architecture.
  - the hidden layers consist of convolutional layers, pooling layer, fully connected layers and normalization layers
  - great successful applications in image and video recognition.



# Artificial neural networks architectures

- Generative adversarial network: unsupervised machine learning system of two neural networks contesting with each other.
  - one network generate candidates while the other discriminates.



# Artificial neural networks architectures

Other popular examples:

- Recursive neural networks: a variation of recurrent neural network where pairs of layers or nodes are merged recursively.
  - successful applications on natural language processing.
  - some recent applications for model inference.
- Long short-term memory: another variation of recurrent neural networks composed by custom units cells:
  - LSTM cells have an input gate, an output gate and a forget gate.
  - powerful when making predictions based on time series data.
- Boltzmann Machines: is a generative stochastic recursive artificial neural network.
  - comes with energy-based model features and advantages.

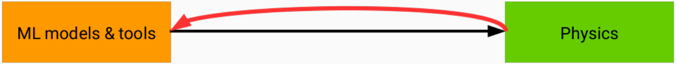
# From physics to ML

# Introduction

Lets try to build a model:

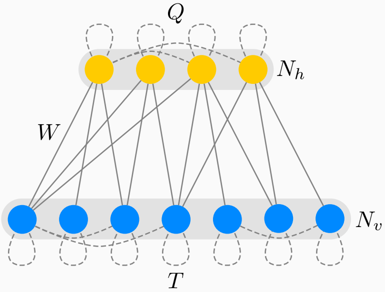
- well suited for pdf estimation and pdf sampling
- built-in pdf normalization (close form expression)
- very flexible with a small number of parameters

We decided to look at energy models, specifically Boltzmann Machines.



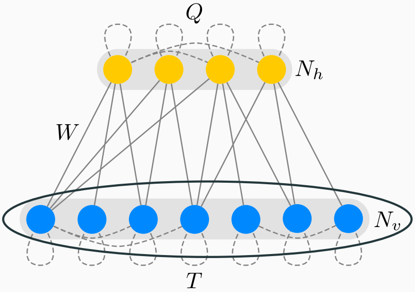
# Boltzmann machine

**Graphical representation:**



# Boltzmann machine

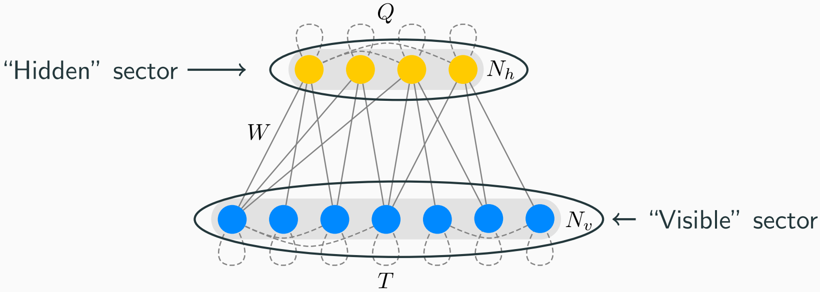
**Graphical representation:** **[Hinton, Sejnowski '86]**




