Statistics for HEP

Part 2: Machine Learning

Invited lectures, 12th Course on Physics of the LHC
(LIP, Lisboa, Portugal)

Dr. Pietro Vischia
pietro.vischia@cern.ch
@pietrovischia

Supported by project
RYC2021- 033305-I
funded by

If you are reading this as a web page: have fun! If you are reading this as a PDF: please visit

https://www.hep.uniovi.es/vischia/persistent/2024-03-20-22_StatisticsAtCoursePhysicsLHC_vischia_part2.html

to get the version with working animations

Lecture 2

Artificial Intelligence

Practicalities

- Significantly restructured with respect to the past years
  - Lecture 1: Probability and Statistics (minus hypothesis testing)
  - Lecture 2: Machine Learning (and hypothesis testing)
- More detailed material in my twenty-hours intensive course
  - It may be useful if you tried out the exercises, at your pace!
- Many references here and there, and in the last slide
  - Try to read some of the referenced papers!
  - Unreferenced stuff copyrighted P. Vischia for inclusion in my (finally) upcoming textbook

AI, the eternal buzzword?

- Artificial Intelligence (AI)
- Machine Learning (ML)
- Statistical Learning

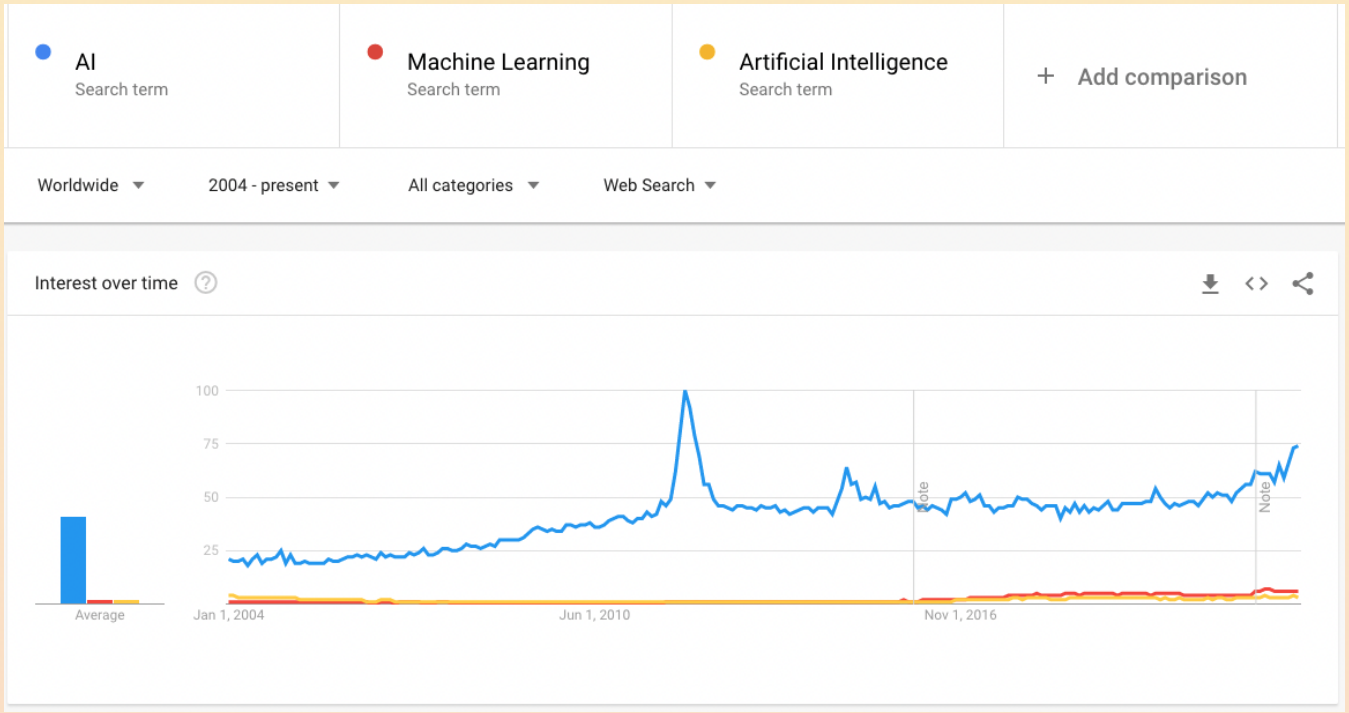
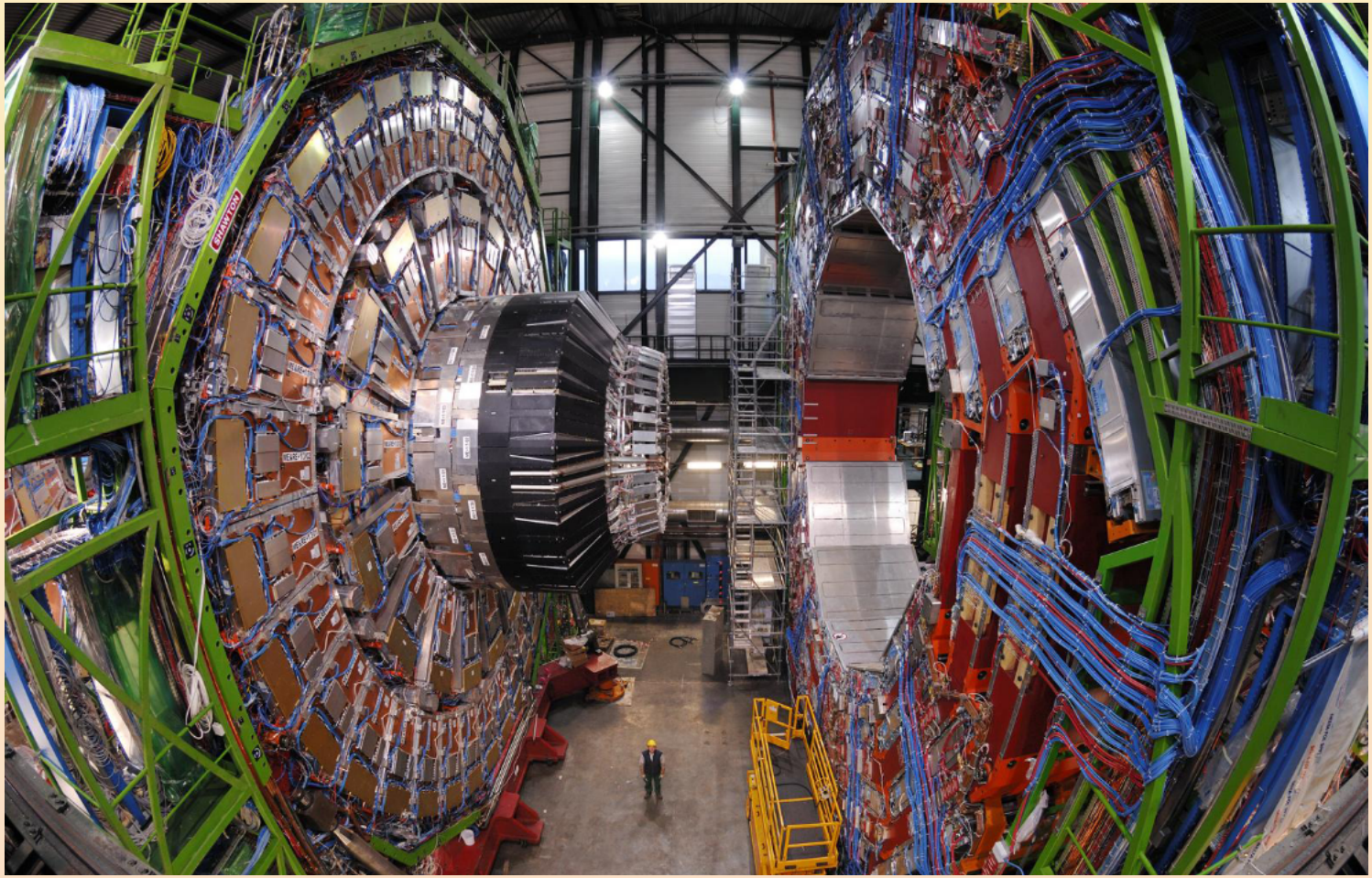
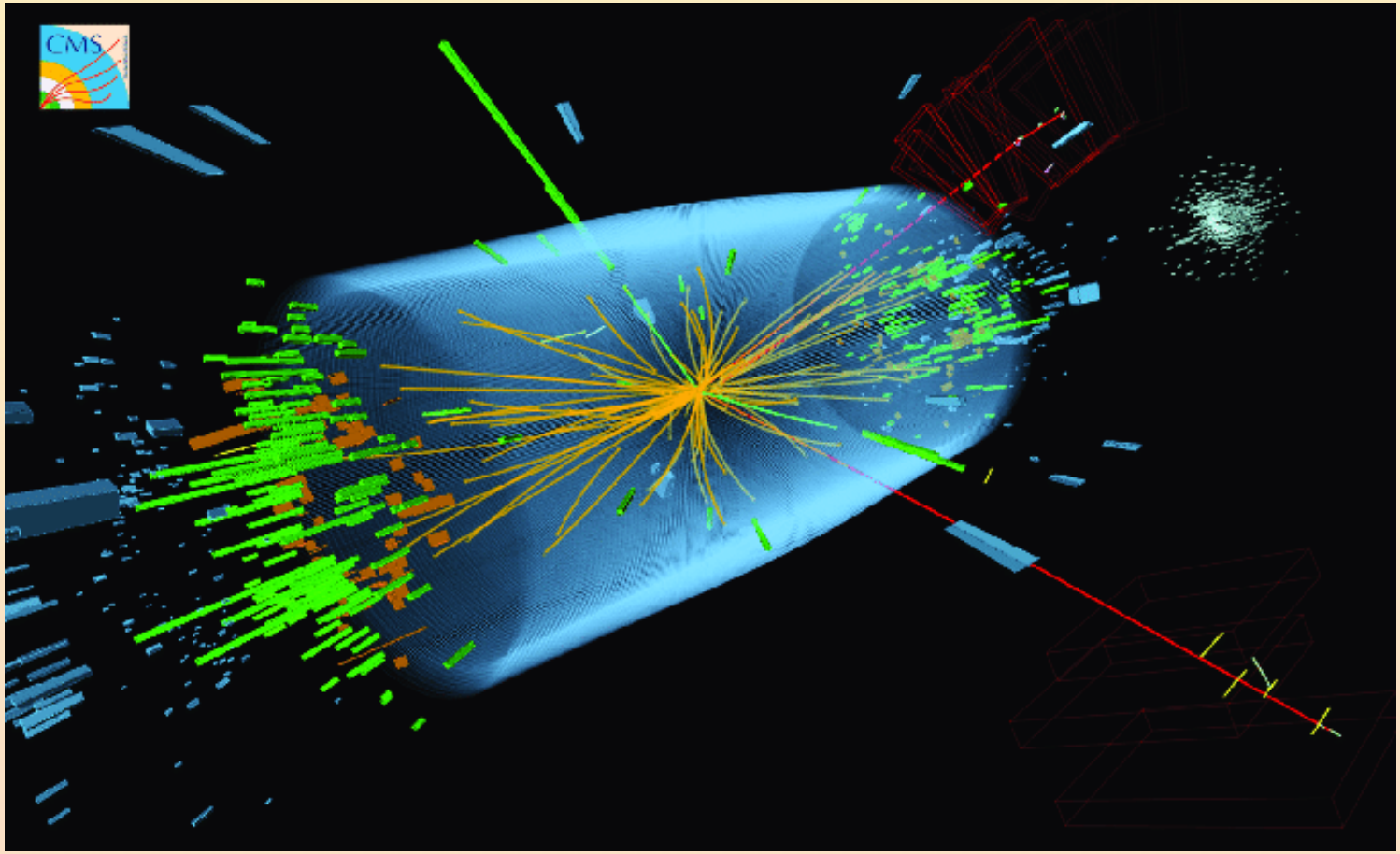


Image produced by using Google Trends

Complex Experiments



Complex Data



Likelihood and information

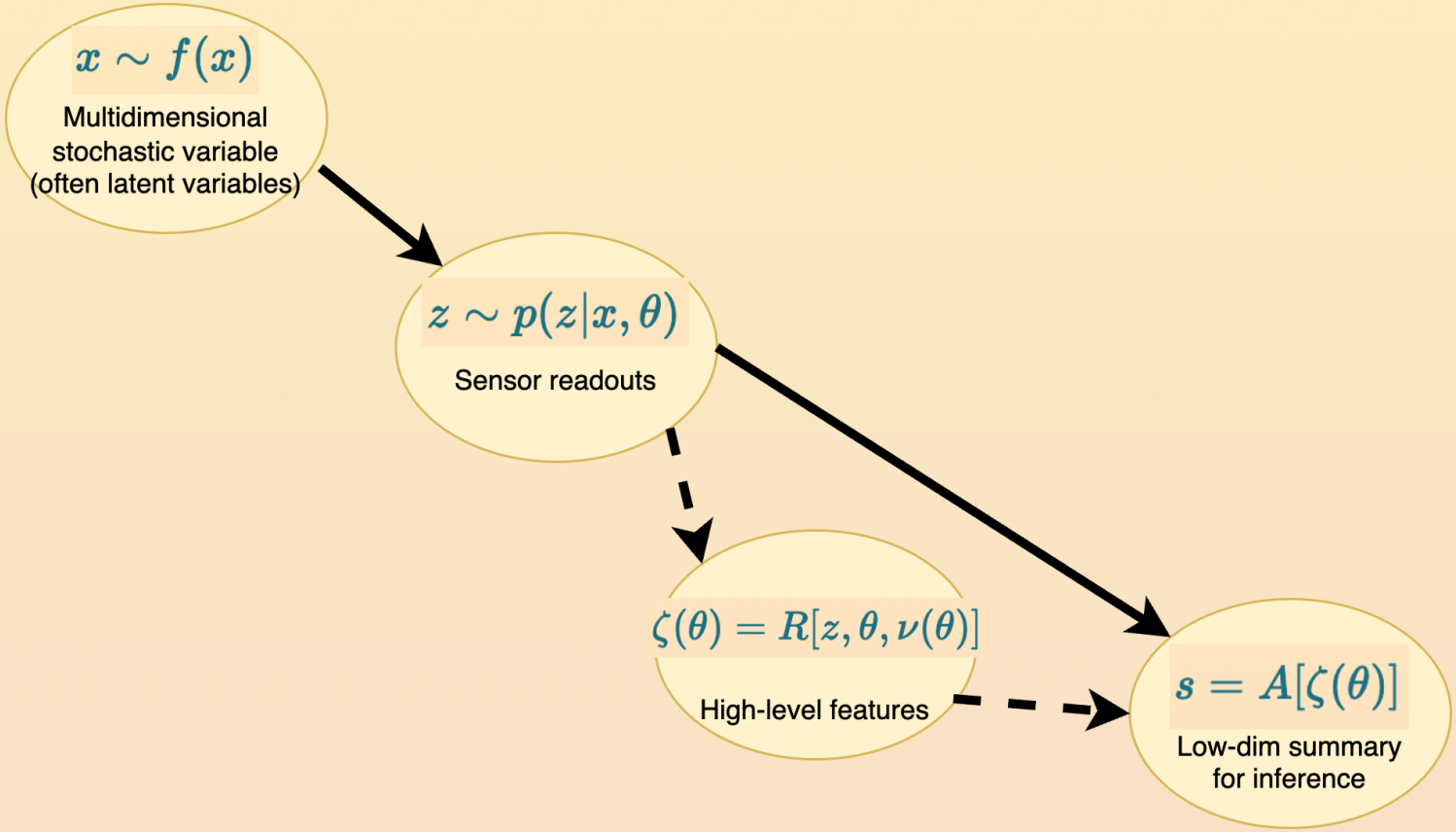
- Data sample X_{obs}

$$\mathcal{L}(X;\theta) := P(X|\theta)|_{X_{obs}}$$

- The Likelihood Principle: The likelihood function $L(\vec{x};\theta)$ contains all the information available in the data sample relevant for the estimation of θ
  - ✓ Bayesian statistics
  - ✗ Frequentist statistics

$$I(\theta) = -E\left[\left(\frac{\partial^2}{\partial\theta^2} lnL(X;\theta)\right)^2 |\theta_{true}\right]$$

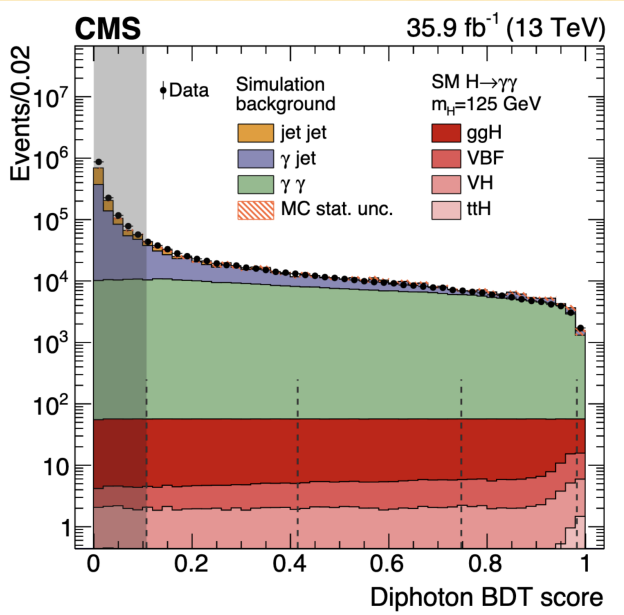
Typical analysis pipeline



Formulas from our white paper

We like low-dim summaries

- Discard uninteresting regions



- Physical observable for inference

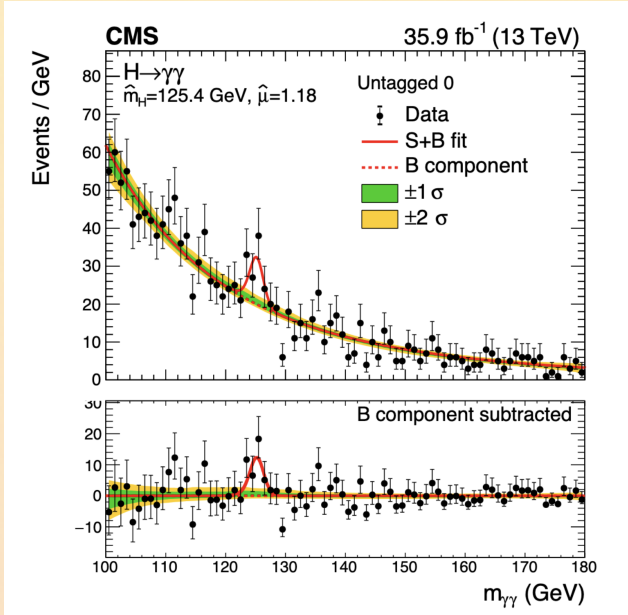


Figure from JHEP 11 (2018) 185

Brain activity...

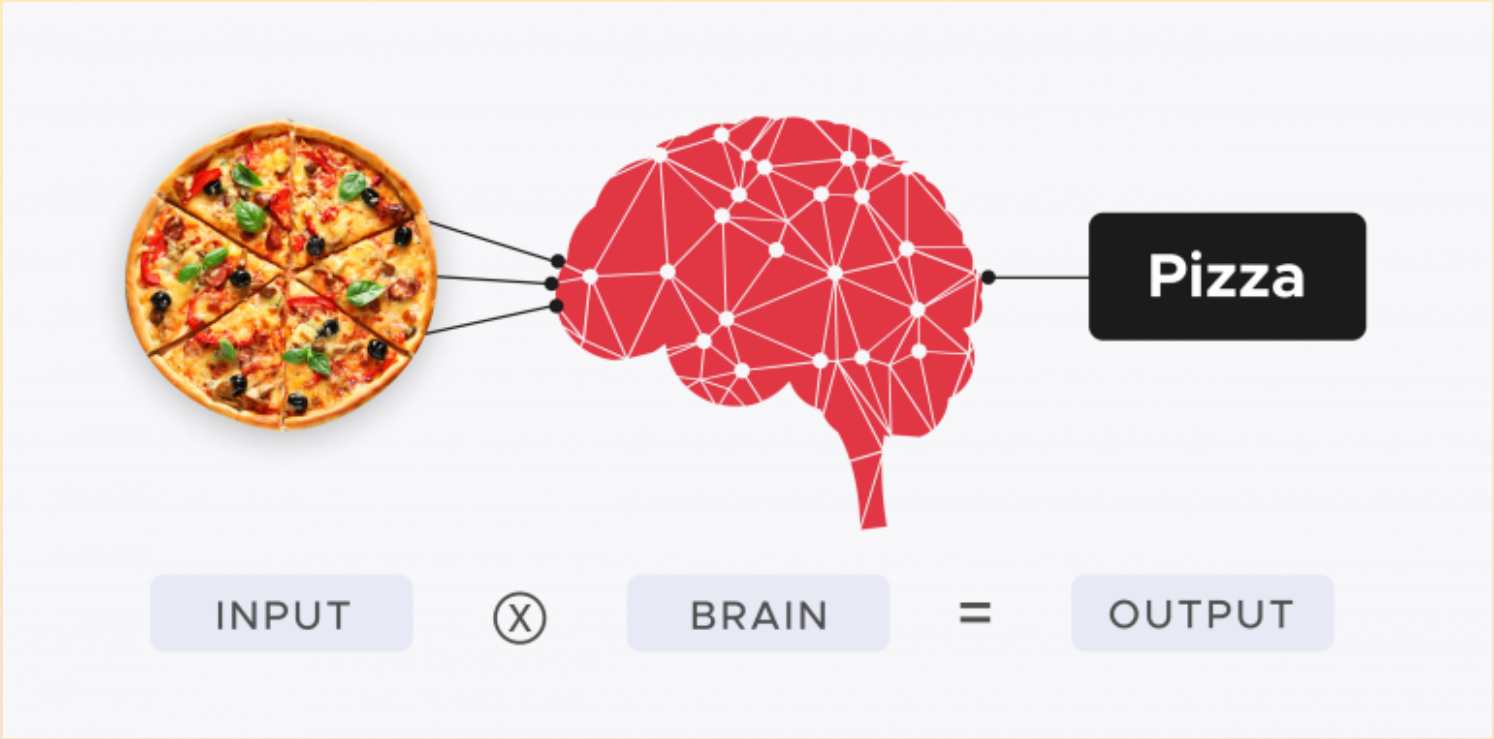


Image from zomato

...approximated...