← "Visible" sector

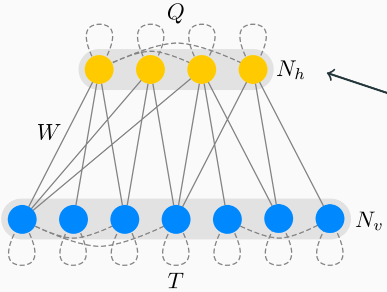
# Boltzmann machine

**Graphical representation:** **[Hinton, Sejnowski '86]**



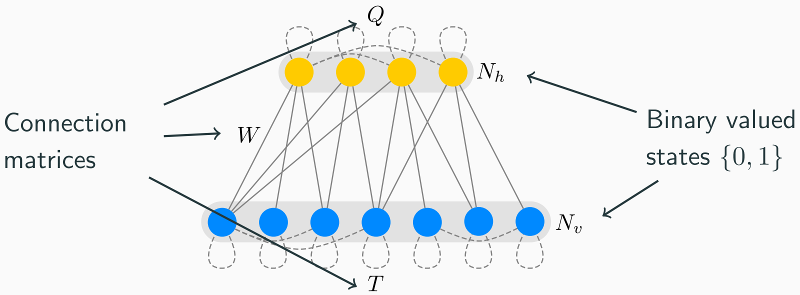
# Boltzmann machine

**Graphical representation:** **[Hinton, Sejnowski '86]**



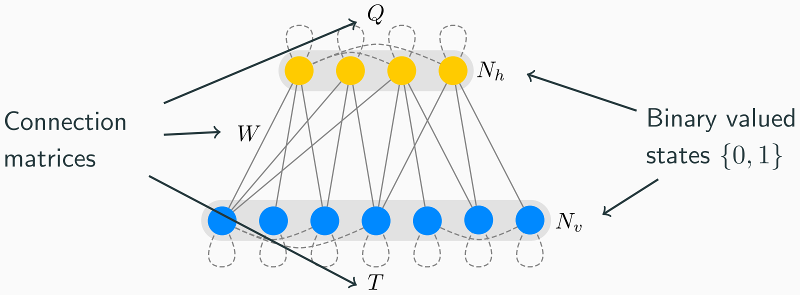

Binary valued states $\{0, 1\}$

**Graphical representation:** **[Hinton, Sejnowski '86]**



# Boltzmann machine

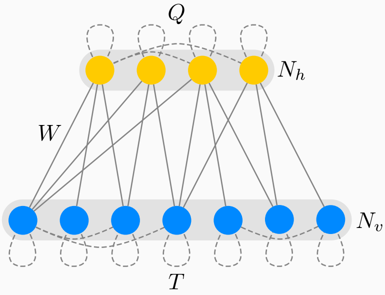
**Graphical representation:** **[Hinton, Sejnowski '86]**




- Boltzmann machine (BM): $T$ and $Q \neq 0$.
- Restricted Boltzmann machine (RBM): $T = Q = 0$.

# Boltzmann machine

**Energy based model:** **[Hinton, Sejnowski '86]**



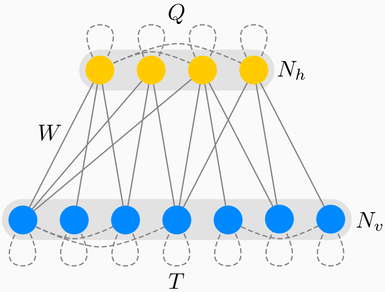

View as statistical mechanical system.

The system energy for given state vectors $(v, h)$:

$$E(v,h) = \frac{1}{2}v^tTv + \frac{1}{2}h^tQh + v^tWh + B_hh + B_vv$$

# Boltzmann machine

**Energy based model:** **[Hinton, Sejnowski '86]**




View as statistical mechanical system.

The system energy for given state vectors $(v, h)$:

$$E(v,h) = \frac{1}{2}v^t T v + \frac{1}{2}h^t Q h + v^t W h + B_h h + B_v v$$

State vectors Connection matrices Biases

# Boltzmann machine

**Energy based model:** **[Hinton, Sejnowski '86]**

Starting from the system energy for given state vectors $(v, h)$:

$$E(v,h) = \frac{1}{2}v^t T v + \frac{1}{2}h^t Q h + v^t W h + B_h h + B_v v$$

The canonical partition function is defined as:

$$Z = \sum_{h,v} e^{-E(v,h)}$$

Probability the system is in specific state given by Boltzmann distribution:

$$P(v,h) = \frac{e^{-E(v,h)}}{Z}$$

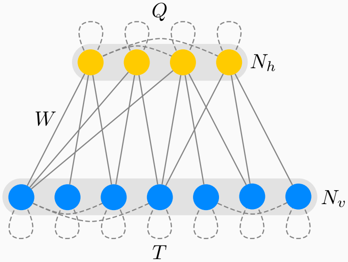
with marginalization:

$$P(v) = \frac{e^{-F(v)}}{Z}$$

Free energy

# Boltzmann machine

**Learning:** **[Hinton, Sejnowski '86]**




Theoretically, general compute medium.
Via adjusting $W, T, Q, B_h, B_v$ able to learn the underlying probability distribution of a given dataset.

**However: practically not feasible**

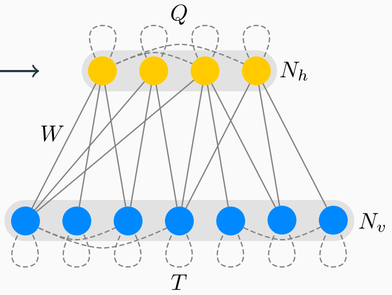
For applications only RBMs have been considered.

# Riemann-Theta Boltzmann machine

**How to change the status quo?** **[Krefl, S.C., Haghighat, Kahlen '17]**

Keep the inner sector couplings non-trivial, but the machine solvable?

Continuous valued $\in \mathbb{R}$ $\longrightarrow$




$\longleftarrow$ Continuous valued $\in \mathbb{R}$

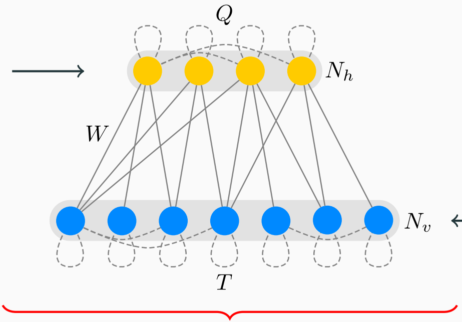
$P(v) \equiv$ multi-variate gaussian (*too trivial*)

# Riemann-Theta Boltzmann machine

**How to change the status quo?** **[Krefl, S.C., Haghighat, Kahlen '17]**

Keep the inner sector couplings non-trivial, but the machine solvable?

"Quantized" $\in \mathbb{Z}$




Continuous valued $\in \mathbb{R}$

**Something interesting happens**

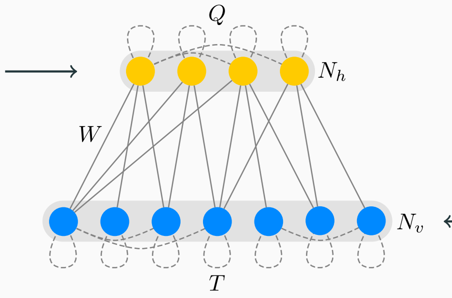
Under mild constraints on connection matrices (positive definiteness,...)

# Riemann-Theta Boltzmann machine

**How to change the status quo?** **[Krefl, S.C., Haghighat, Kahlen '17]**

Keep the inner sector couplings non-trivial, but the machine solvable?

"Quantized" $\in \mathbb{Z}$




Continuous valued $\in \mathbb{R}$

$$P(v) \equiv \sqrt{\frac{\det T}{(2\pi)^{N_v}}} e^{-\frac{1}{2}v^t T v - B_v^t v - \frac{1}{2}B_v^t T^{-1} B_v} \frac{\tilde{\theta}(B_h^t + v^t W | Q)}{\tilde{\theta}(B_h^t - B_v^t T^{-1} W | Q - W^t T^{-1} W)}$$

**Closed form analytic solution still available!**

# Riemann-Theta Boltzmann machine

**RTBM** **[Krefl, S.C., Haghighat, Kahlen '17]**

Novel very generic probability density:

$$P(v) \equiv \sqrt{\frac{\det T}{(2\pi)^{N_v}}} \underbrace{e^{-\frac{1}{2}v^tTv - B_v^t v - \frac{1}{2}B_v^t T^{-1} B_v}}_{\text{Damping factor}} \frac{\tilde{\theta}(B_h^t + v^t W|Q)}{\tilde{\theta}(B_h^t - B_v^t T^{-1} W|Q - W^t T^{-1} W)}$$

Riemann-Theta function

The Riemann-Theta definition:

$$\theta(z, \Omega) := \sum_{n \in \mathbb{Z}^{N_h}} e^{2\pi i \left(\frac{1}{2} n^t \Omega n + n^t z\right)}$$

**Key properties:** Periodicity, modular invariance, solution to heat equation, etc.

**Note:** Gradients can be calculated analytically as well so gradient descent can be used for optimization.

# RTBM properties

We observe that $P(v)$ stays in the same distribution under affine transformations, *i.e.* rotation and translation

$$\mathbf{w} = A\mathbf{v} + b, \quad \mathbf{w} \sim P_{A,b}(v),$$

if the linear transformation $A$ has full column rank.