Image from zomato, elaborated with image from pixabay

...using computers

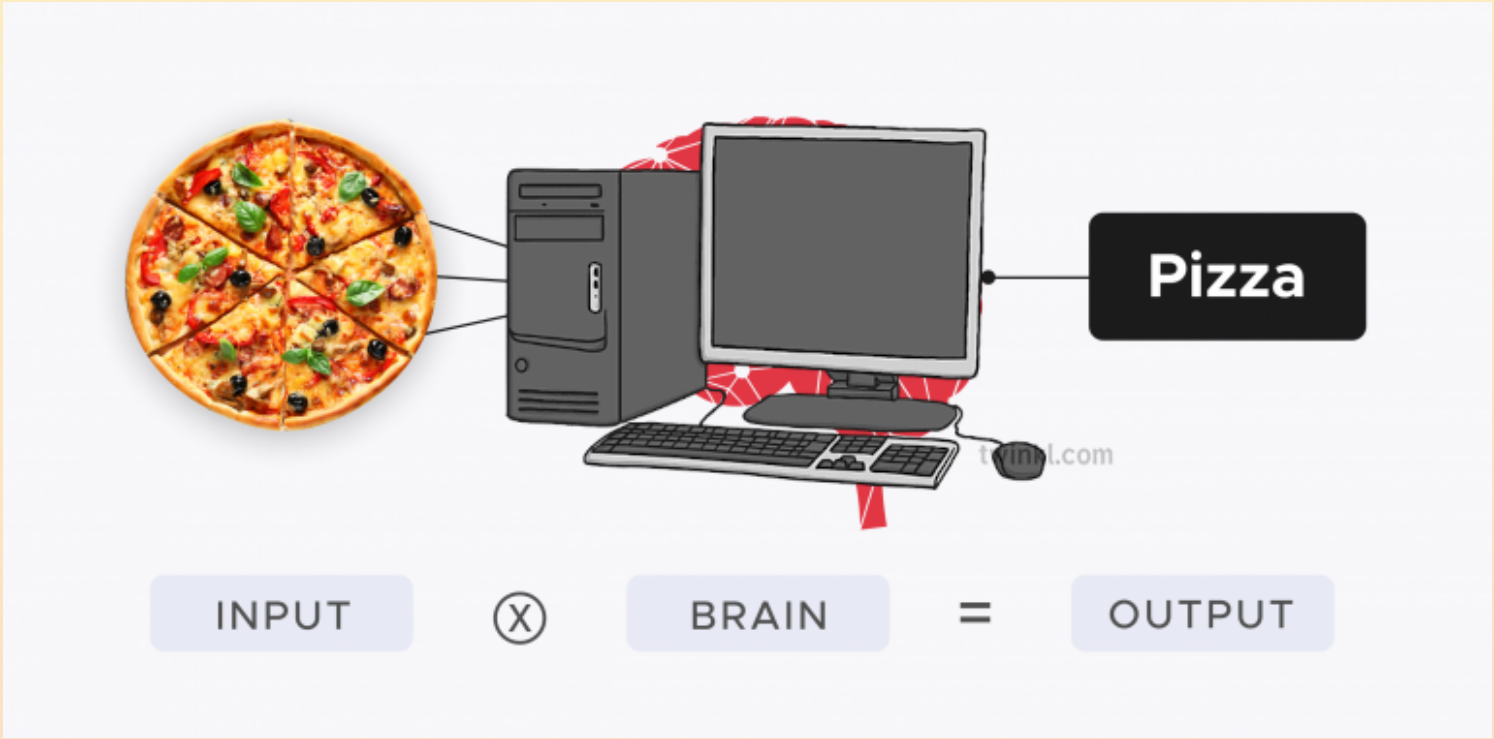
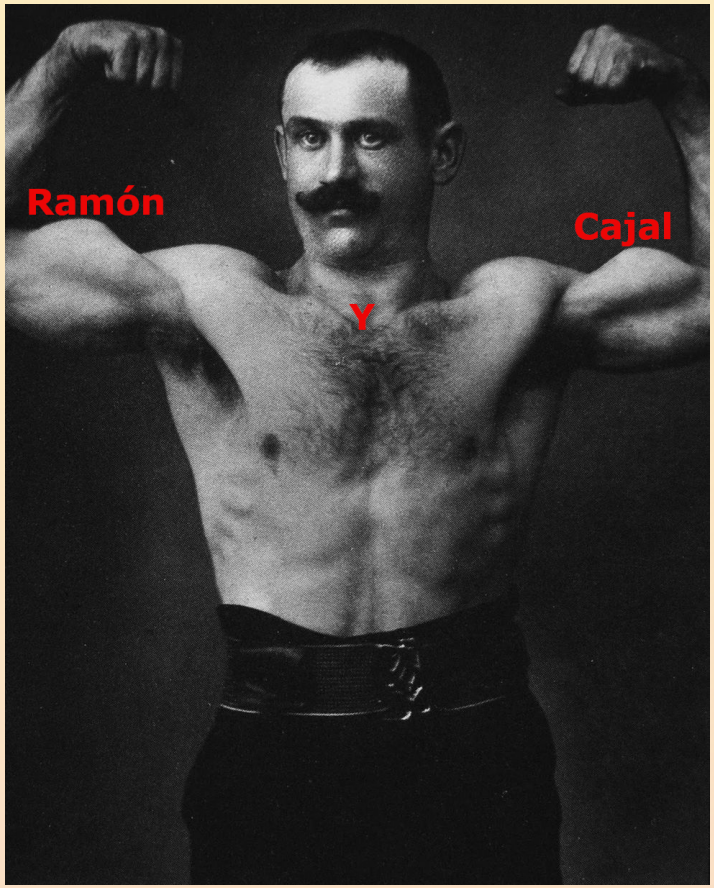


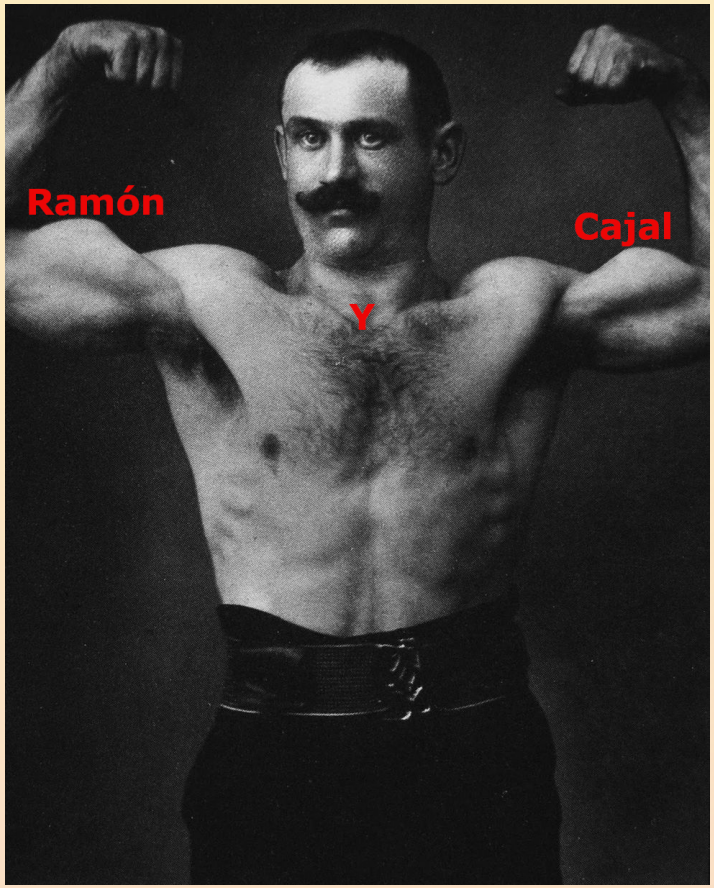
Image from zomato, elaborated with image from twinkl

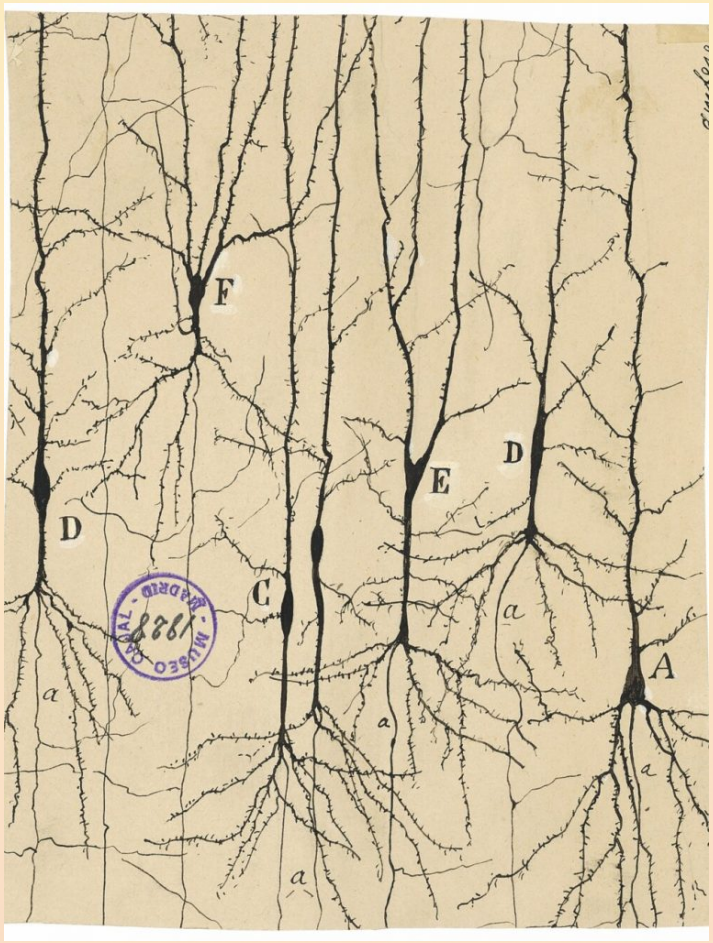
Santiago Ramón y Cajal



—
Images from menshealth and The Nobel Prize

Santiago Ramón y Cajal





Images from menshealth and The Nobel Prize

Real neurons

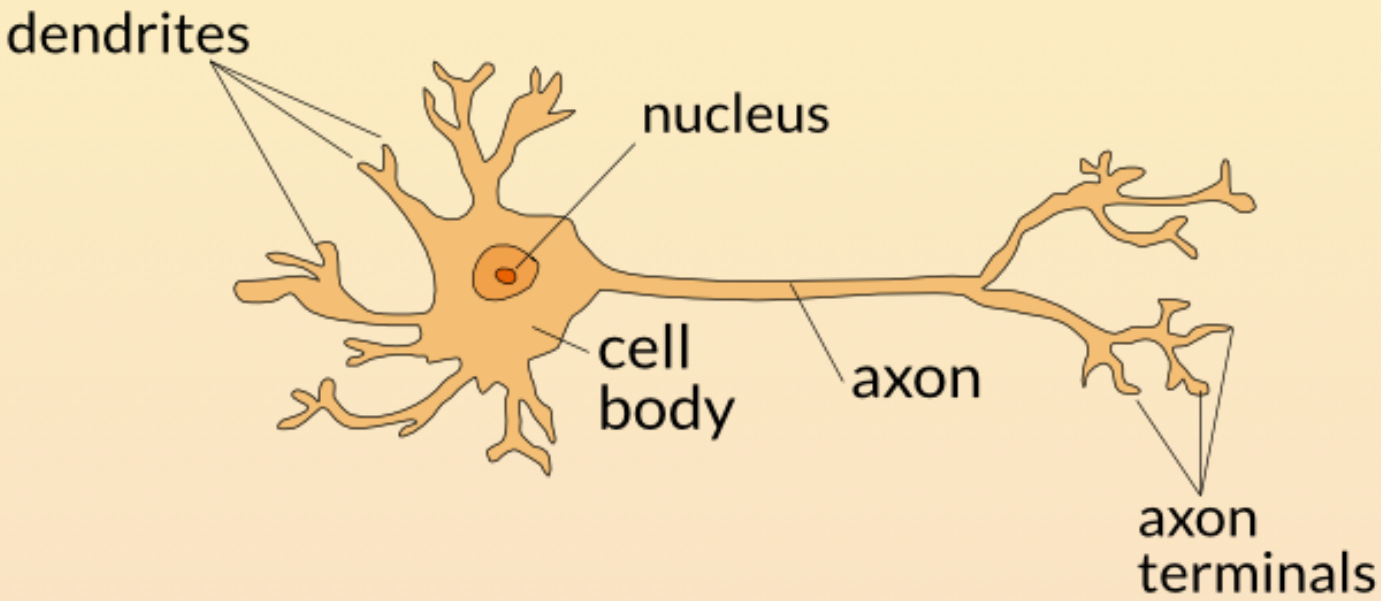
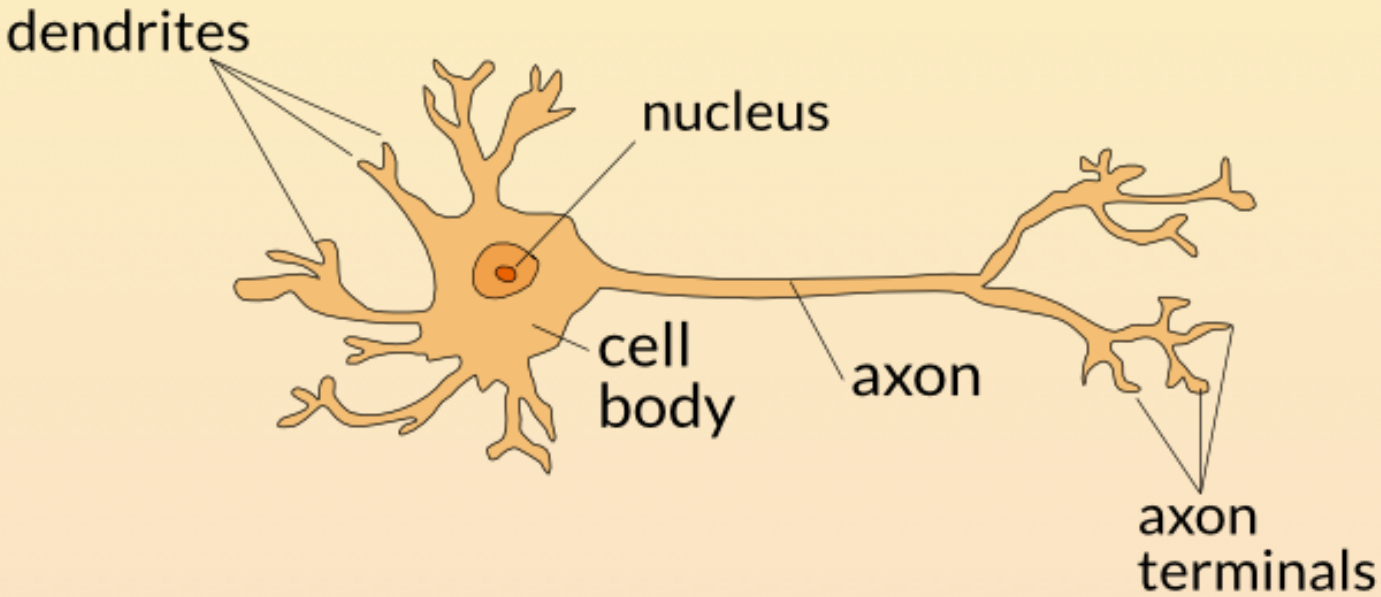


Image from https://appliedgo.net/perceptron/

Real neurons



$$I = C\frac{dV}{dt} + G_{Na}m^3h(V - V_{Na}) + G_K n^4(V - V_K) + G_L(V - V_L)$$

Image from https://appliedgo.net/perceptron/

Computationally heavy

—
Animation from giphy.com

Simplified Neurons

Bulletin of Mathematical Biology Vol. 52, No. 1/2, pp. 99–115, 1990.
Printed in Great Britain.

0092–8240/90$3.00 + 0.00
Pergamon Press plc
Society for Mathematical Biology

A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY*

■ WARREN S. MCCULLOCH AND WALTER PITTS
University of Illinois, College of Medicine,
Department of Psychiatry at the Illinois Neuropsychiatric Institute,
University of Chicago, Chicago, U.S.A.

Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of more complicated logical means for nets containing circles; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes. It is shown that many particular choices among possible neurophysiological assumptions are equivalent, in the sense that for every net behaving under one assumption, there exists another net which behaves under the other and gives the same results, although perhaps not in the same time. Various applications of the calculus are discussed.

Perceptrons

$$y = f\left(b_i + \sum w_i x_i\right)$$

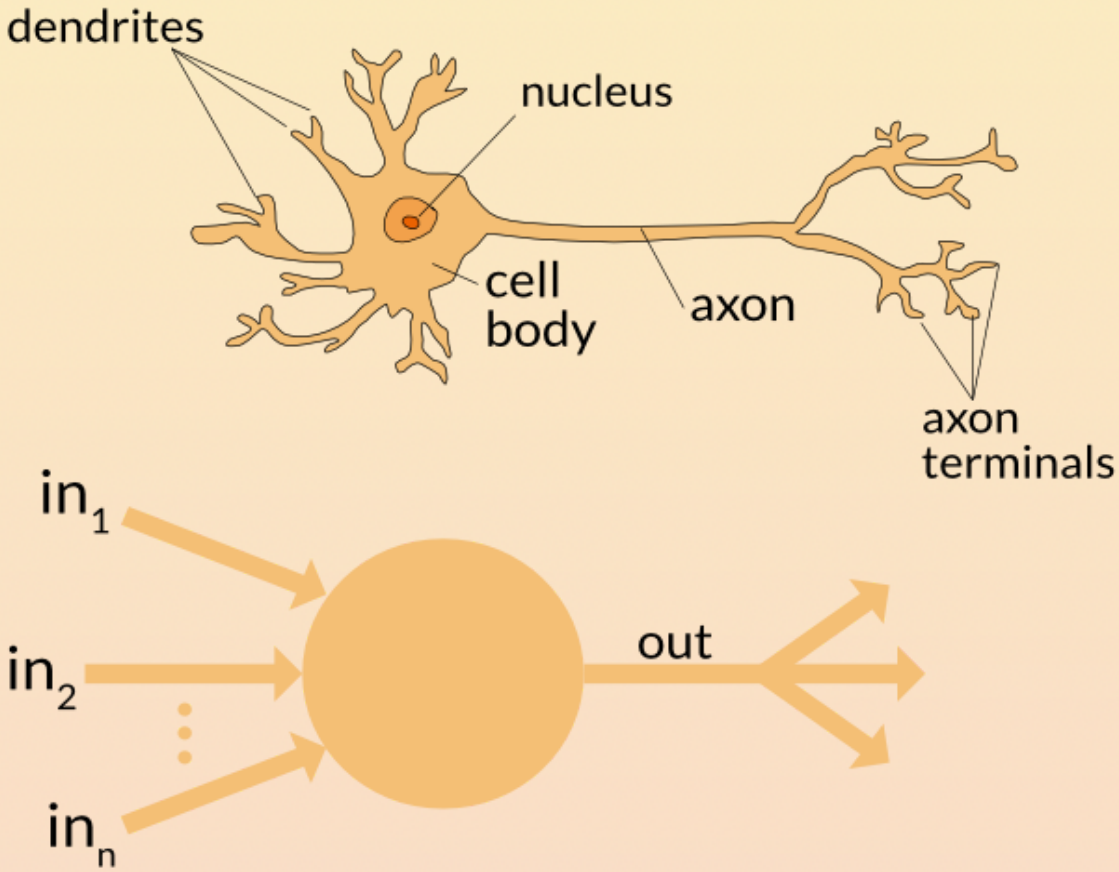


Image from https://appliedgo.net/perceptron/

The perceptron

- Linear combination of the inputs

$$\sum_j w_j x_j$$

- Activation function (imitating activation potentials in cells)

$$g\left(w_0 + \sum_j w_j x_j\right)$$

- Bias term (order zero in the inputs $\longrightarrow$ traslation)

$$\hat{y} = g\left(w_0 + \sum_j w_j x_j\right)$$

- In matrix form, it's a line-by-column product

$$\hat{y} = g\left(w_0 + \mathbf{X}^T \mathbf{W}\right)$$

Activation function

- The original activation function was linear

$$g(z) = 1 \text{ if } z = w_0 + \sum w_i x_i >= 0$$
$$g(z) = -1 \text{ if } z = w_0 + \sum w_i x_i < 0$$

- Not possible to get nonlinear separation surfaces
- Sigmoid

$g(z) = \frac{1}{1+e^{-z}}$, with derivative $g'(z) = g(z)\Big(1 - g(z)\Big)$

- Rectified Linear Unit (ReLU)