$P_{A,b}(v)$ is the distribution $P(v)$ with parameters rotated as

$$T^{-1} \to AT^{-1}A^t\,, \quad B_v \to (A^+)^t B_v - Tb\,,$$
$$W \to (A^+)^t W\,, \quad B_h \to B_h - W^t b\,.$$

where $A^+$ is the left pseudo-inverse defined as

$$A^+ = (A^t A)^{-1} A^t.$$

# RTBM Applications

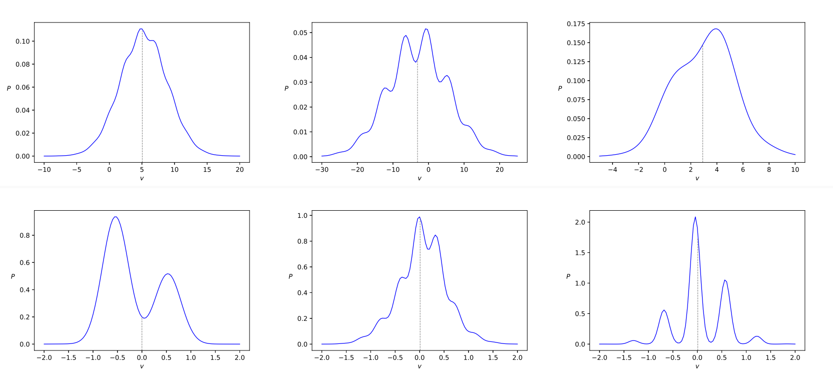
# Riemann-Theta Boltzmann machine

In the next we show examples of RTBMs for

- Probability determination
- Data classification
- Data regression
- Sampling

# Riemann-Theta Boltzmann machine

**RTBM $P(v)$ examples:** **[Krefl, S.C., Haghighat, Kahlen '17]**

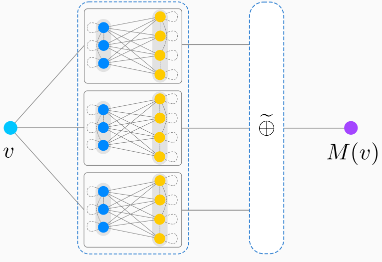



For different choices of parameters (with hidden sector in 1D or 2D).

# Riemann-Theta Boltzmann machine

**Mixture model:**

**[Krefl, S.C., Haghighat, Kahlen '17]**

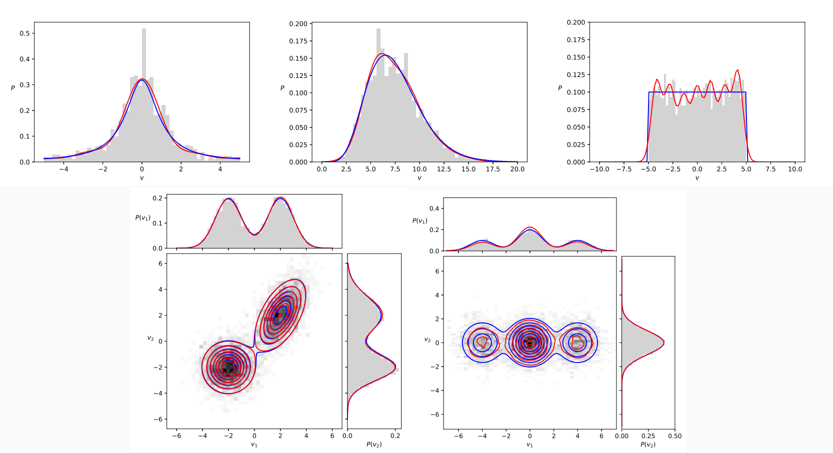



**Expectation:**

As long as the density is well enough behaved at the boundaries it can be learned by an RTBM mixture model.

# Riemann-Theta Boltzmann machine

**Examples:** [Krefl, S.C., Haghighat, Kahlen '17]




Top $N_v = 1$, $N_h = 3, 2, 3$, button $N_v = 2$, $N_h = 1\,(\text{2x RTBM}), 2$.