$g(z) = max(0, z)$, with derivative $g'(z) = 1$ if $z > 0, 0$ otherwise

Artificial Neural Networks

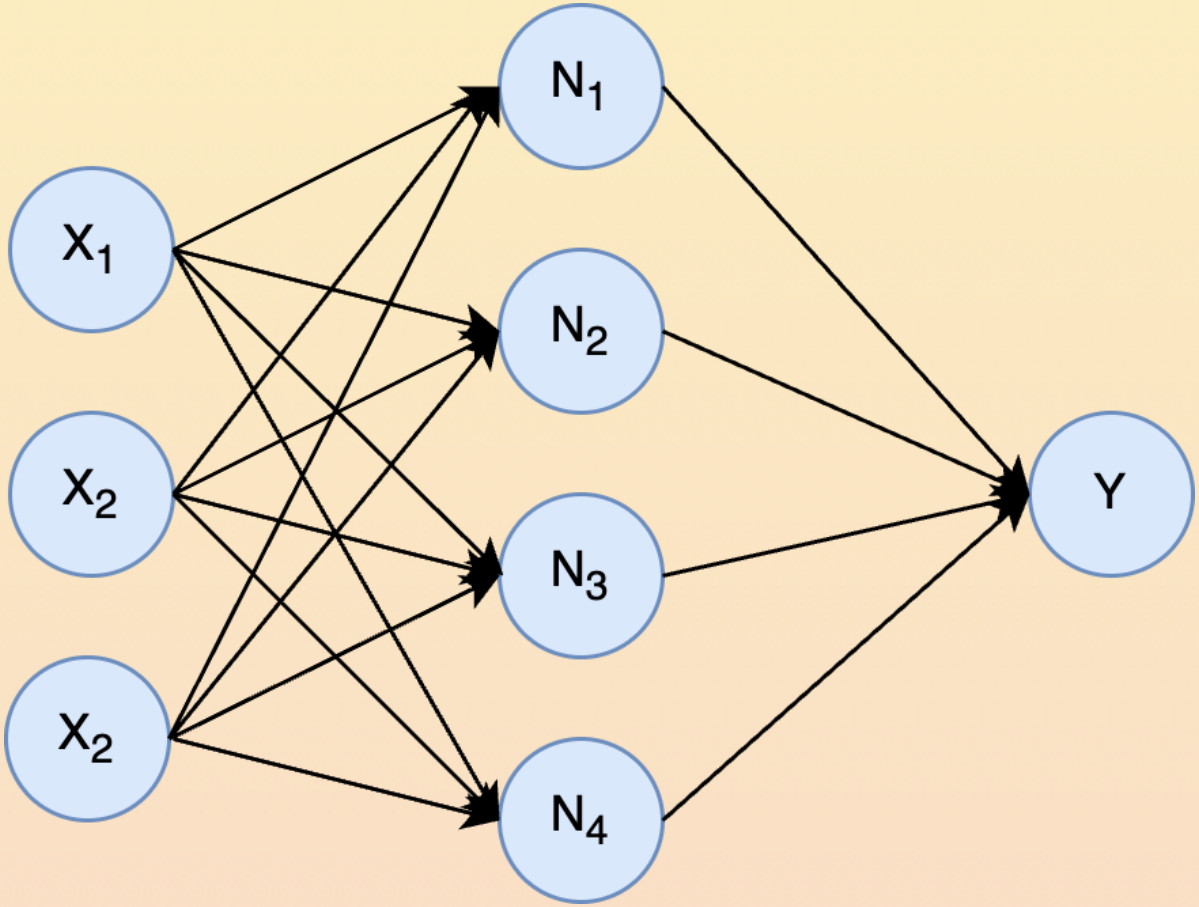


Illustration (c) P. Vischia, book in preparation

Learn in different ways

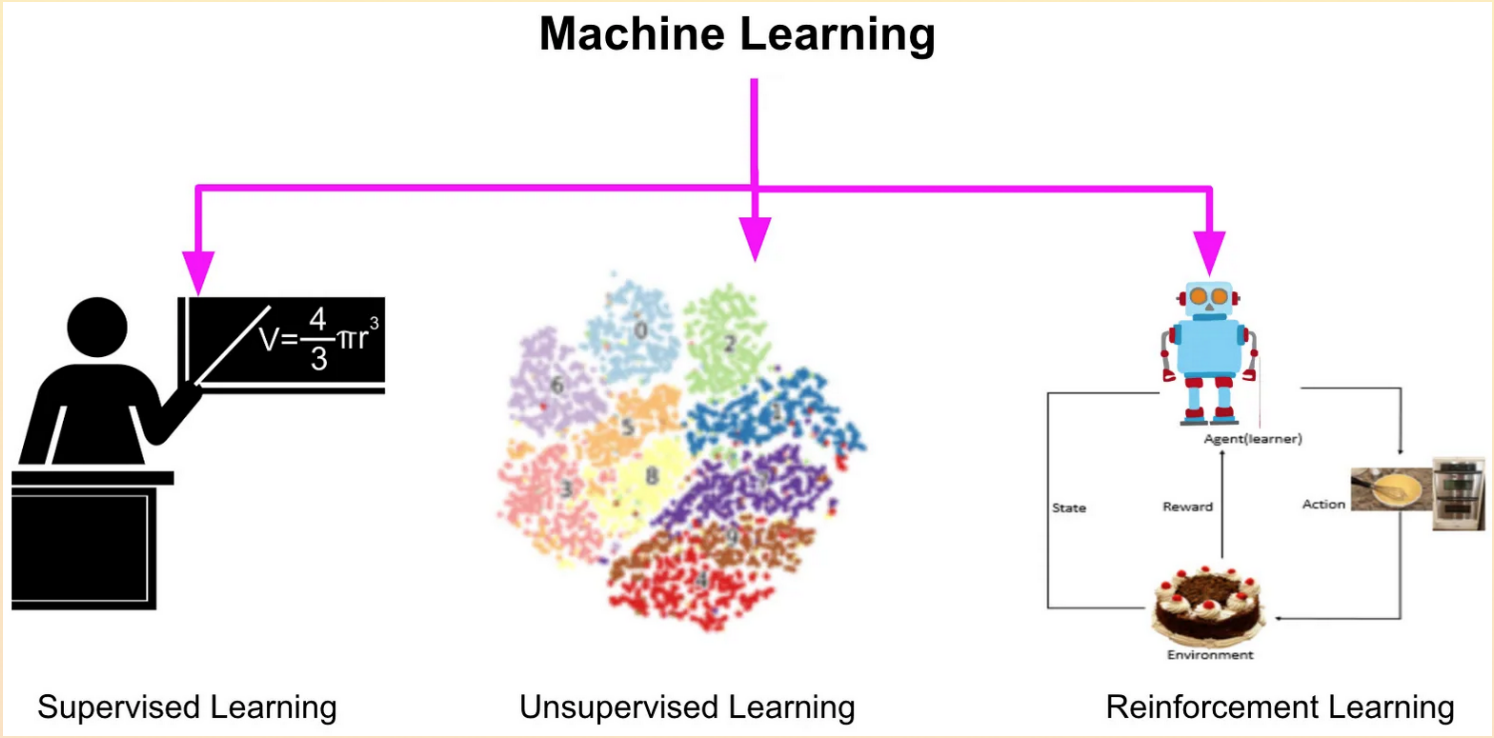


Image by Renu Khandelwal

Supervised learning

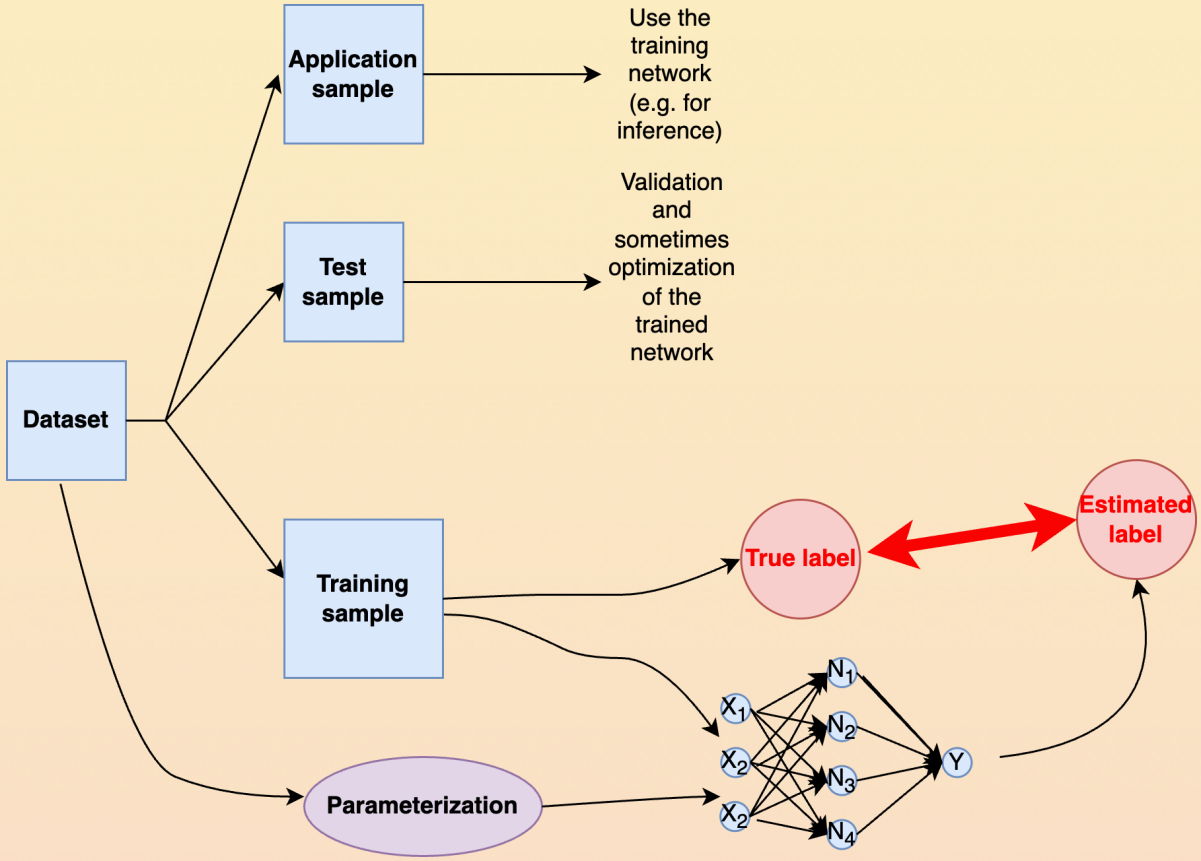
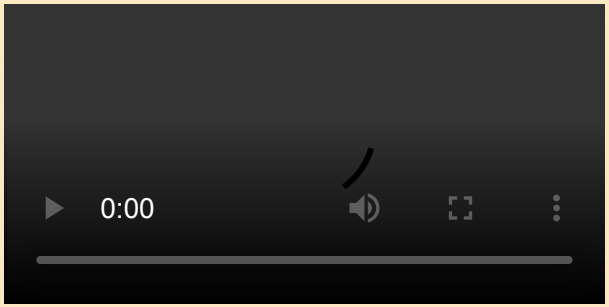


Illustration (c) P. Vischia, book in preparation

Gradient Descent

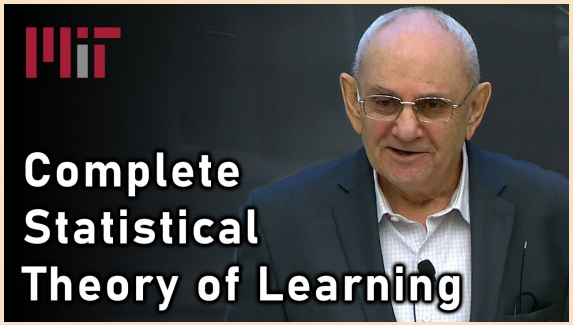
- Optimize/learn by finding the minimum of a function $\mathcal{L} : \mathbb{R}^n \to \mathbb{R}$
- Nonconvex problems: saddle points, manifolds of minima



- Empirical risk minimization

$$\hat{L}(f) = \frac{1}{n} \sum_{i=1}^{n} |f(x_i) - f^*(x_i)|^2$$

- Generalization
  (for learning problems)

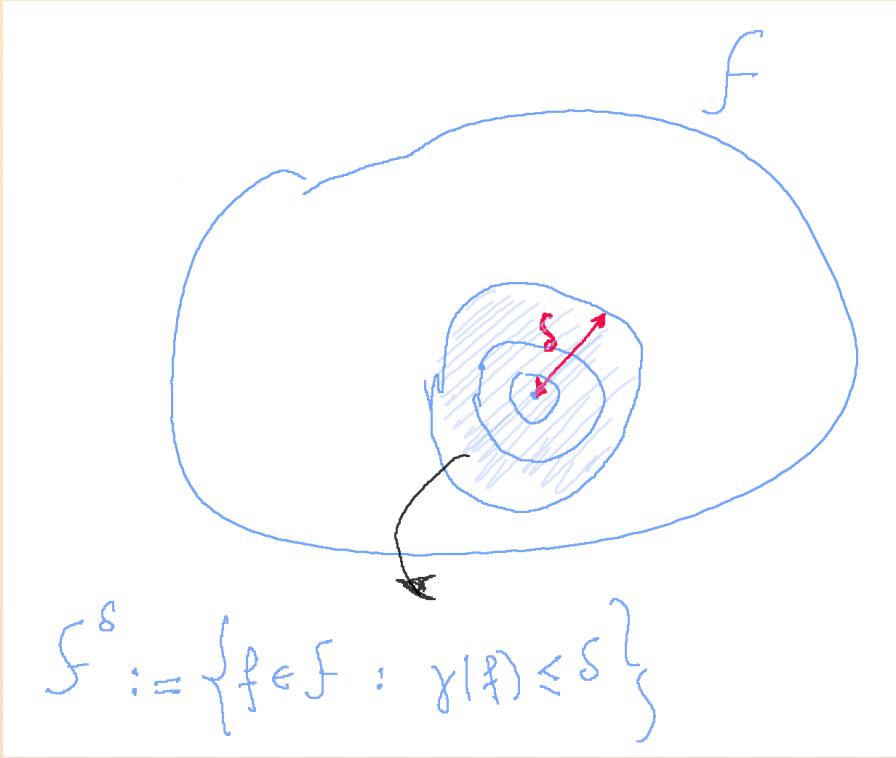


–

Vapnik's image from youtube, Gradient descent animation by Alec Radford

Organizing the space

- Formally, act on a space of function
- Need a notion of complexity.
  - Norm $\longrightarrow$ Banach space!



The Fundamental Theorem of Machine Learning

- We want to relate the result of the empirical risk minimization (ERM) with the prediction
  - Let's use the constraint form
- Let's assume we have solved the ERM at a precision ϵ (we are ϵ-away from...). We then have $\hat{f} \in \mathcal{F}^{\delta}$ such that $\hat{L}(\hat{f}) \leq \epsilon + \min_{f \in \mathcal{F}^{\delta}} \hat{L}(f)$
- How good is $\hat{f}$ at predicting f^{*}? In other words, what's the true loss?
  - Can use the triangular inequality

The Fundamental Theorem of Machine Learning

$$L(\hat{f}) - \inf_{f \in \mathcal{F}} L(f) \leq \inf_{f \in \mathcal{F}^\delta} L(f) - \inf_{f \in \mathcal{F}} L(f)$$

Approximation error

(how appropriate is my measure of complexity)

$$+2 \sup_{f \in \mathcal{F}^\delta} \left| L(f) - \hat{L}(f) \right|$$

Statistical error

(impact of having the) empirical loss instead of the true loss)

$$+\epsilon$$

Optimization error

Loss function

- Empirical Loss Function
- Cost function
- Empirical risk

$$\mathbf{J}(\mathbf{W}) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x^{(i)}; \mathbf{W}), y^{*(i)})$$

- Minimized by:

$$\mathbf{W}^0 = argmin_{\mathcal{W}} \mathbf{J}(\mathbf{W}) = argmin_{\mathcal{W}} \mathbf{J}(\mathbf{W}) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x^{(i)}; \mathbf{W}), y^{*(i)})$$

Backpropagation

$$\mathbf{J}(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(f(x^{(i)};\mathbf{W}), y^{*(i)}), \qquad \mathbf{W}^0 = argmin_{\mathcal{W}}\mathbf{J}(\mathbf{W})$$

$$\mathbf{W} \leftarrow \mathbf{W} + \eta\frac{\partial \mathbf{J}(\mathbf{W})}{\partial \mathbf{W}}$$

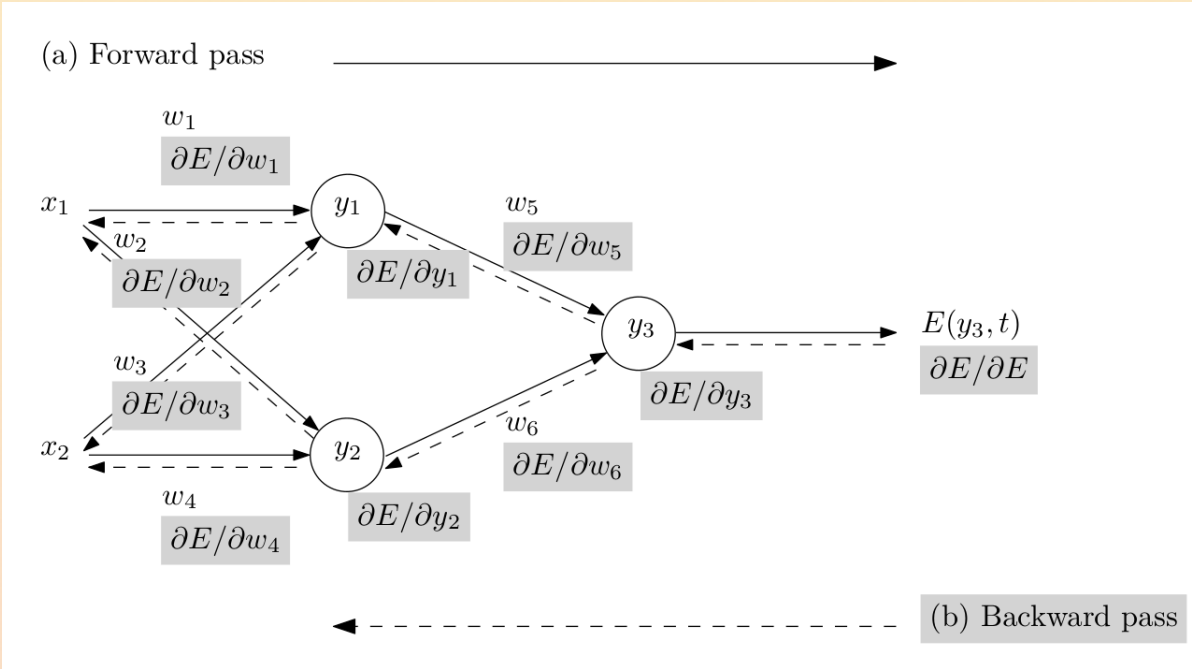


Image from Güneş Baydin et al, JMLR 18 (2018) 1--43

Derive

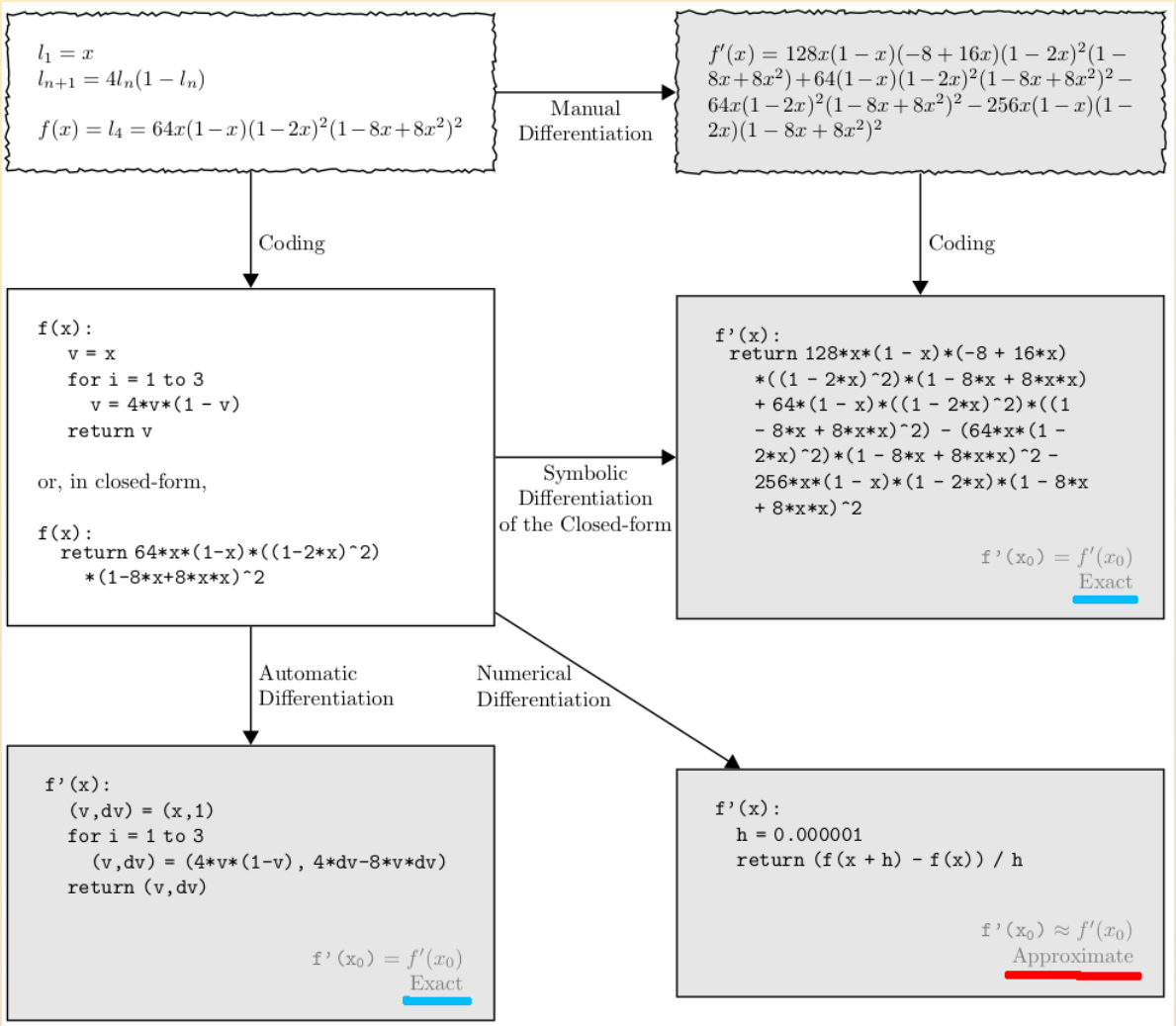


Image from Güneş Baydin et al, JMLR 18 (2018) 1--43

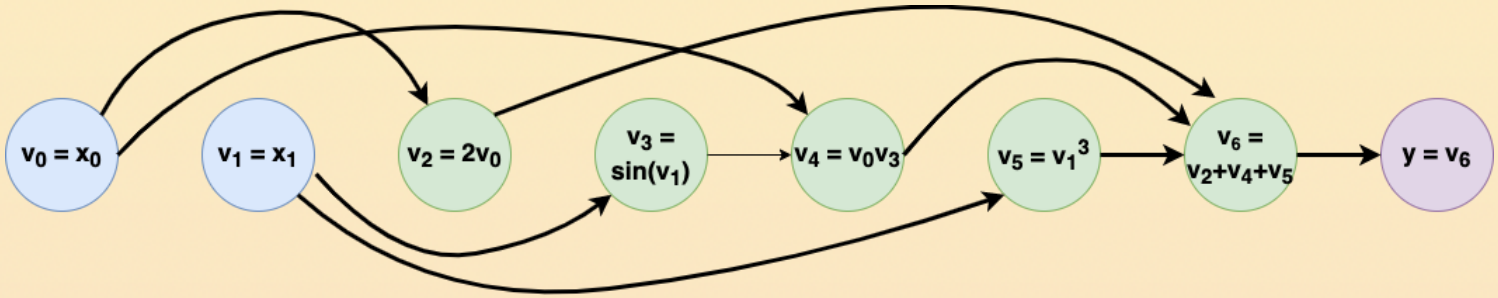
Automatic differentiation

has many names

- Automatic differentiation
- Algorithmic differentiation
- AD
- Autodiff
- Algodiff
- Autograd

Automatic differentiation

$$z(x, y) = 2x + x\,sin(y) + y^3$$



Forward mode

- To the extreme, $f : \mathbb{R} \to \mathbb{R}^m$
- Evaluates $(\frac{\partial f_1}{\partial x}, \ldots, \frac{\partial f_m}{\partial x})$

Reverse mode

- To the extreme, $f : \mathbb{R}^n \to \mathbb{R}$
- Evaluate $\nabla f(\mathbf{x})(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$

- Computational cost of calculating $\mathbf{J}_f(\mathbf{x})$ for $f : \mathbb{R}^n \to \mathbb{R}^m$ in $\mathbb{R}^n \times \mathbb{R}^m$

$\mathcal{O}(n\ \text{time}(f))$ $\mathcal{O}(m\ \text{time}(f))$

Forward and reverse (==backprop) modes

Primal: independent to dependent

Adjoint (derivatives): dependent to independent

$$y(\mathbf{x}) = 2x_0 + x_0\, sin(x_1) + x_1^3$$

Fwd Primal Trace **Atomic operation**	**Value in** $(1,2)$	***Fwd Tangent Trace (set*** $\dot{x}_0 = 1$ ***to compute*** $\frac{\partial y}{\partial x_0}$***)*** **Atomic operation**	**Value in** $(1,2)$
$v_0 = x_0$ $v_1 = x_1$	1 2	$\dot{v_0} = \dot{x_0}$ $\dot{v_1} = \dot{x_1}$	1 0
$v_2 = 2v_0$ $v_3 = sin(v_1)$ $v_4 = v_0 v_3$ $v_5 = v_1^3$ $v_6 = v_2 + v_4 + v_5$	2 0.9093 0.9093 8 10.9093	$\dot{v_2} = 2\dot{v_0}$ $\dot{v_3} = \dot{v_1} cos(v_1)$ $\dot{v_4} = \dot{v_0} v_3 + v_0 \dot{v_3}$ $\dot{v_5} = 3\dot{v_1} v_1^2$ $\dot{v_6} = \dot{v_2} + \dot{v_4} + \dot{v_5}$	2×1 0×-0.41 $1 \times 0.9093 +$ 1×0 $3 \times 0 \times 4$ $2 + 0.9093 + 0$
$y = v_6$	10.9093	$\dot{y} = \dot{v_6}$	2.9093

Fwd Primal Trace **Atomic operation**	**Value in** $(1,2)$	***Rev Adjoint Trace (set*** $\bar{y} = 1$ ***to compute*** $\frac{\partial v}{\partial y}$***)*** **Atomic operation**	**Value in** $(1,2)$
$v_0 = x_0$ $v_1 = x_1$	1 2	$\bar{x}_0 = \bar{v}_0$ $\bar{x}_1 = \bar{v}_1$	2.9093 11.5839
$v_2 = 2v_0$ $v_3 = sin(v_1)$ $v_4 = v_0 v_3$ $v_5 = v_1^3$ $v_6 = v_2 + v_4 + v_5$	2 0.9093 0.9093 8 10.9093	$\bar{v}_0 = \bar{v}_0 + \bar{v}_2 \partial v_2/\partial v_0$ $\bar{v}_0 = \bar{v}_4 \partial v_4/\partial v_0$ $\bar{v}_1 = \bar{v}_1 + \bar{v}_3 \partial v_3/\partial v_1$ $\bar{v}_1 = \bar{v}_5 \partial v_5/\partial v_1$ $\bar{v}_2 = \bar{v}_6 \partial v_6/\partial v_2$ $\bar{v}_3 = \bar{v}_4 \partial v_4/\partial v_3$ $\bar{v}_4 = \bar{v}_6 \partial v_6/\partial v_4$ $\bar{v}_5 = \bar{v}_6 \partial v_6/\partial v_5$	$\bar{v}_0 + \bar{v}_2 \times 2 = 2.9093$ $\bar{v}_4 \times v_3 = 0.9093$ $\bar{v}_1 + \bar{v}_3 \times cos(v_1) = 11.5839$ $\bar{v}_5 \times 3v_1^2 = 12$ $\bar{v}_6 \times 1 = 1$ $\bar{v}_4 \times v_0 = 1$ $\bar{v}_6 \times 1 = 1$ $\bar{v}_6 \times 1 = 1$
$y = v_6$	10.9093	$\bar{v}_6 = \bar{y}$	1

Designed to be simple in software

```
import torch, math
x0 = torch.tensor(1., requires_grad=True)
x1 = torch.tensor(2., requires_grad=True)
p = 2*x0 + x0*torch.sin(x1) + x1**3
print(p)
p.backward()
print(x0.grad, x1.grad)
```

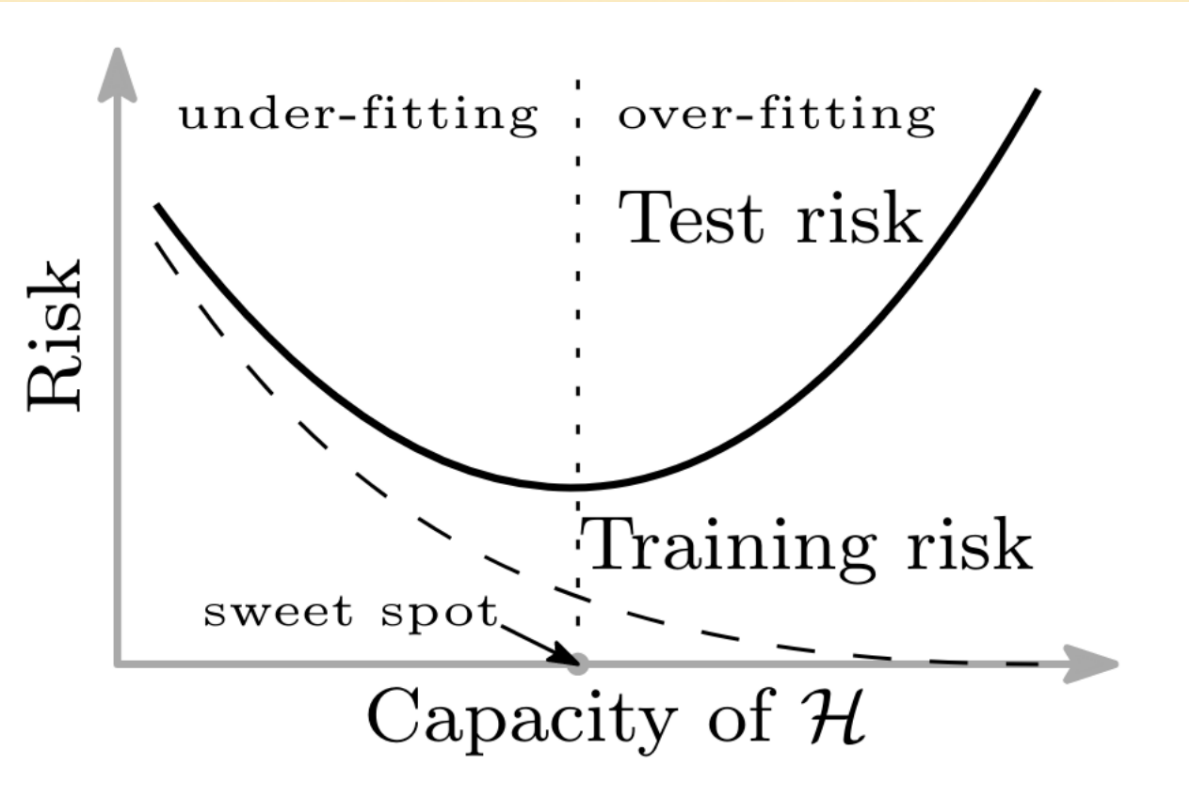
yielding

```
Primal: tensor(10.9093, grad_fn=<AddBackward0>)
Adjoint: tensor(2.9093) tensor(11.5839)
```

- `Torch` (and similar software) will correctly differentiate only when the atomic operations are supported within it
  - Common operations are overloaded (`__mul__` rewritten by `torch._mult_`)
  - Operations from libraries (`math.sin()`) must be replaced by their differentiation-aware equivalents (`torch.sin()`)

Learning as a tradeoff...

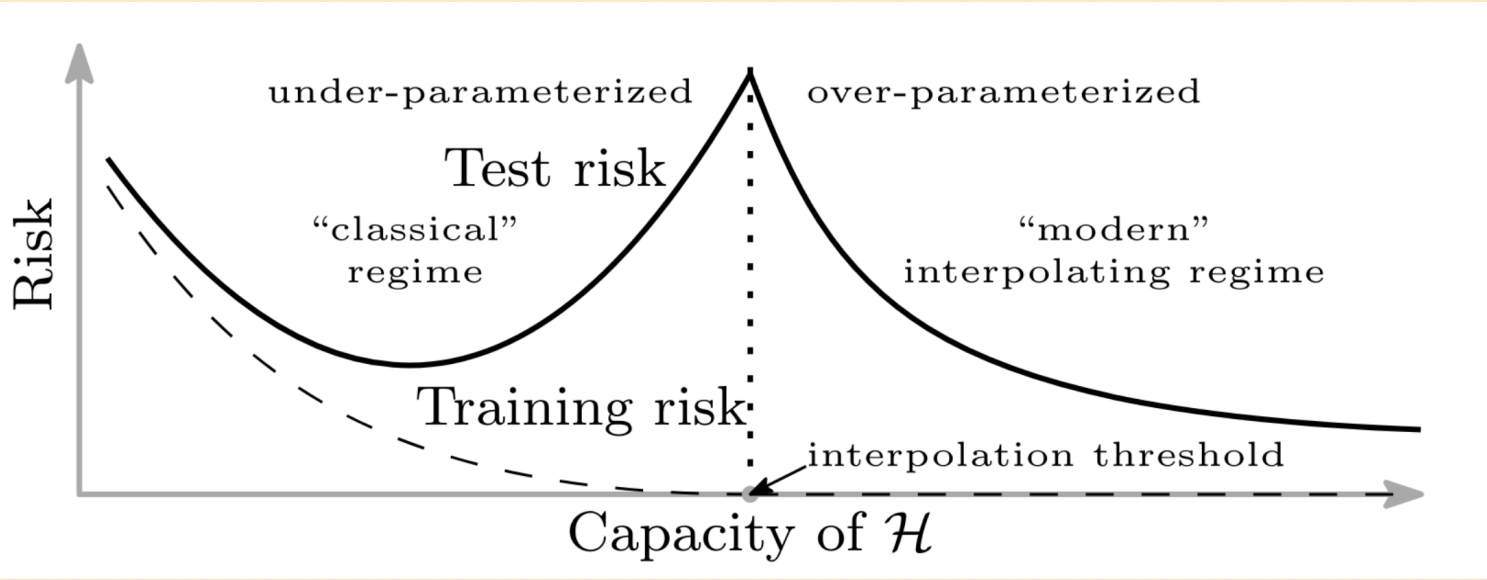
- ``a model with zero training erroris overfit to the training data and will typically generalize poorly'' (Hastie, Tibshirani, Friedman)



From Belkin et al., 1812.11118

...is still an open problem!

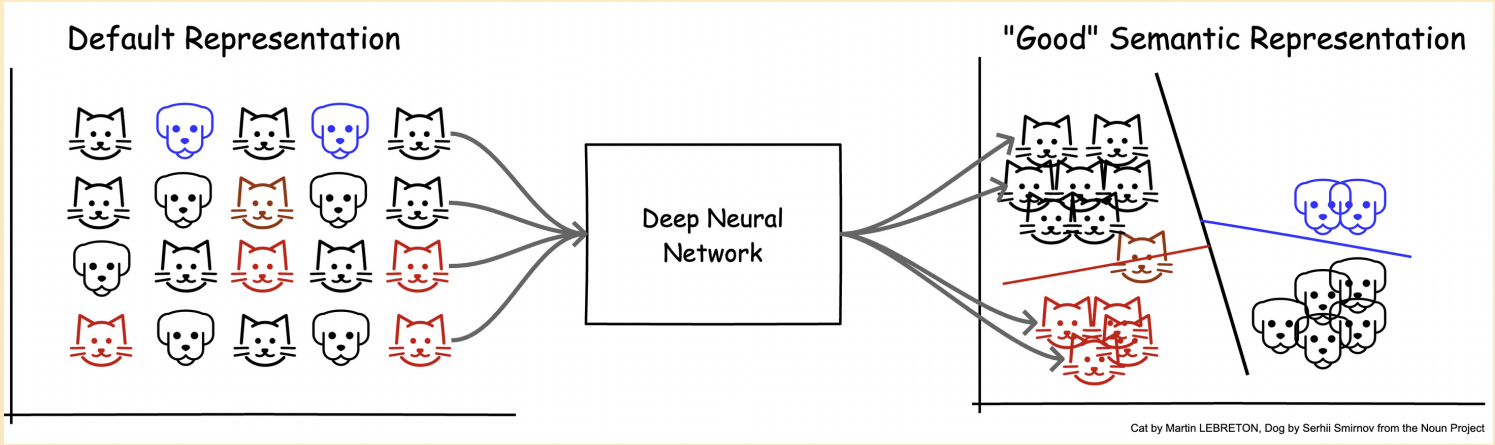
- Increasing the class of allowed functions, it's more likely to find a smooth function with lower norm (complexity): Occam's razor

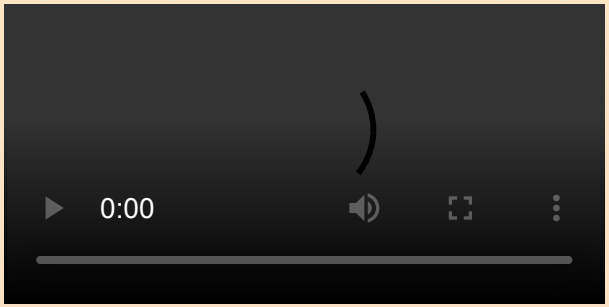


Mapping Improves Understanding

Tabula Rogeriana by Al Idrisi (1154) 1929 reproduction

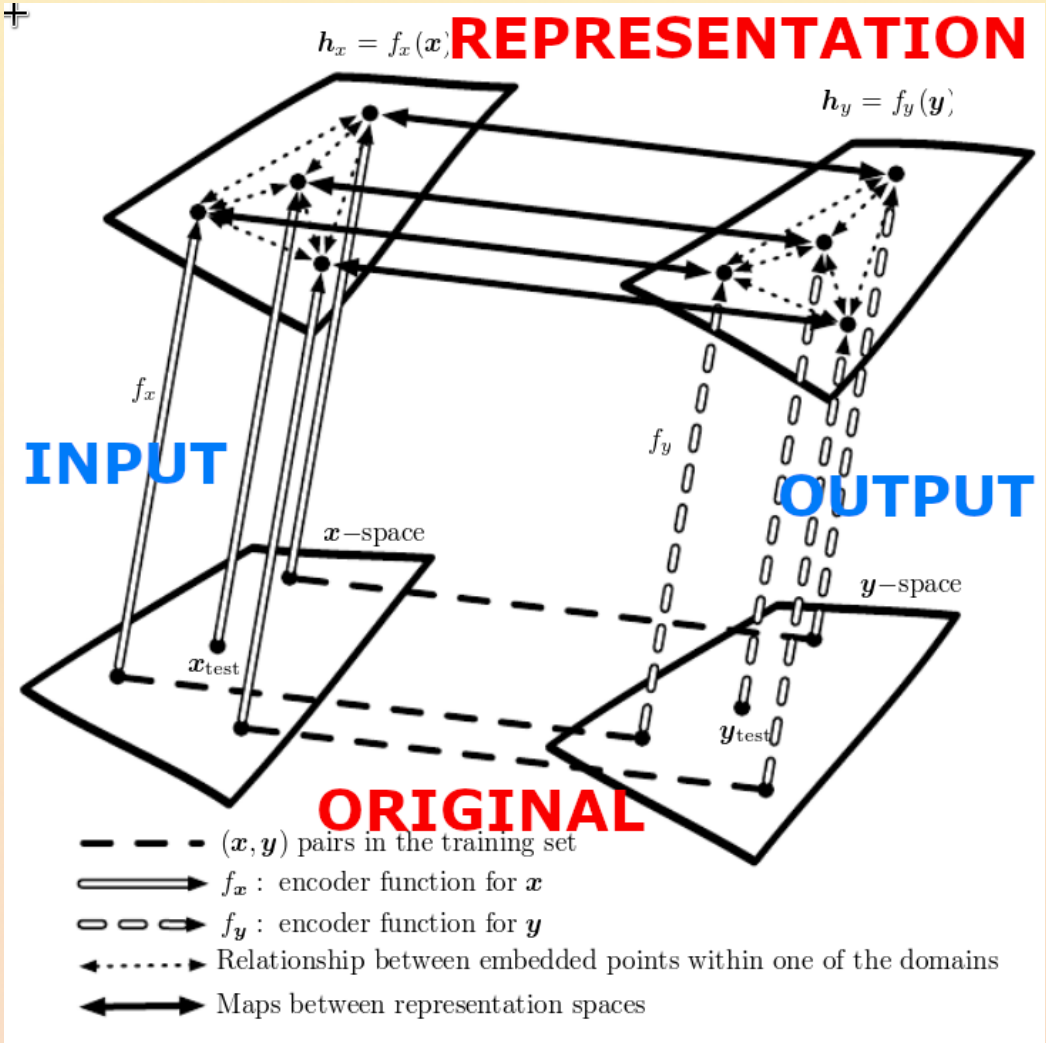
Representations Make Tasks Easier





Animation and picture from FastForward Labs

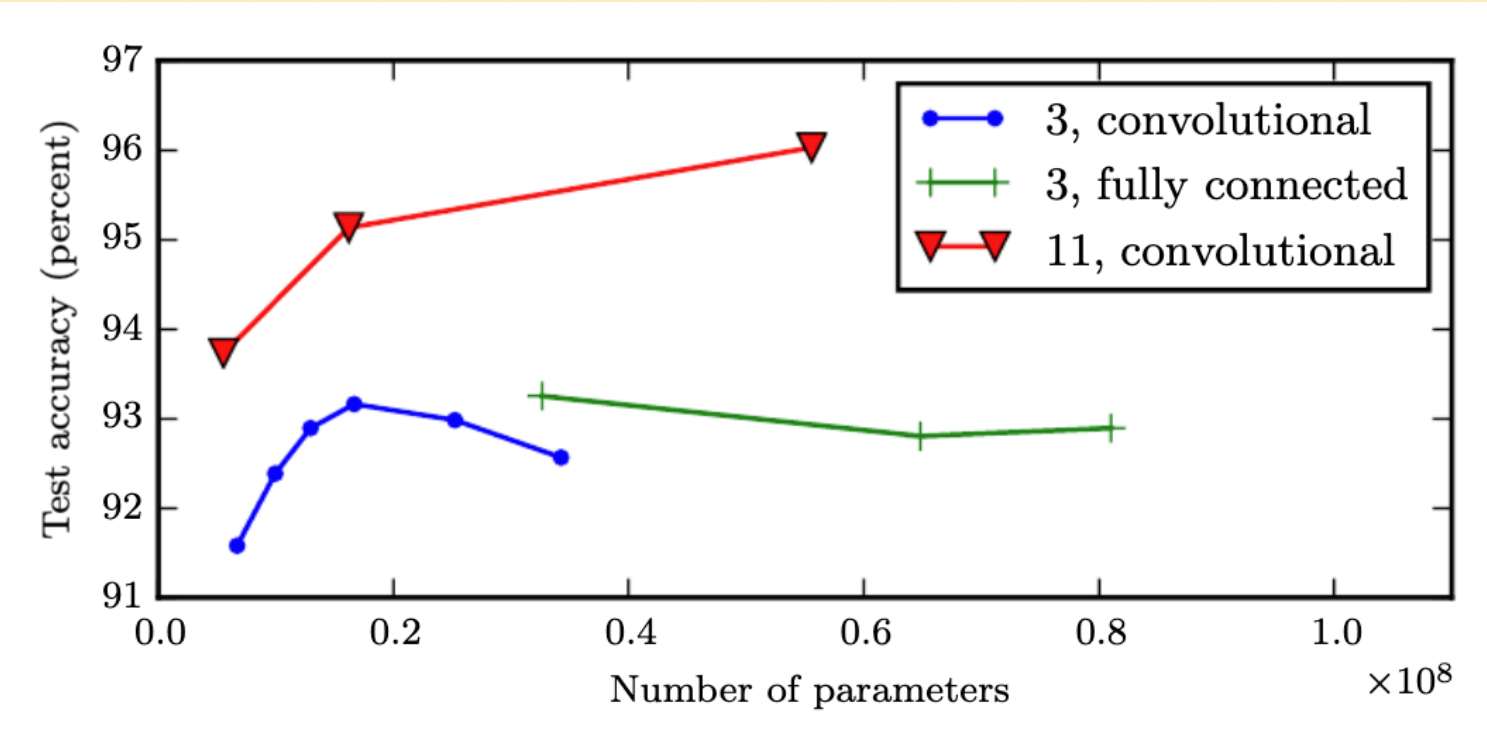
Learn Representations



Images from Goodfellow, Bengio, Courville, 2016

Number of parameters

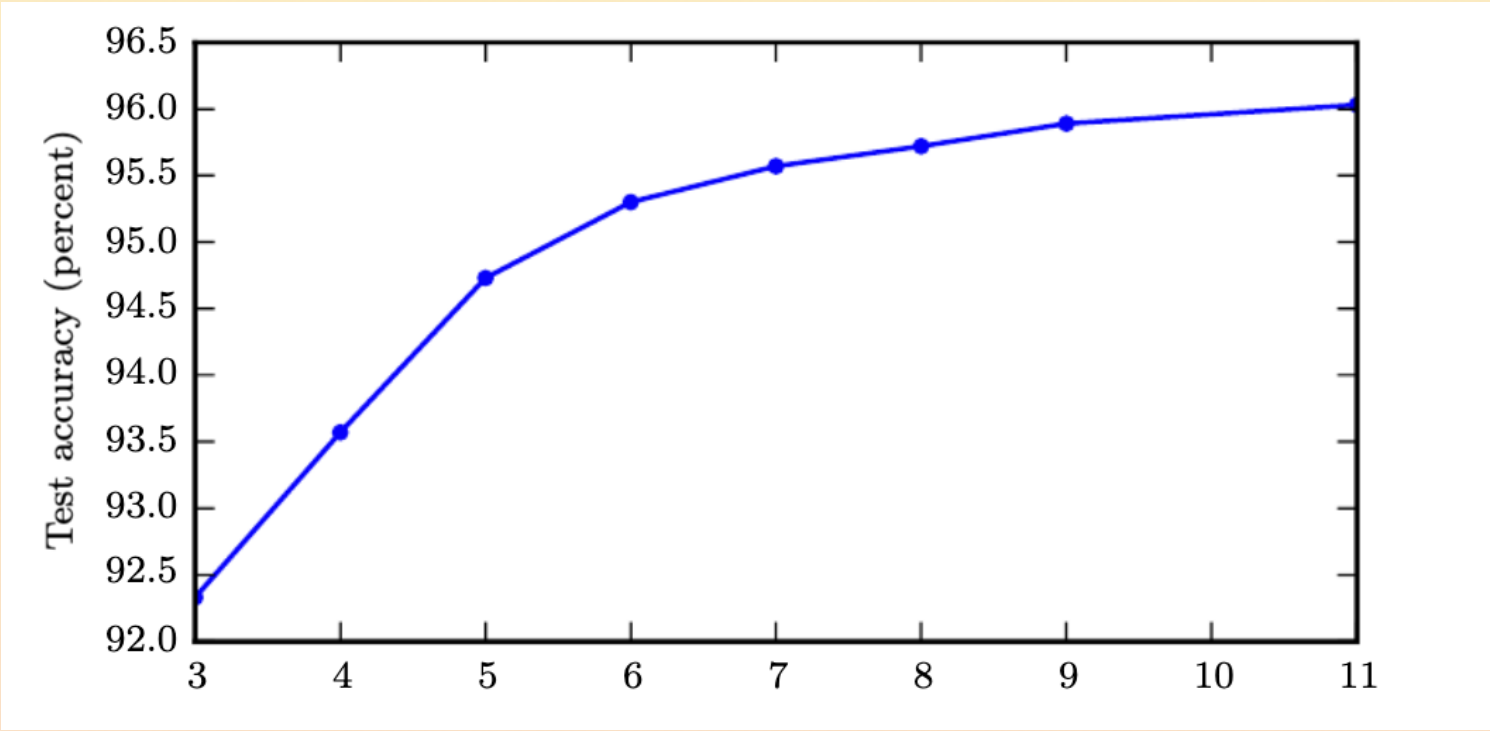
- Empirical studies: increasing number of parameters doesn't help beyond a certain point



Images from Goodfellow, Bengio, Courville, 2016

Depth

- Empirical studies: increasing depth tends to always result in some improvement

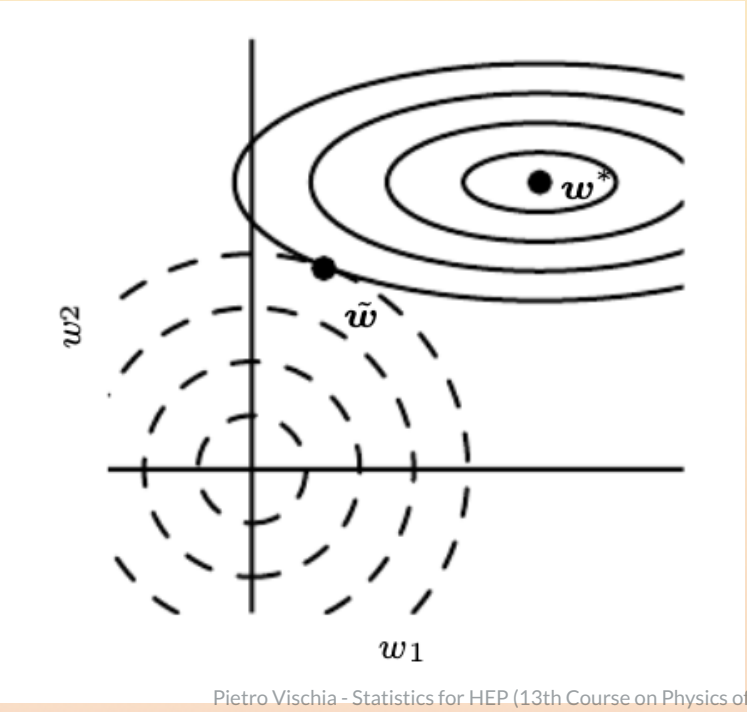


Images from Goodfellow, Bengio, Courville, 2016

Regularization: weight decay

- Another way of regularizing is via weight decay (L^2 regularization)
  - Tradeoff between good fitting (small MSE) and small norm (smaller slope, or fewer features with large weights)
  - In statistics, "ridge regression", "Tikhonov regularization"

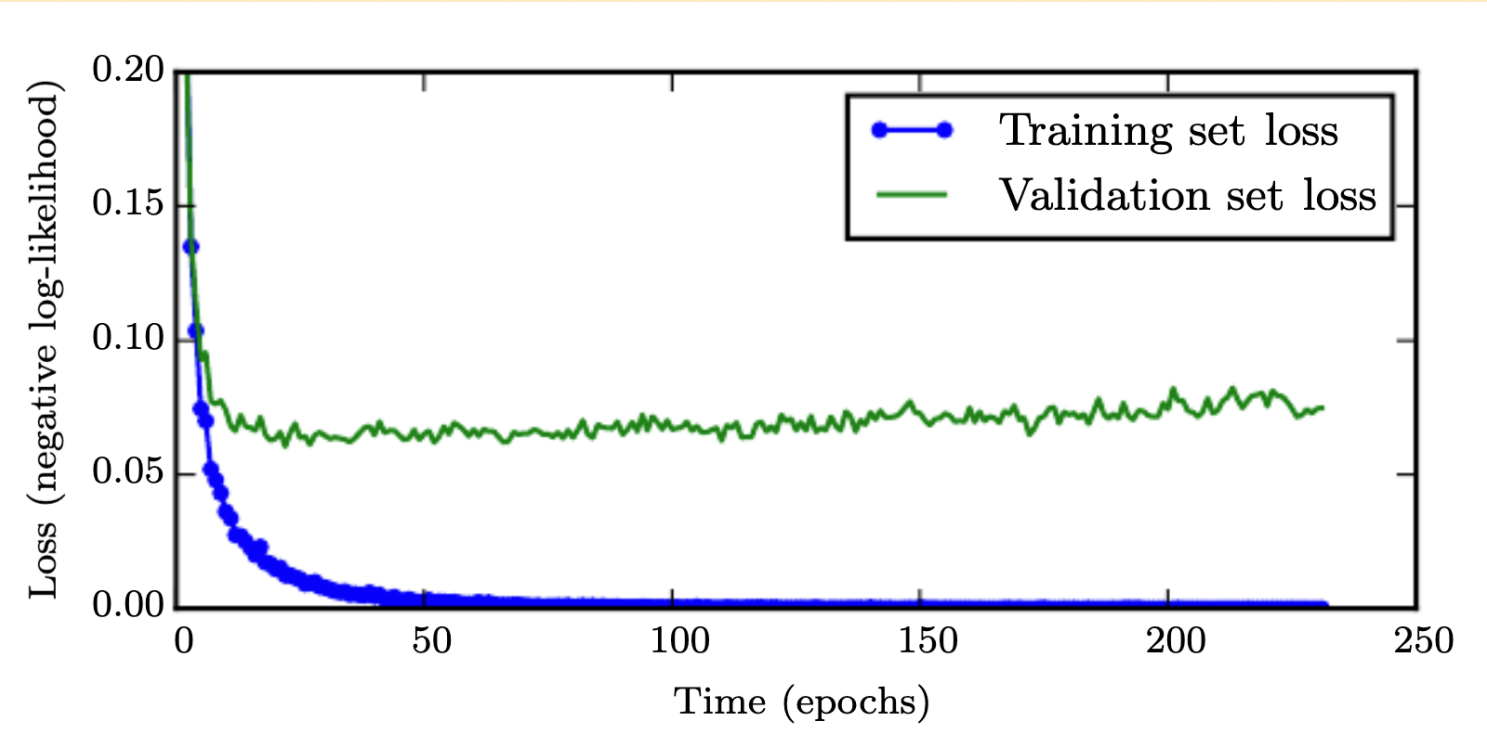
$$J(\mathbf{w}) = MSE_{train} + \lambda \mathbf{w}^T \mathbf{w}$$



Images from Goodfellow, Bengio, Courville, 2016

Early stopping...