# Riemann-Theta Boltzmann machine

**Feature detector:**

**[Krefl, S.C., Haghighat, Kahlen '17]**

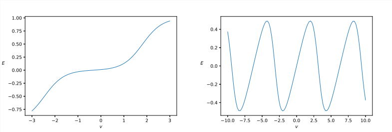
**Similar to [Krizhevsky '09]**

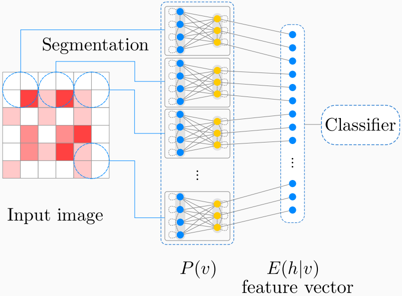
**New:**

Conditional expectations of hidden states after training

$$E(h_i|v) = -\frac{1}{2\pi i}\frac{\nabla_i \tilde{\theta}(v^t W + B_h^t|Q)}{\tilde{\theta}(v^t W + B_h^t|Q)}$$

The detector is trained in probability mode and generates a feature vector.





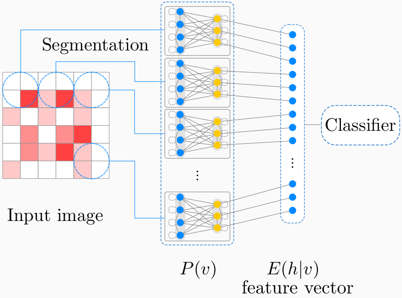
# Feature detector example - jet classification

## Jet classification:

**[Krefl, S.C., Haghighat, Kahlen '17]**

**Data from [Baldi et al. '16, 1603.09349]**

Descriminating jets from single hadronic particles and overlapping jets from pairs of collimated hadronic particles.




**Data (images of 32x32 pixels)**

- 5000 images for training
- 2500 images for testing

| Classifier | Test dataset precision |
|---|---|
| Logistic regression (LR) | 77% |
| RTBM feature detector + LR | 83% |

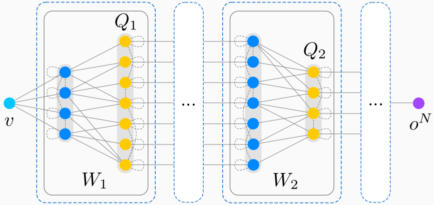
# Riemann-Theta Boltzmann machine

## Theta Neural Network:

**[Krefl, S.C., Haghighat, Kahlen '17]**

**Idea:**

Use as activation function in a standard NN. The particular form of non-linearity is learned from data.

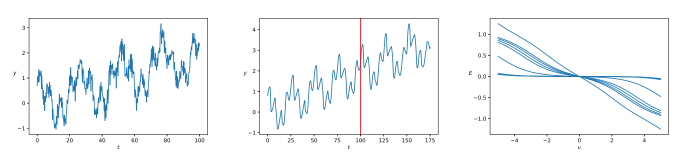



**Key point:**

smaller networks needed but Riemann-Theta evalution is expensive.

**Example (1:3-3-2:1):**

$$y(t) = 0.02t + 0.5\sin(t + 0.1) + 0.75\cos(0.25t - 0.3) + \mathcal{N}(0, 1)$$

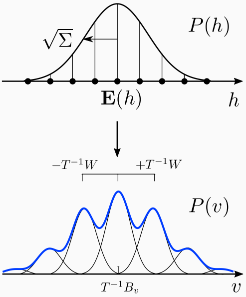


$y(t)$ TNN fit TNN activations

# RTBM sampling algorithm

The probability for the visible sector can be expressed as:

$$P(v) = \sum_{[h]} P(v|h)P(h)$$




where $P(v|h)$ is a multivariate gaussian. The $P(v)$ sampling can be performed easily by:

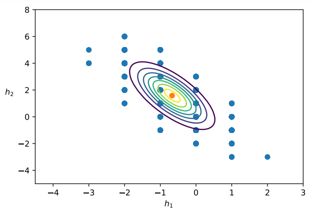
- sampling $\mathbf{h} \sim P(h)$ using the RT numerical evaluation $\theta = \theta_n + \epsilon(R)$ with ellipsoid radius $R$ so

  $$p = \frac{\epsilon(R)}{\theta_n + \epsilon(R)} \ll 1$$

  is the probability that a point is sampled outside the ellipsoid of radius $R$, while

  $$\sum_{[h](R)} P(h) = \frac{\theta_n}{\theta_n + \epsilon(R)} \approx 1$$

  *i.e.* sum over the lattice points inside the ellipsoid.