- Train until the validation set loss starts increasing, and pick the model corresponding to the minimum validation loss

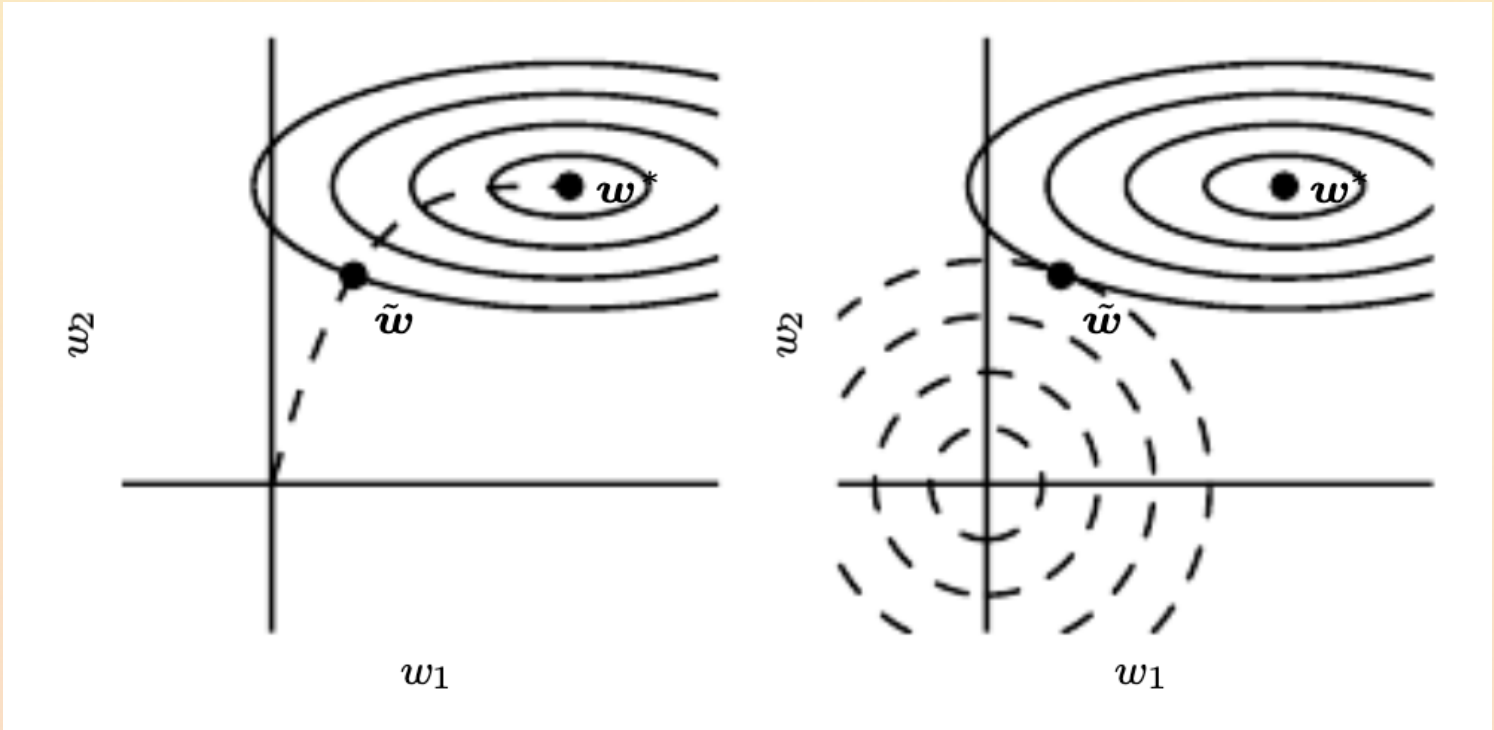


Images from Goodfellow, Bengio, Courville, 2016

...is a form of regularisation

- Early stopping limits the reachable phase space, and is therefore analogous to L2 regularization (weight decay)

$$J(\mathbf{w}) = MSE_{train} + \lambda \mathbf{w}^T \mathbf{w}$$



Images from Goodfellow, Bengio, Courville, 2016

Many ways of inserting our biases

- Regularization corresponds to inserting our bias into the algorithm
  - "I know that the solution should not wiggle", "I know that the curvature must not be too large"
- Two models A and B performing the same classification task
  - $\hat{y}^{(A)} = f(\mathbf{w}^{(A)}, \mathbf{x})$
  - $\hat{y}^{(B)} = f(\mathbf{w}^{(B)}, \mathbf{x})$
- If the inputs distributions are somehow different, but we know (or want that) the output are related, we can assume that the weights should be similar

$$J(\mathbf{w}) = MSE_{train} + \lambda ||\mathbf{w}^{(A)} - \mathbf{w}^{(B)}||_2^2$$

Parameter sharing

- Simply require parameters are equal
  - If they are equal, you can store only one number in memory (sometimes dramatic memory footprint reduction)

–
Images from goodhousekeeping.com

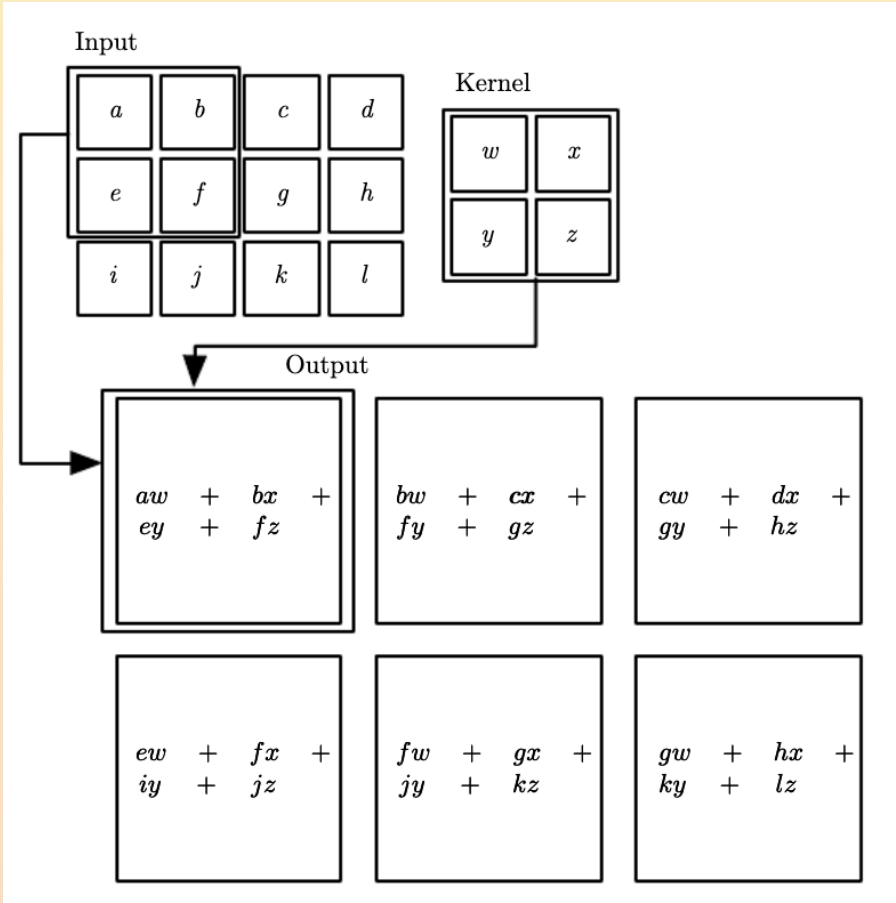
Convolution: a form of averaging

$$s(t) = \int x(a) w(t-a) da$$

- When discretized, integral becomes a sum
  - x input
  - w kernel: specifies how far does the averaging goes
  - s feature map

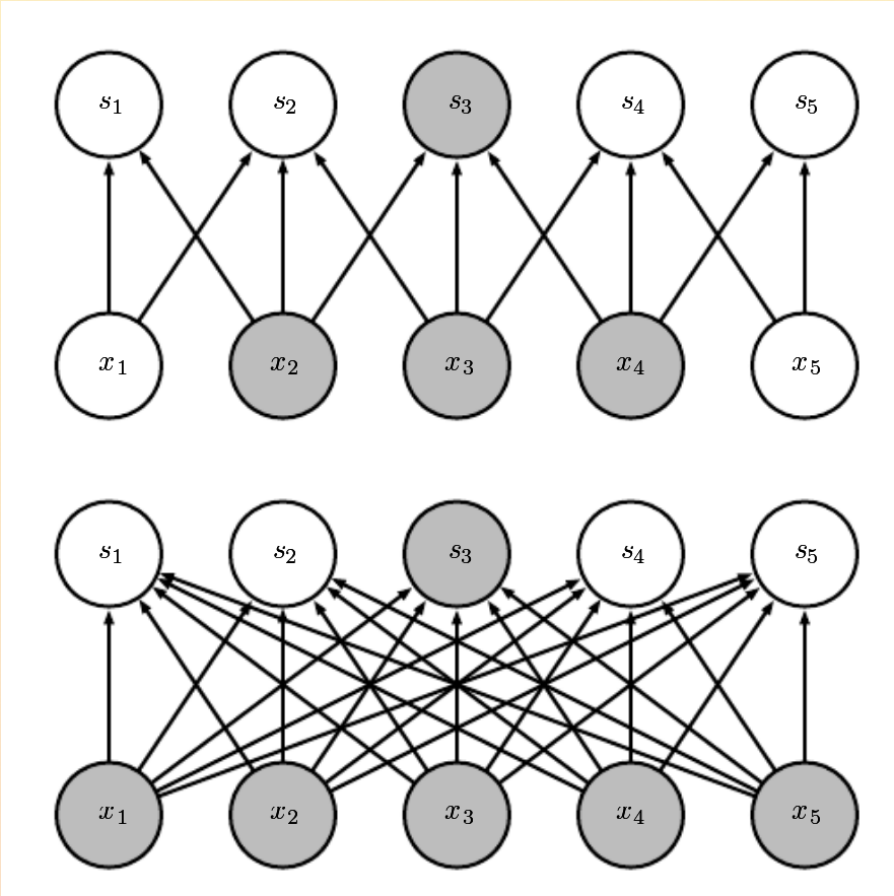
Convolution: a form of averaging

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n) K(m,n)$$



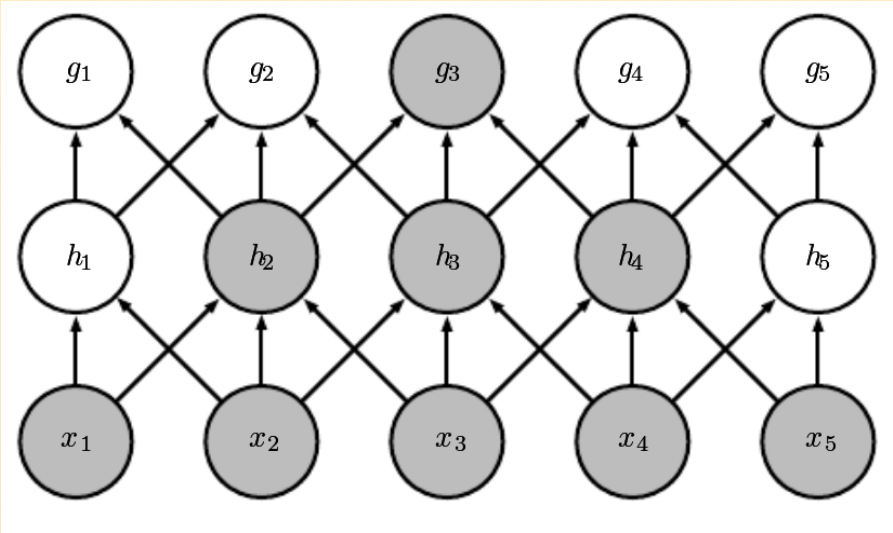
Images from Goodfellow, Bengio, Courville, 2016

Receptive field



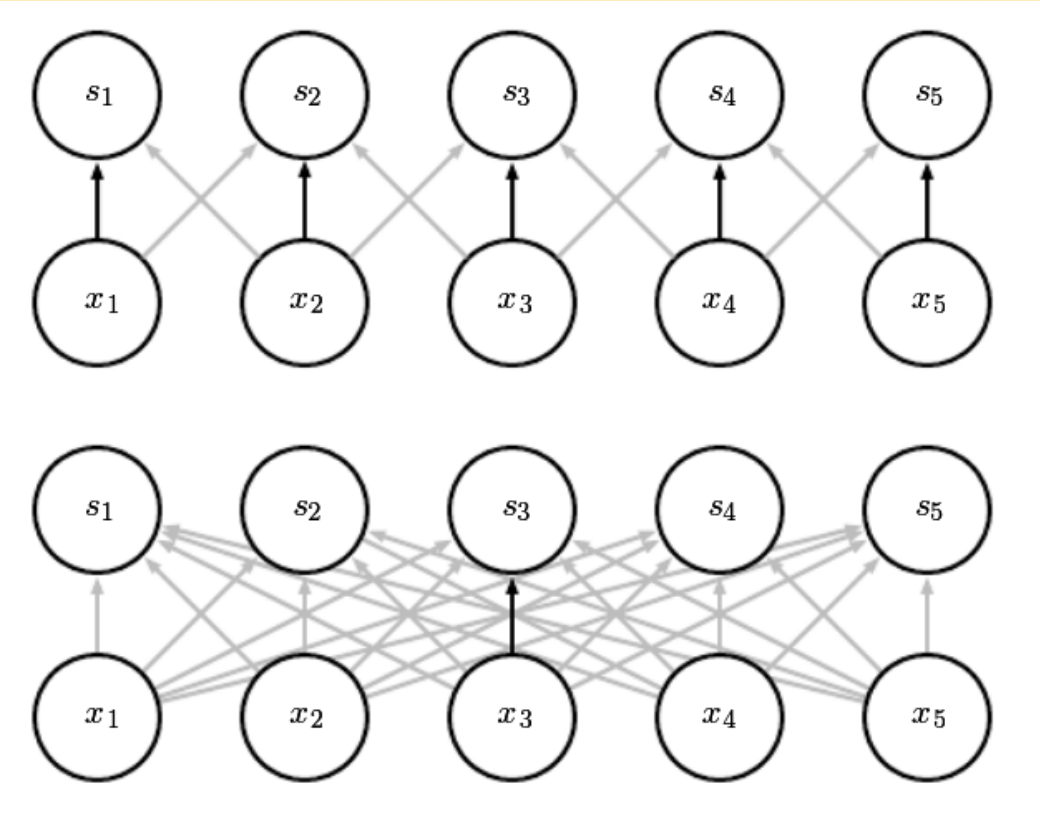
Images from Goodfellow, Bengio, Courville, 2016

Receptive field: deeper = larger



Images from Goodfellow, Bengio, Courville, 2016

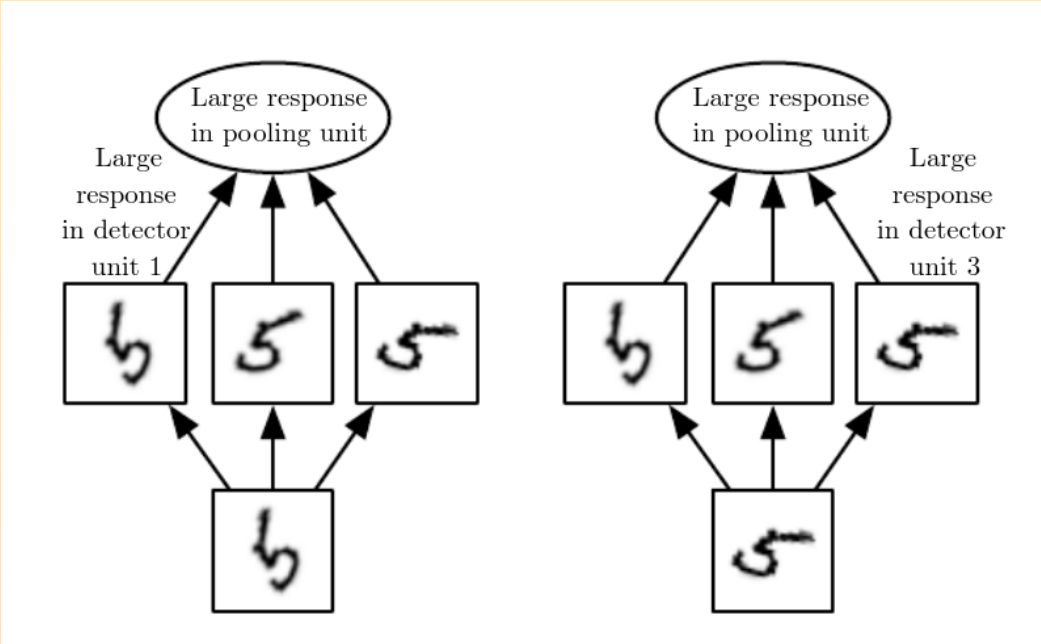
Parameter sharing



Images from Goodfellow, Bengio, Courville, 2016

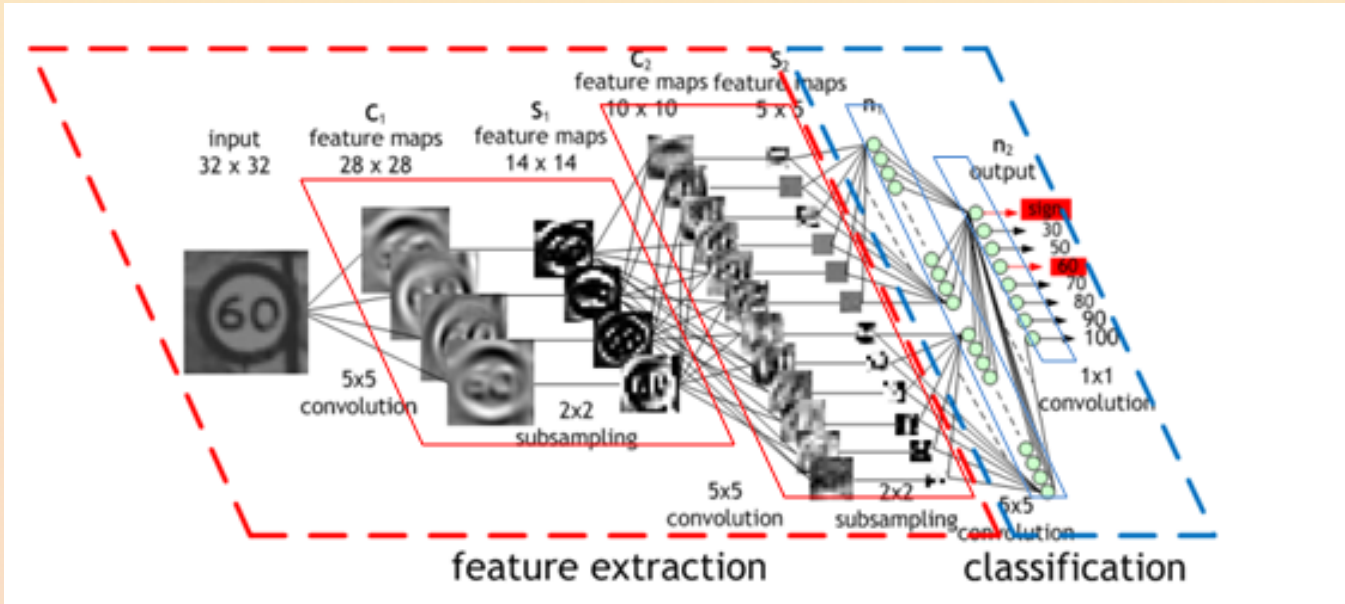
Convolutional network

- Convolution $\longrightarrow$ nonlinear activation $\longrightarrow$ pooling
- Pooling: replace output at a location with a summary statistic
  - e.g., max pooling = report the maximum output in a neighbourhood
  - Helps with invariance for translations



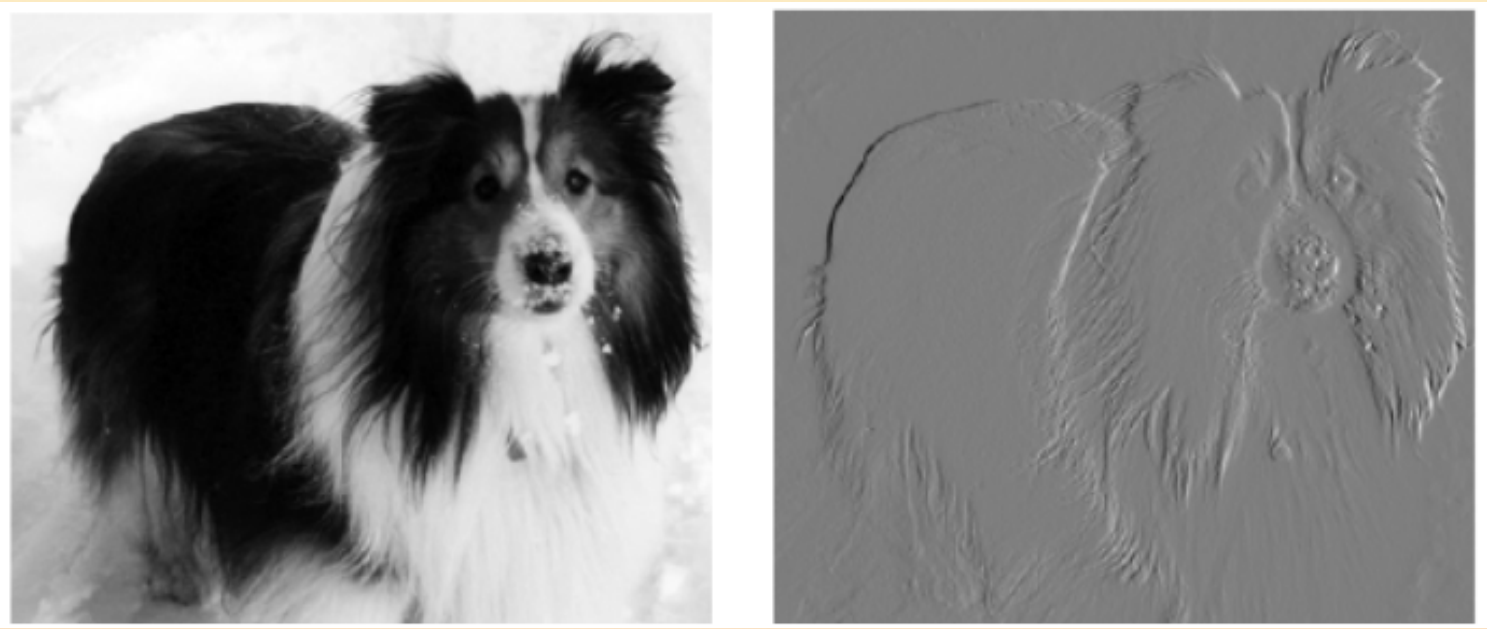
Images from Goodfellow, Bengio, Courville, 2016