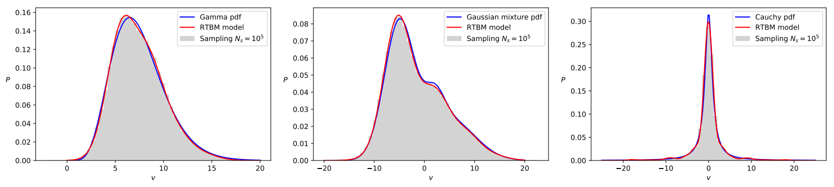


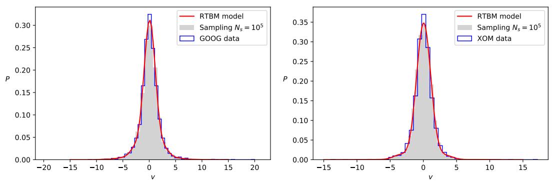

- then sampling $\mathbf{v} \sim P(v|\mathbf{h})$

# Sampling examples

## RTBM $P(v)$ sampling examples:

[S.C. and Krefl '18]




Top $N_v = 1,\ N_h = 2, 3\,(\text{2x RTBM}), 3,$ button $N_v = 1,\ N_h = 3.$

# Sampling distance estimators

| **Distribution** | $\chi^2_{\mathrm{RTBM}}/N_{\mathrm{bins}}$ | $\mathrm{MSE}^{\mathrm{sampling}}_{\mathrm{RTBM}}$ | $\mathrm{MSE}^{\mathrm{sampling}}_{\mathrm{pdf}}$ | $\mathrm{MSE}^{\mathrm{pdf}}_{\mathrm{RTBM}}$ | KS distance |
|---|---|---|---|---|---|
| Gamma | 0.02/50 | $2 \cdot 10^{-5}$ | $2.6 \cdot 10^{-5}$ | $3.4 \cdot 10^{-4}$ | 0.01 |
| Cauchy | 0.12/50 | $2.9 \cdot 10^{-4}$ | $3.7 \cdot 10^{-4}$ | $1.5 \cdot 10^{-3}$ | 0.02 |
| Gaussian mixture | 0.01/50 | $6.7 \cdot 10^{-6}$ | $1.4 \cdot 10^{-5}$ | $9.3 \cdot 10^{-5}$ | 0.01 |
| GOOG | 0.10/50 | $2.7 \cdot 10^{-4}$ | $9.5 \cdot 10^{-3}$ | $2.5 \cdot 10^{-4}$ | 0.02 |
| XOM | 0.09/50 | $2.6 \cdot 10^{-4}$ | $6.7 \cdot 10^{-3}$ | $3.7 \cdot 10^{-4}$ | 0.02 |

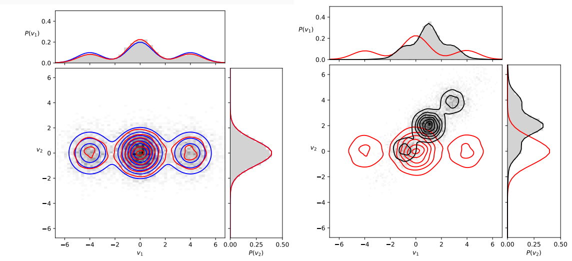
TABLE I: Distance estimators for the sampling examples in figures 3 and 4. Exact definitions for all distance estimators are given in section VII. The mean squared error (MSE) is taken between the sampling, the RTBM model and the underlying distribution (pdf). The Kolmogorov-Smirnov (KS) distance is shown in the last column of the table. For GOOG and XOM the empirical distribution is employed as underlying pdf.