Convolutional networks



Images from tue.nl and Yann LeCun's LeNet

Intermediate representations



Images from Goodfellow, Bengio, Courville, 2016

Morphology of galaxies

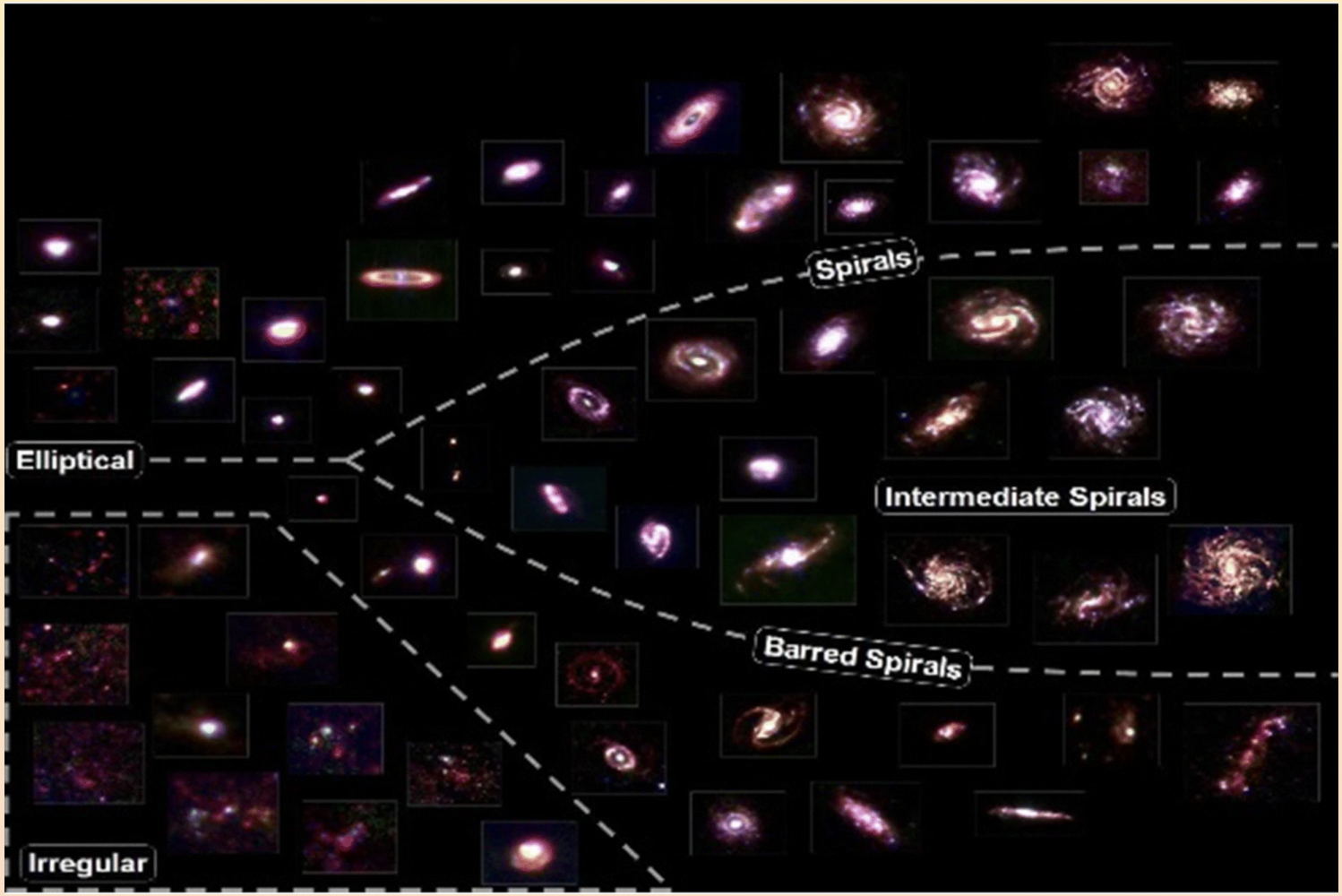
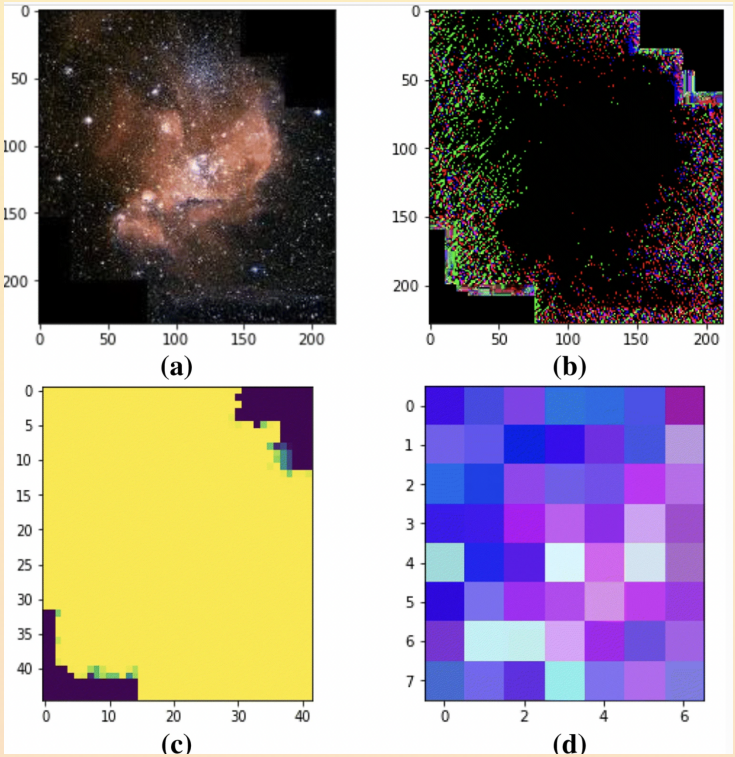


Image from Earth Science Informatics volume 13, pages 601–617 (2020)

Representations of galaxies...



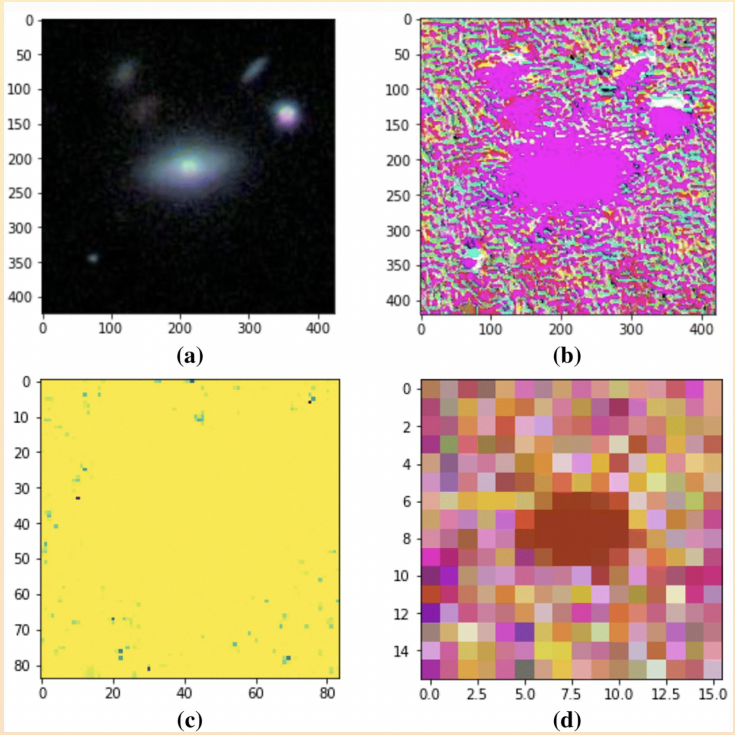


Image from Earth Science Informatics volume 13, pages 601–617 (2020)

...work pretty well

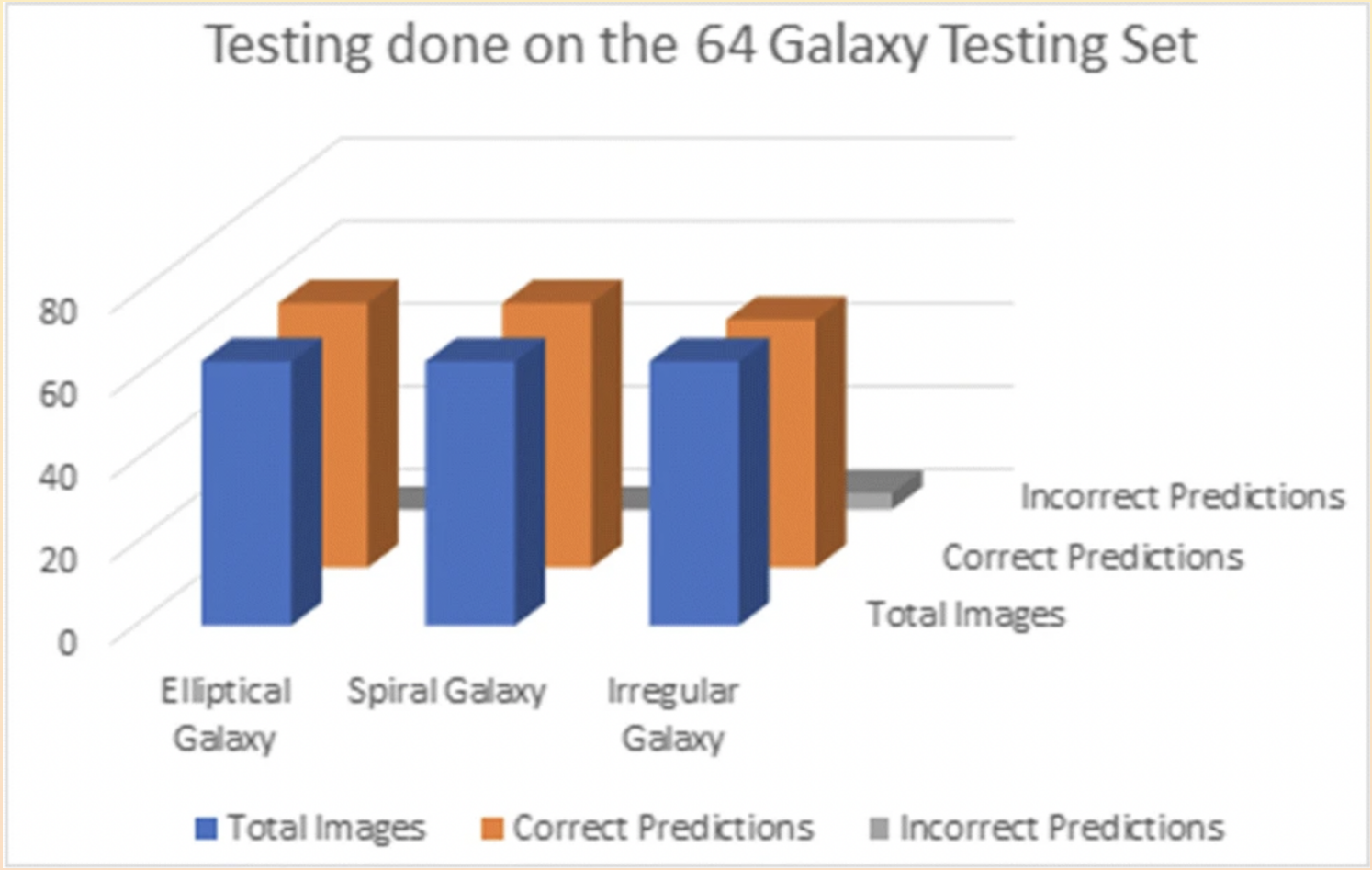


Image from Earth Science Informatics volume 13, pages 601–617 (2020)

Semantic representations

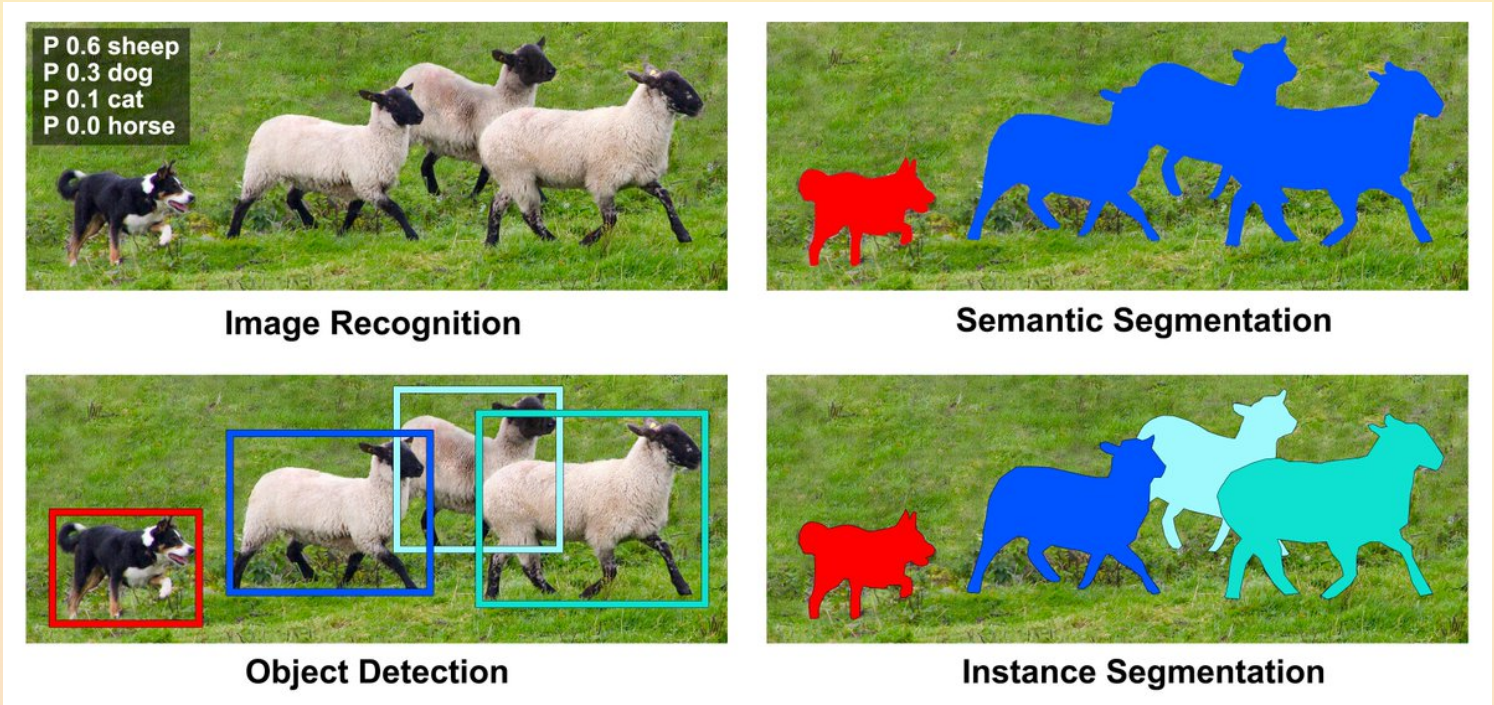


Image from ai-pool.com

What about time?

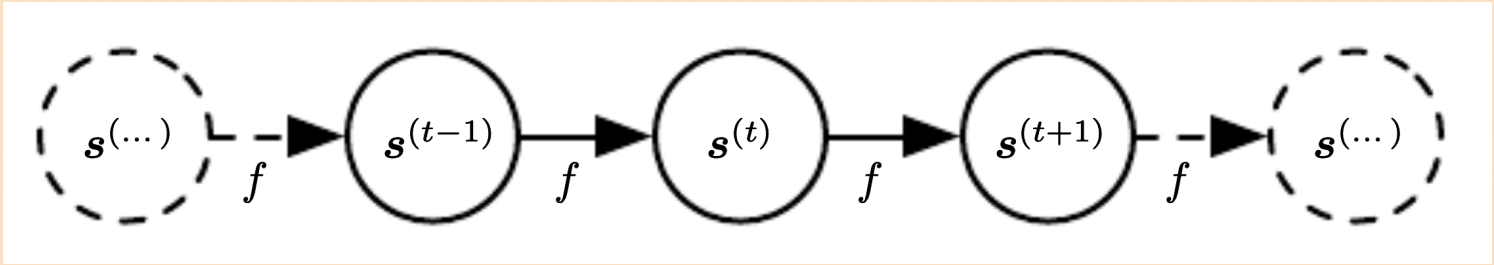
- Convolutional network: process grid of values (e.g. images)
- Recurrent networks: process a sequence of values indicised by a "time" component
  - Language is a sequence
- Parameter sharing crucial to generalize:
  - lengths unseen in training
  - different positions in the sentences
- Without parameter sharing, a network would have to learn all the language rules at each step of the sequence
  - Very impractical
- Both scale very well (thanks to parameter sharing)

Convolutional networks for sequences?

- Could "link" the steps of the sequence via the convolution
- Use the same kernel at each time step
- Shallow: it links only neighbouring time steps

Recurrent network

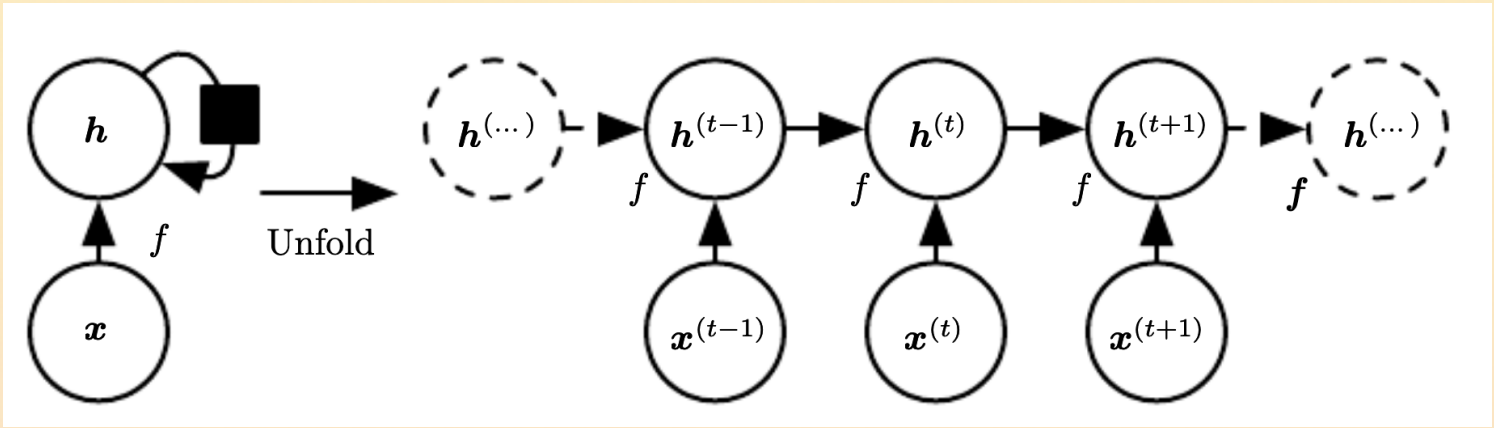
- Use the same parameter at the same step, $s^{(t)} = f(s^{(t-1)}, \theta)$
  - Very deep structure



Images from Goodfellow, Bengio, Courville, 2016

Unfold the graph

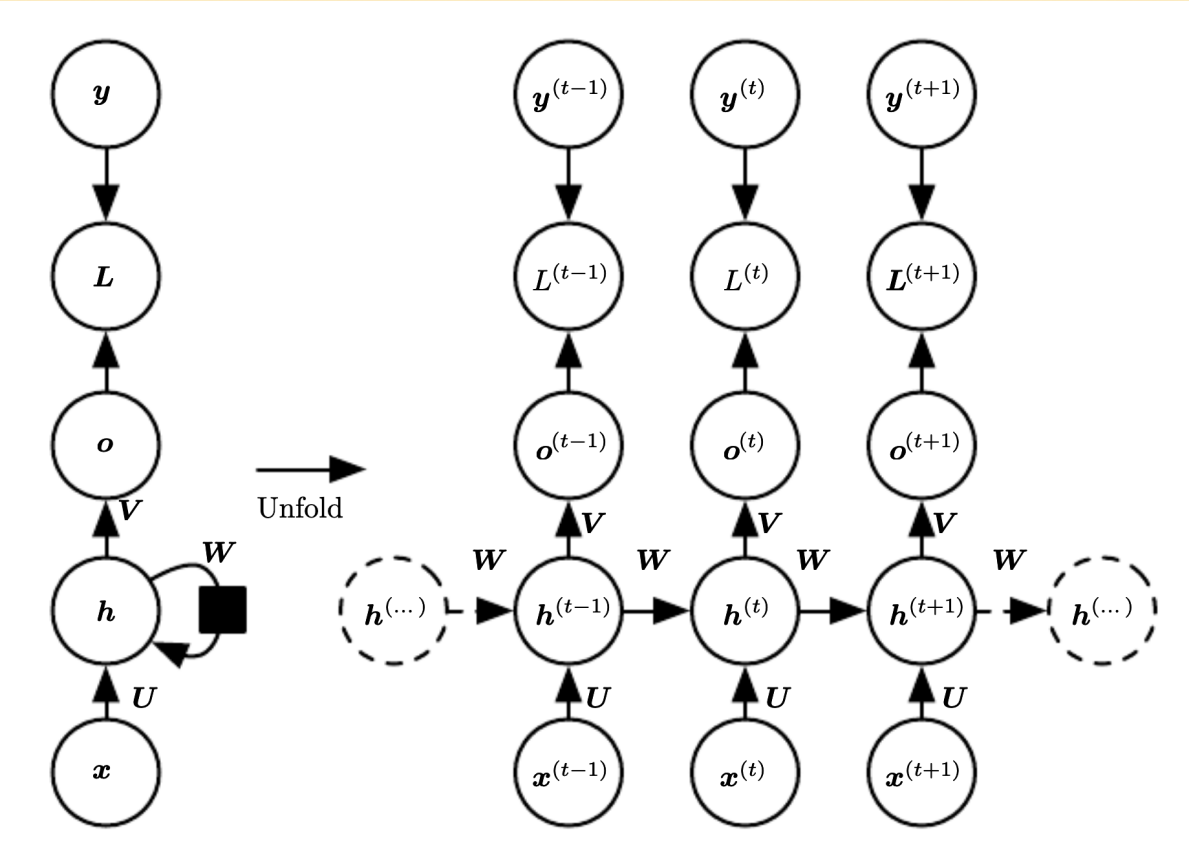
$$s^{(3)} = f(s^{(2)}, \theta) = f(f(s^{(1)}, \theta), \theta)$$



Images from Goodfellow, Bengio, Courville, 2016

Vast zoology

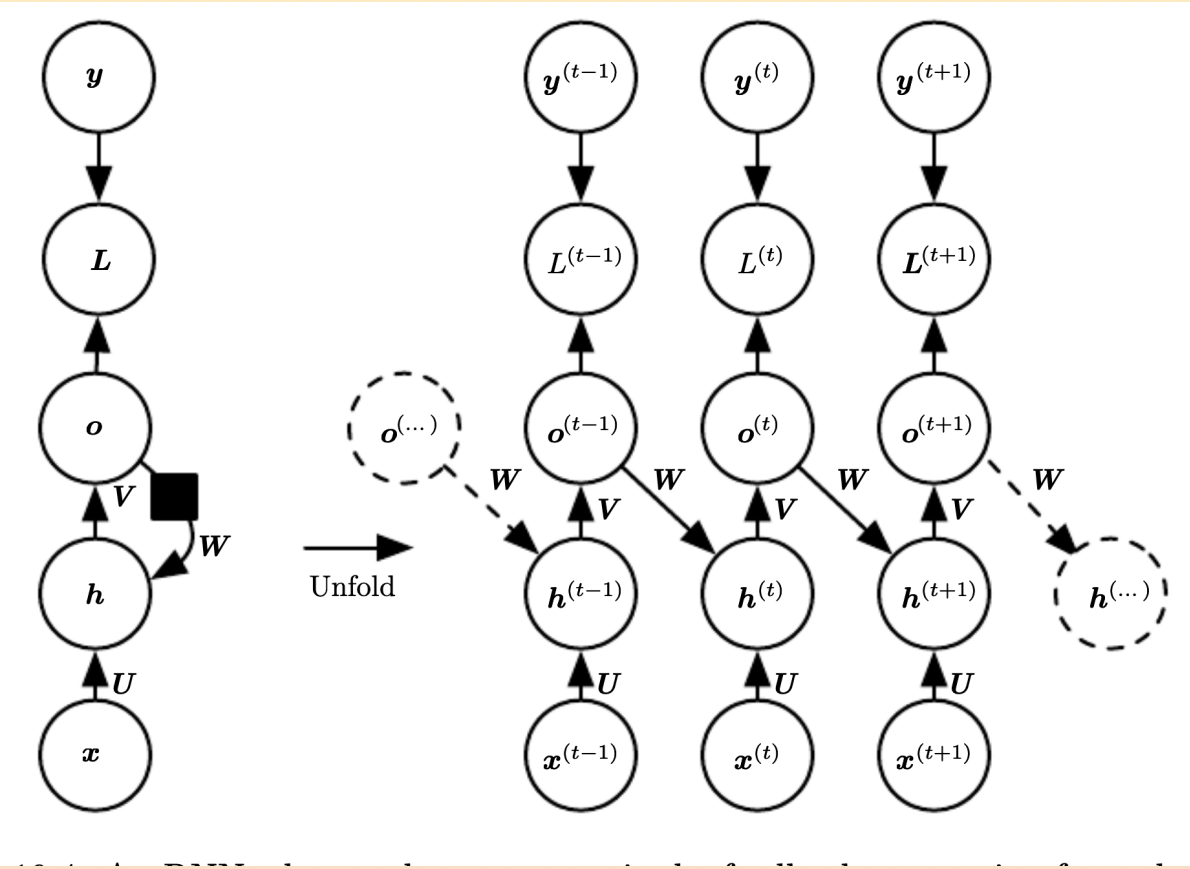
= An output at each time step, recurrent connections between hidden units