| **Distribution** | Mean | 2nd moment | 3th moment | 4th moment |
|---|---|---|---|---|
| Gamma | 7.43 (7.43) [7.49] | 6.91 (6.89) [7.41] | 10.03 (10.03) [13.79] | 154 (153.23) [195.8] |
| Cauchy | -0.057 (-0.057) [-] | 11.64 (11.64) [-] | -4.63 (-4.97) [-] | 1749.8 (1753) [-] |
| Gaussian mixture | -1.48 (-1.48) [-1.31] | 34.45 (34.45) [34.29] | 134.35 (136.67) [131.78] | 3558.7 (3571.8) [3569.1] |
| GOOG | 0.06 (0.06) [0.08] | 3.28 (3.23) [3.58] | 1.52 (1.42) [6.04] | 117 (108) [191] |
| XOM | 0.02 (0.02) [0.03] | 2.13 (2.15) [2.36] | -0.42 (-0.18) [1.44] | 38.3 (40.2) [97.1] |

TABLE II: Mean and central moments for the sampling data, the RTBM model (round brackets) and the underlying true distribution (square brackets). Note that the moments of the Cauchy distribution are either undefined or infinite. The given values correspond to the RTBM model approximation and its sampling, which are defined and finite, *cf.*, (4). For the GOOG and XOM distributions the true moments (square brackets) are evaluated from the underlying empirical distribution.

# Sampling examples with affine transformation

**RTBM $P(v)$ sampling with affine transformation:** **[S.C. and Krefl '18]**




For a rotation of $\theta = \pi/4$ and scaling of 2 ($N_v = 2$, $N_h = 2$).

# Conclusion

# Summary and outlook

**In summary:**

- ML is becoming very popular and strongly used in our field.
- Results are encouraging, several application opportunities.

**For the future:**

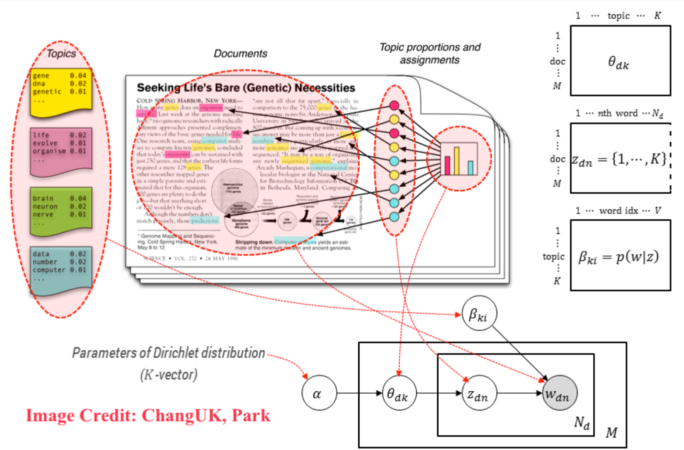
- New models based on physical systems.
- Try to extend the ML usage in physics.

# Backup

# Latent Dirichlet Allocation

In LDA each document is a considered as a mixture of various topics and that the topic distribution is assumed to have a sparse Dirichlet prior.

This means that each document cover only a small set of topics and that topics use only a small set of words frequently.



# Affinity propagation by message passing

AP is an unsupervised clustering algorithm. Let $s$ be the similarity matrix between 2 points $x_i$ and $x_j$, we define

- $R$ the responsibility matrix where $r(i,k)$ quantify how well-suited $x_k$ is to serve as the exemplar for $x_i$, relative to other candidate exemplar for $x_i$.
- $A$ the availability matrix, where $a(i,k)$ represents how appropriate would be for $x_i$ to select $x_k$ as its exemplar, taking into account other points preference for $x_k$ as an exemplar

The algorithm sets both matrix to zero and performs the following updates iteratively:

$$
\begin{aligned}
r(i,k) &\leftarrow s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\} \\
a(i,k) &\leftarrow \min\left(0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k))\right) \quad \text{for } i \neq k \\
a(k,k) &\leftarrow \sum_{i' \neq k} \max(0, r(i',k))
\end{aligned}
$$

# Affinity propagation by message passing