Images from Goodfellow, Bengio, Courville, 2016

Vast zoology

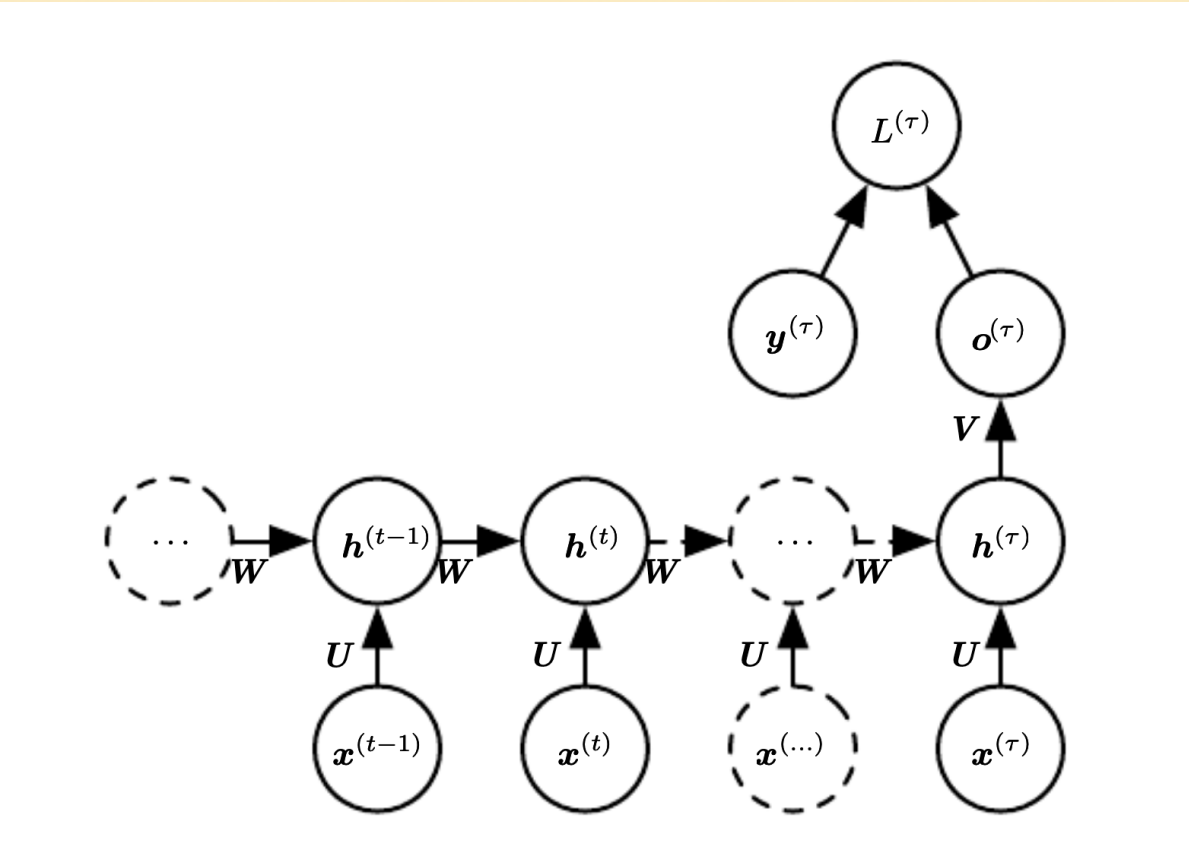
- An output at each time step, recurrent connections only from the output at one time step to the hidden units at the next time step



Images from Goodfellow, Bengio, Courville, 2016

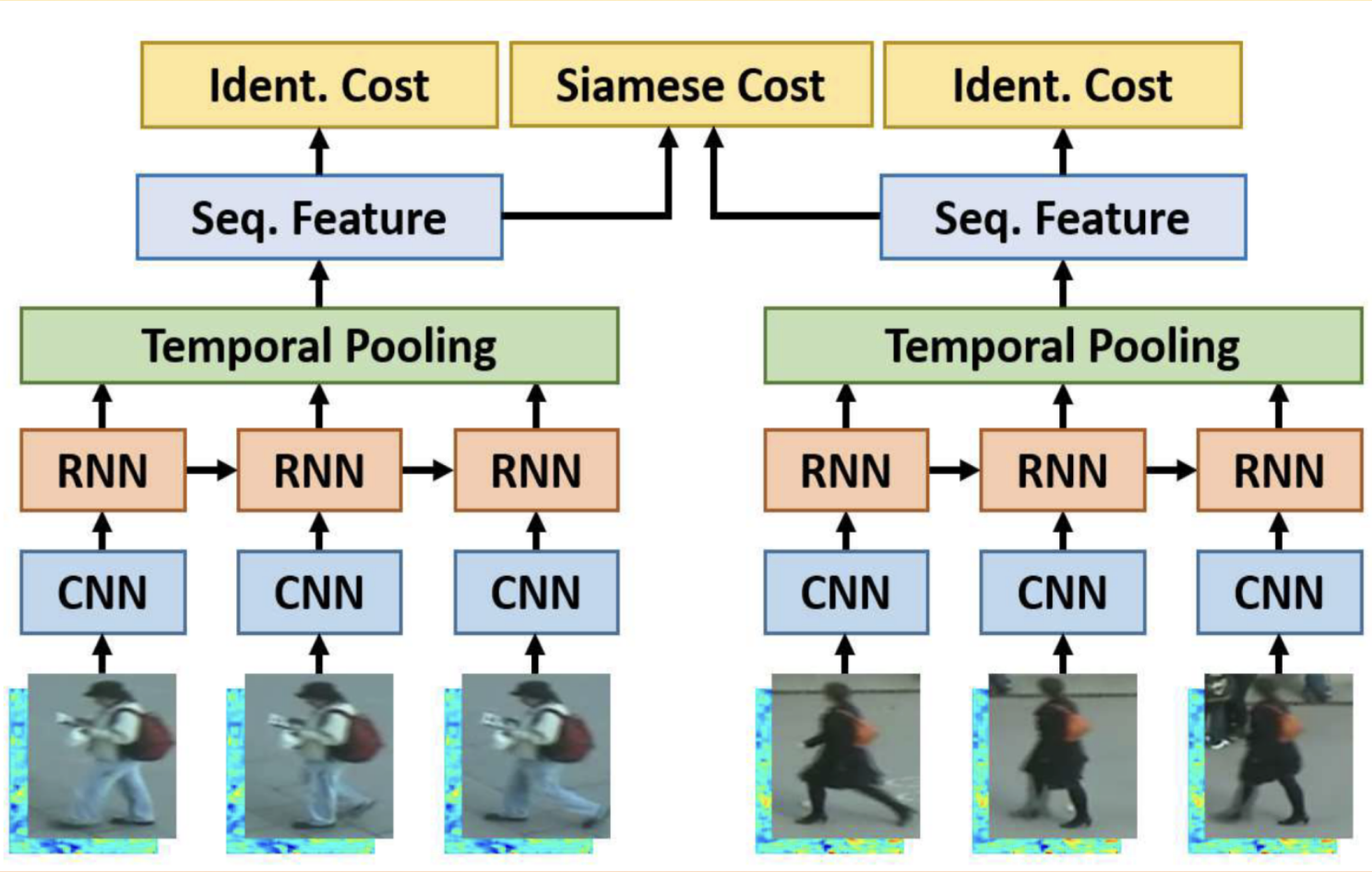
Vast zoology

- Recurrent connections between hidden units, that read an entire sequence and then produce a single output



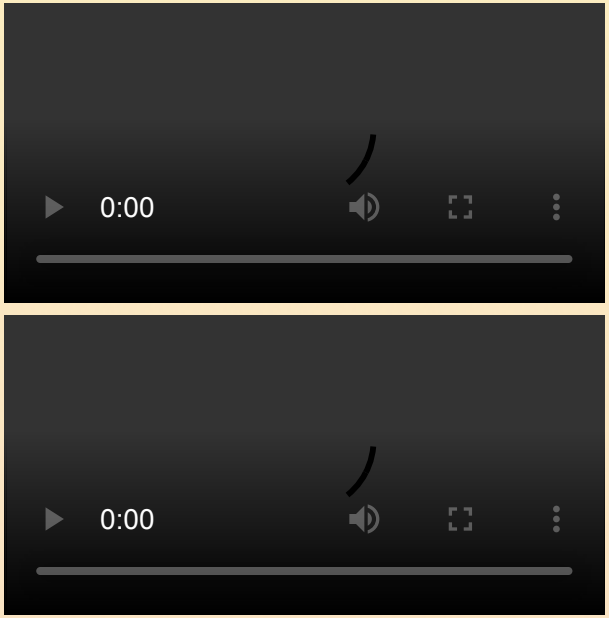
Images from Goodfellow, Bengio, Courville, 2016

Sequences of images



Picture from 10.1109/CVPR.2016.148

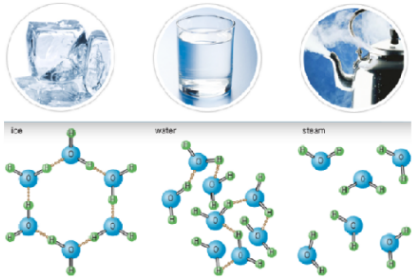
Real-time segmentation



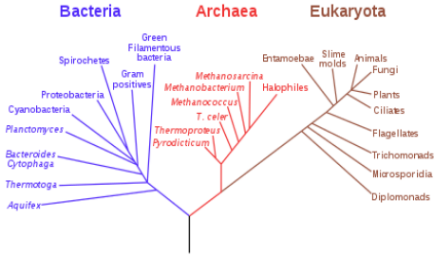
Videos from YouTube, autonomous driving and YouTube, cancer research

Graphs Represent Structure

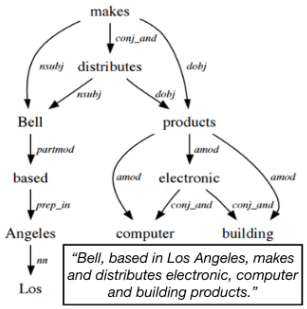
Molecules



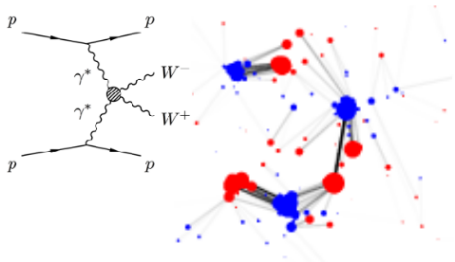
Biological species



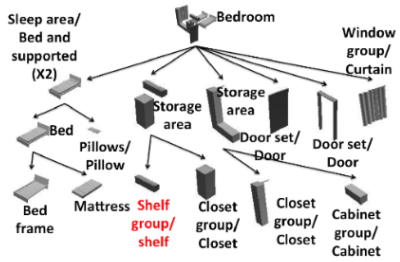
Natural language



Sub-atomic particles



Everyday scenes



Code

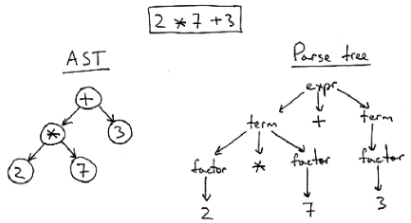
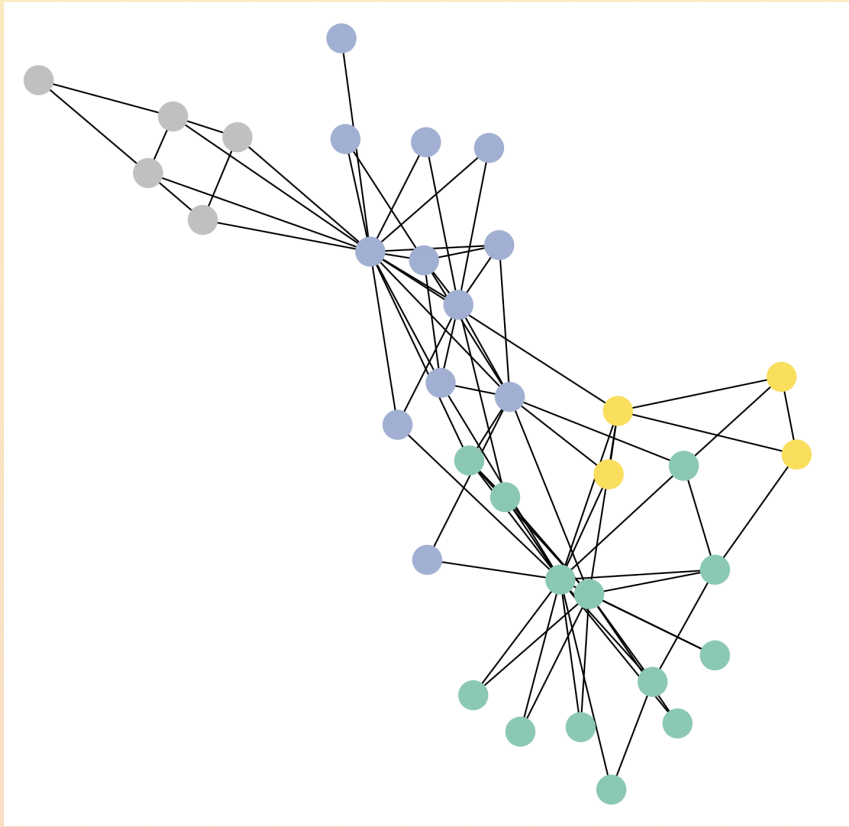


Image from Peter Battaglia

Graph networks

- Represent data as point clouds

- Connect data points with weight-dependent connections
- Train the network to find which weights are strongest
    - Learng the connectivity structure of the data



—
Image by me

CMS High-granularity calorimeter

- 6 million cells with $\sim 3mm$ spatial resolution, over $600m^2$ of sensors
- Non-projective geometry

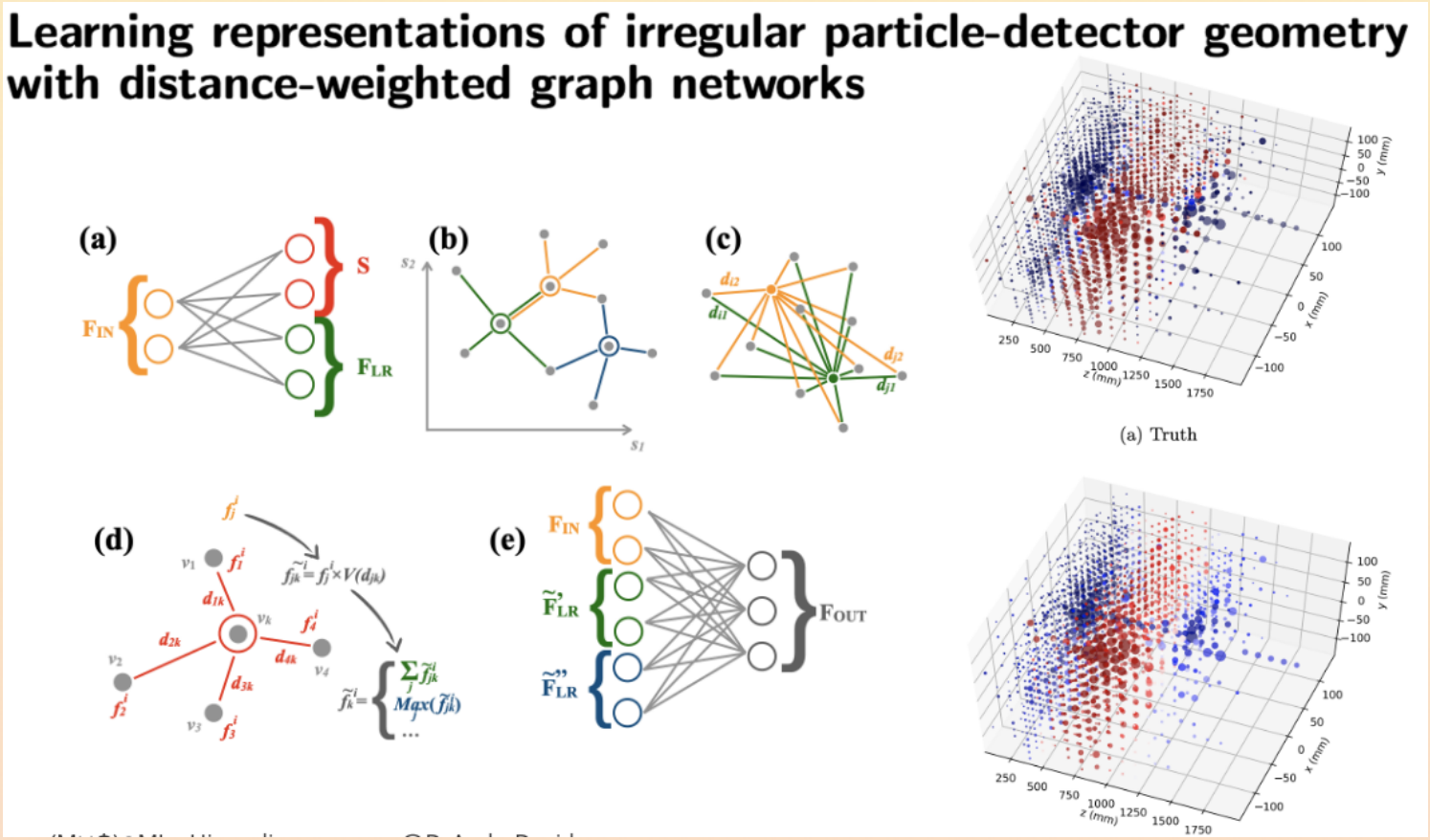
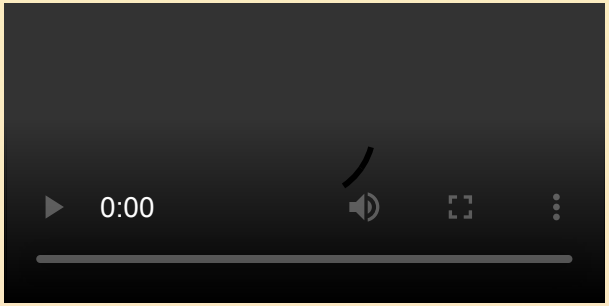


Image from the CMS HGCAL Team

Graphs for water simulation

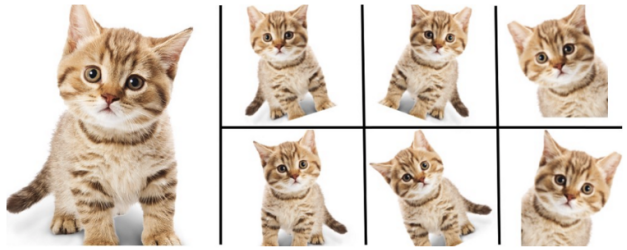


Video by Peter Battaglia's team

Plug the Physics into the AI

How are symmetries implemented?

- Data augmentation
  Li, Dobriban '20
- Loss function penalties
  - conserved quantities
- Architectural design
  - Approximate symmetries (CNN)
  - Exact symmetries
    - Weight sharing (message passing) (group convolutions) Cohen, Welling '19 Ravanbakhsh Rose YU '21, '20. Weiler '21
    - Parameterization of symmetry preserving functions Kondor '18 Maron '18 Cohen '18
    - Symmetries as constraints Finzi et al '21
    - Irreducible representations Kondor, Thomas '18, Fuchs '20 Smidt
    - Steerable CNNs Cohen '17, Welling..



Enlarge your Dataset

Credit: Bharath Raj

Illustration by Soledad Villar

Plug the physics into the AI: constraints

$$\hat{y} = f(\mathbf{x}, \theta)$$

- Encode physics knowledge (e.g. inconsistency of models) inside the loss function as a penalty term

$$\mathbf{J}(\mathbf{w}) = Loss(y, \hat{y}) + \lambda ||\mathbf{w}||_2^2 + \gamma \Omega(\hat{y}, \Phi)$$

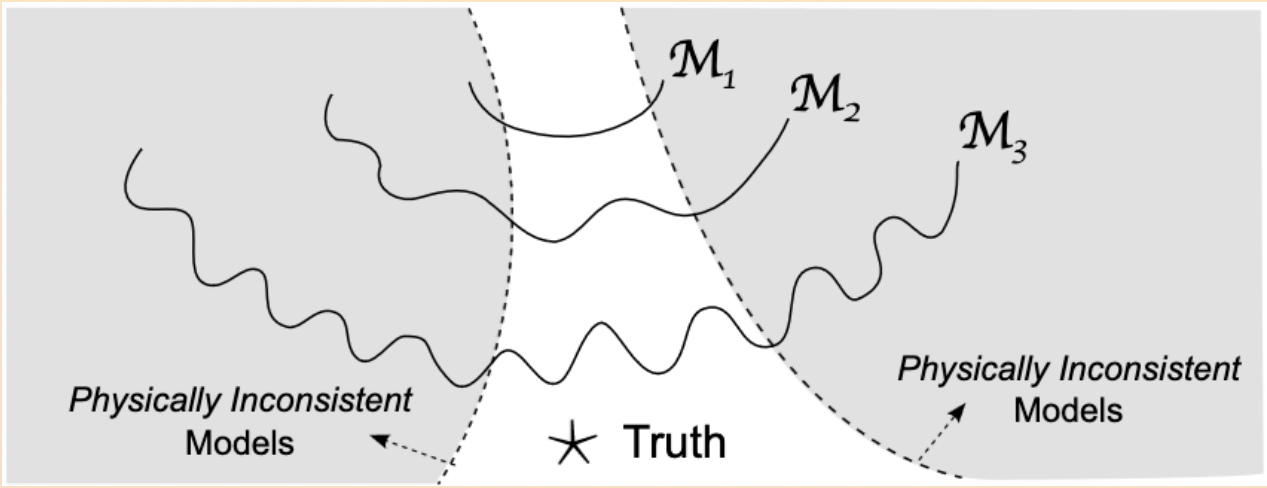


Figure from 10.1109/TKDE.2017.2720168

Plug the physics into the AI: network structure

- Equivariance under group transformation can e.g. enforced by convolutional layers
- Some implementations available in pytorch

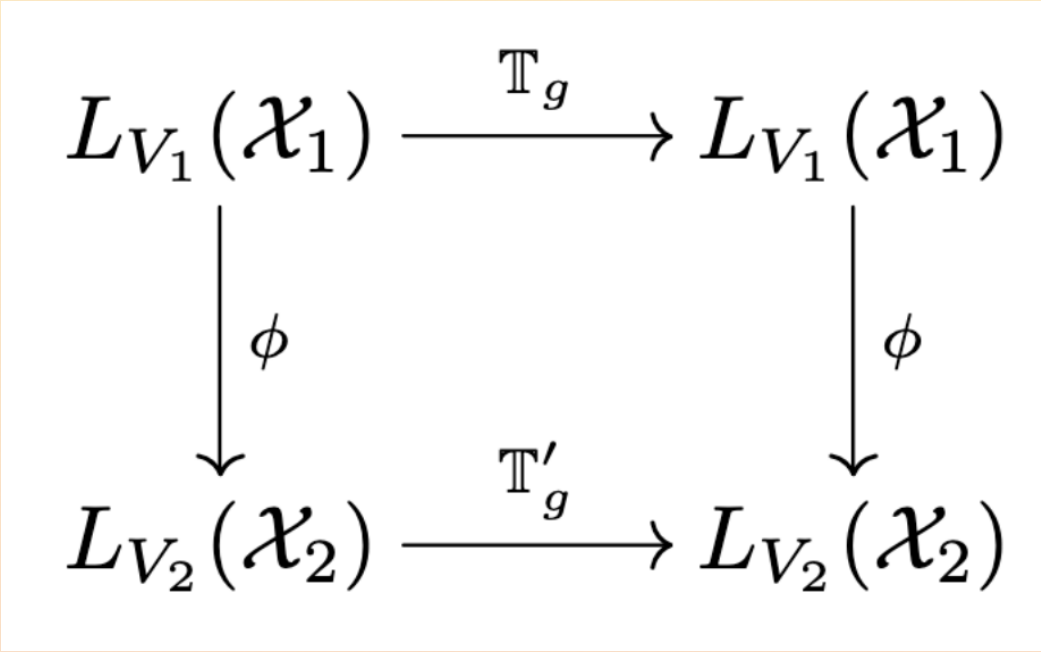
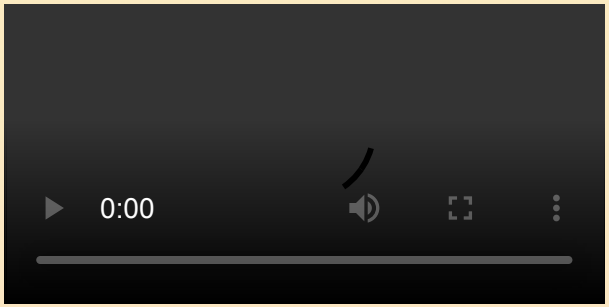


Figure from 10.1109/TKDE.2017.2720168

Plug the Physics into the AI

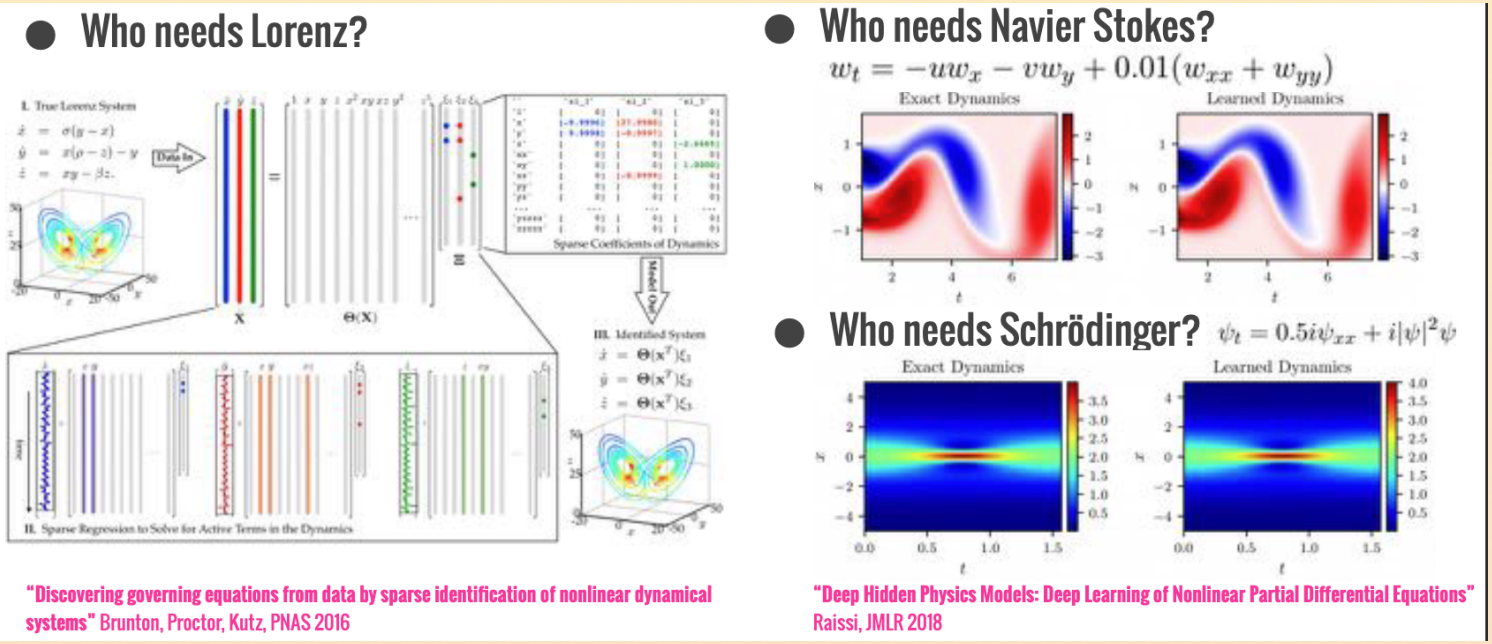
- Physics-aware differential equations solving



Animation from 2202.06988

Plug the Physics into the AI

- Several ODE problems now solvable via neural networks



"Discovering governing equations from data by sparse identification of nonlinear dynamical systems" Brunton, Proctor, Kutz, PNAS 2016

"Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations" Raissi, JMLR 2018

Figure by G. Camps-Valls

Autoencoders

- Learn the data itself passing by a lower-dimensional intermediate representations
  - Capture data generation features into a lower-dimensional space
- Can use for anomaly detection
- Can sample from the latent space to obtain random samples (generative AI)

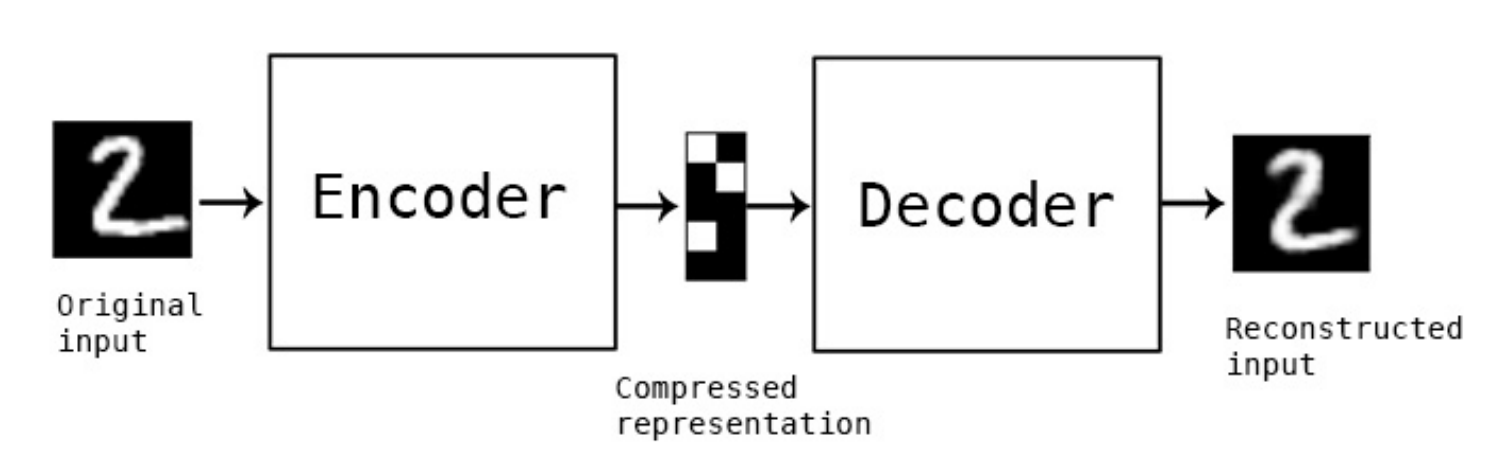
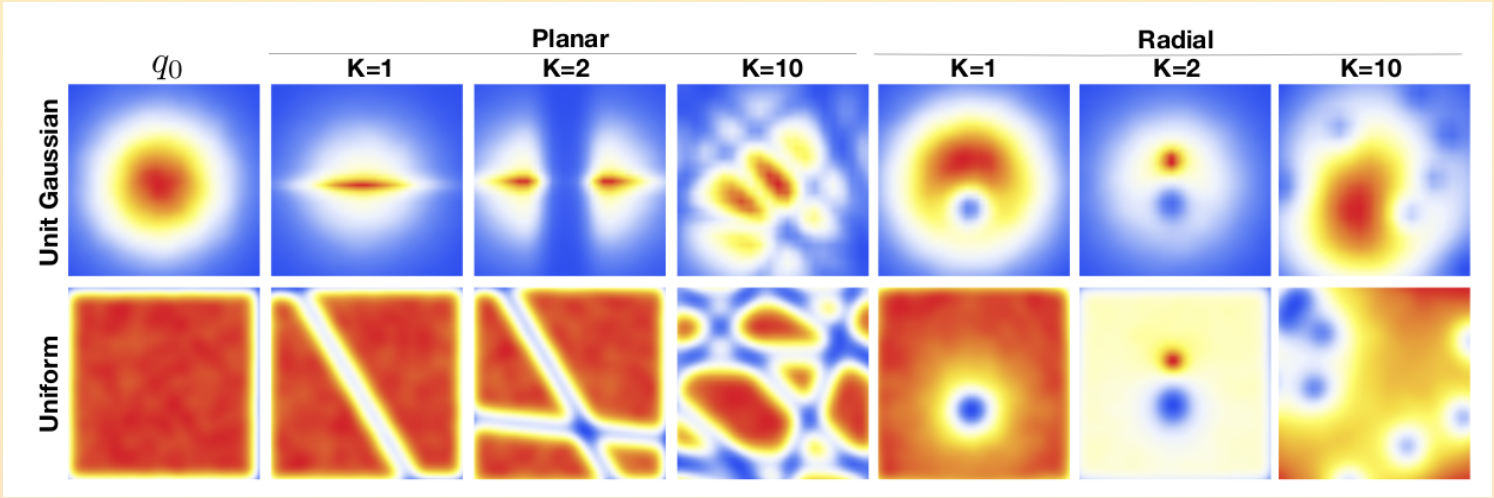
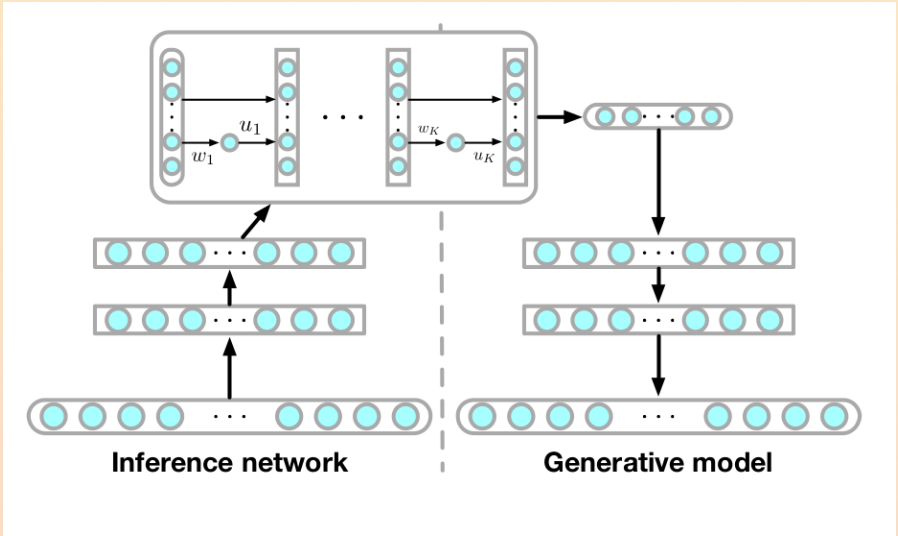


Figure from...

Invertible networks





Images from Rezende and Mohamed, 1505.05770

Solve inverse problems ("unfolding")

- Correct detector observation noise to recover source distribution

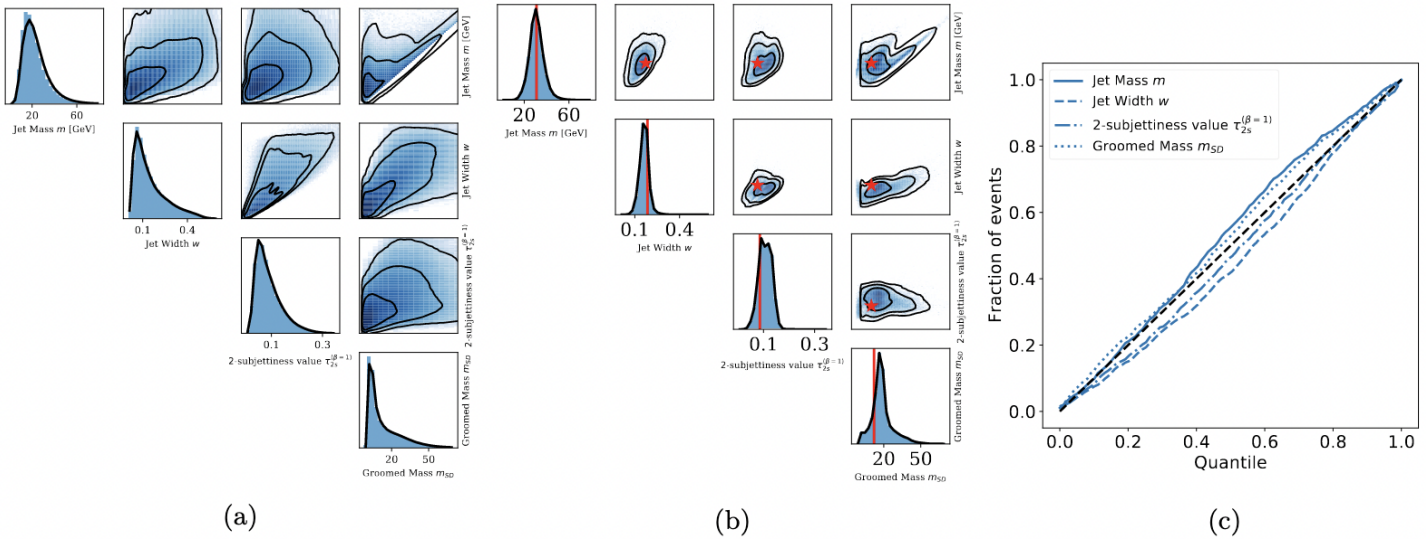
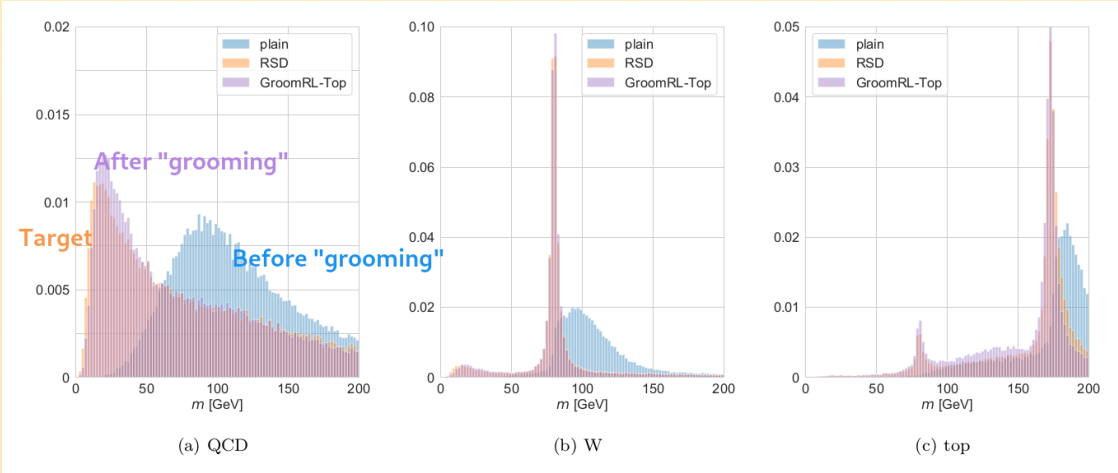
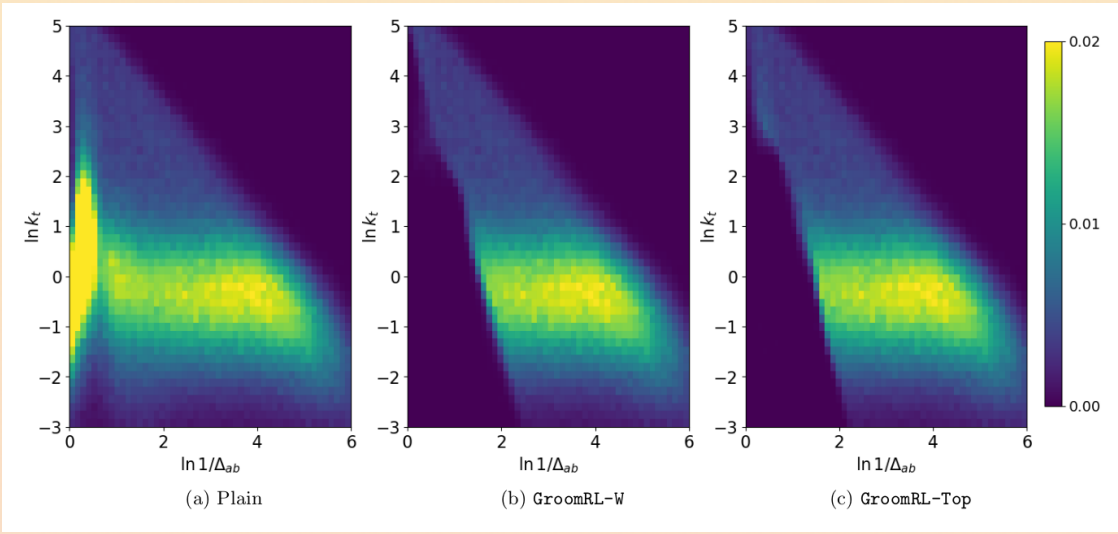


Figure 5: Neural Empirical Bayes for detector correction in collider physics. (a) The source distribution $p(\mathbf{x})$ is shown in blue against the estimated source distribution $q_\theta(\mathbf{x})$ in black. (b) Posterior distribution obtained with rejection sampling, with generating source sample $\mathbf{x}$ indicated in red. (c) Calibration curves for each jet property obtained with rejection sampling on 10000 observations. In (a) and (b), contours represent the 68-95-99.7% levels.

Images from 2011.05836

Interpretability





Images from 1903.09644

Encode sequences

- One-hot encoding for unordered sequences
  - Works e.g. for text

Feature (Color)
Red
Green
Yellow
Green
Red

One Hot Encoding →

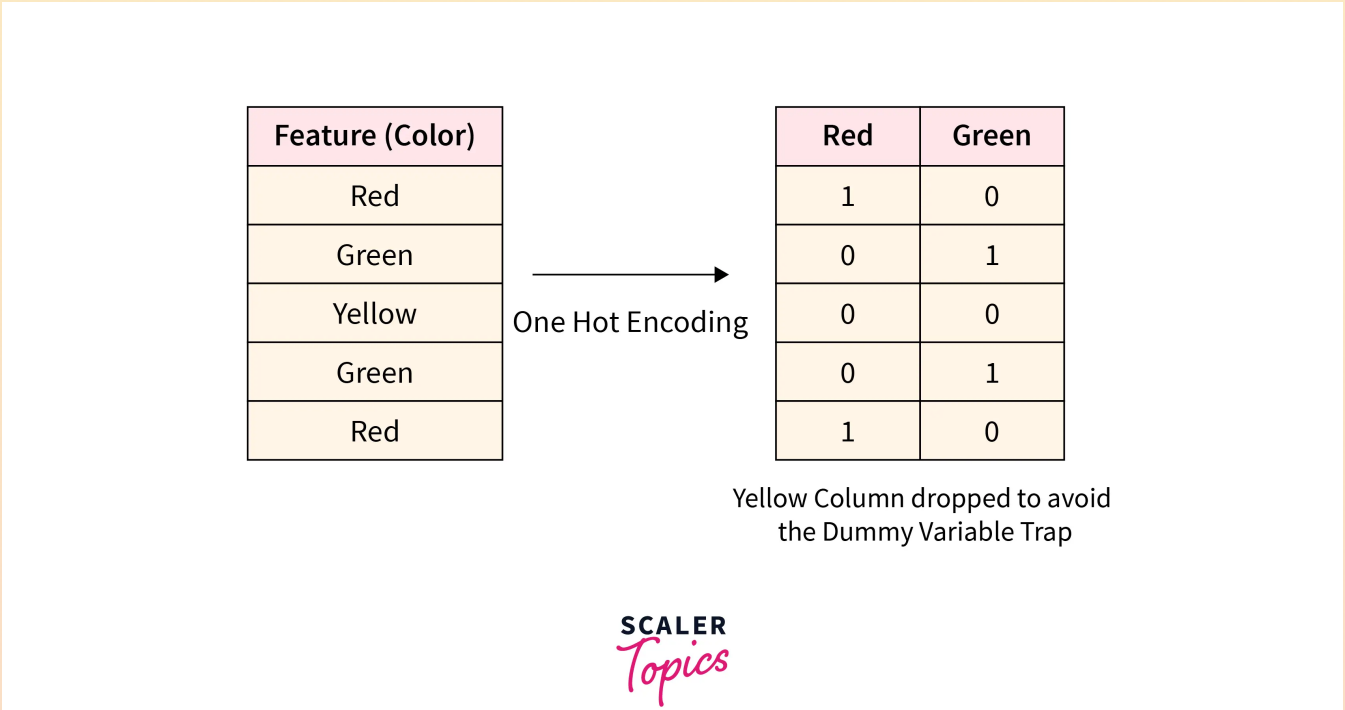
One Hot Encoded Vector	Red	Green	Yellow
[1,00]	1	0	0
[0,1,0]	0	1	0
[0,0,1]	0	0	1
[0,1,0]	0	1	0
[1,00]	1	0	0

SCALER Topics

Picture from scaler.com

Encode sequences

- "Yellow" $[0, 0, 1]$ can be predicted as "0 for red and 0 for green"
  - One-hot-encoded features highly correlated ("multicollinearity")
- Dummy variable trap
  - Drop one of the "dimensions"



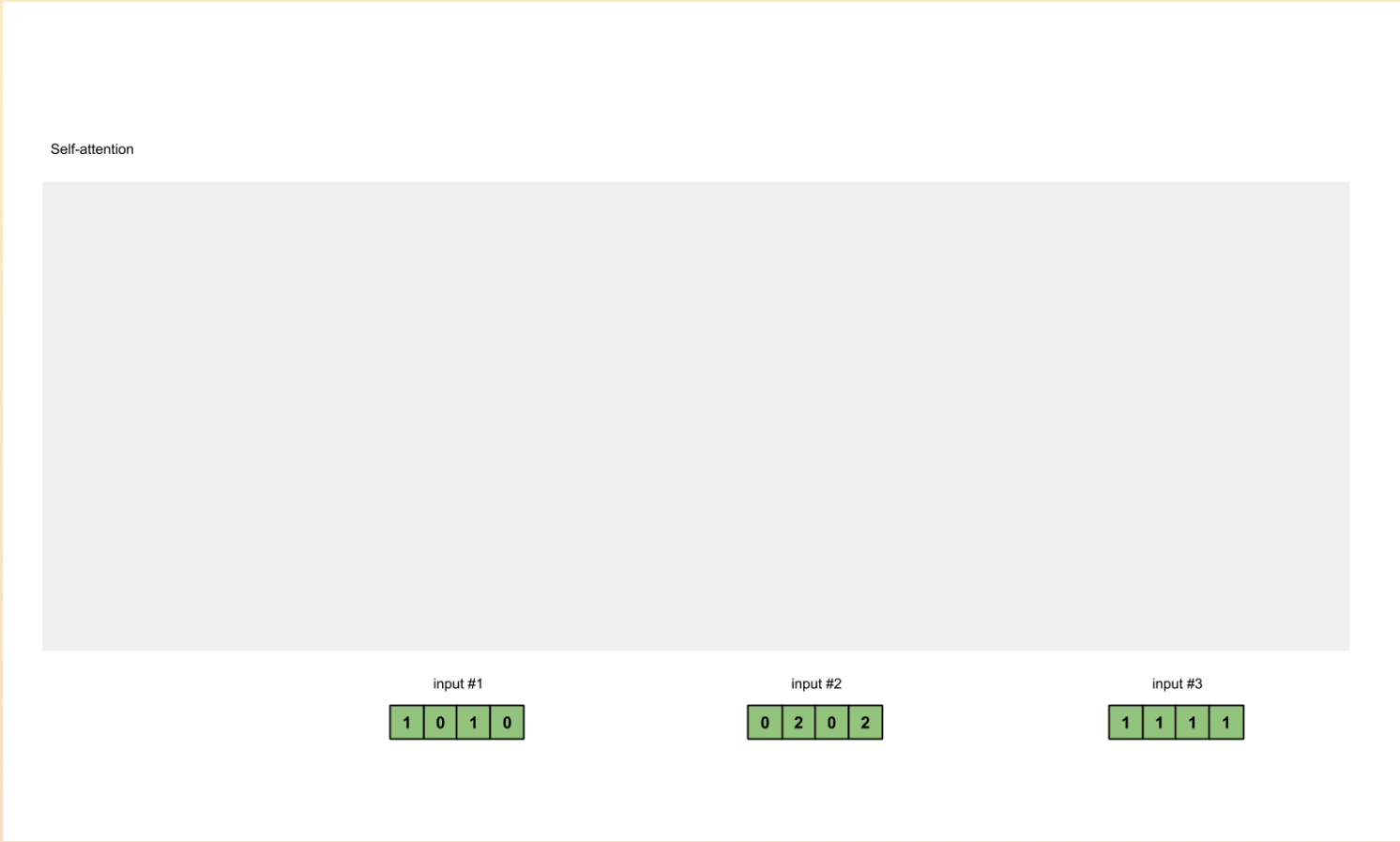
Picture from scaler.com

Self-attention

- Capture dependencies and relationships within inputs
  - Mostly in natural language processing and computer vision
- N inputs, N outputs
  - Allow inputs to interact with eacho other and find out which ones to pay attention to
  - Output is an aggregate of interactions and attention scores
- Useful for:
  - Long-range dependencies: understand complex patterns and dependencies
  - Contextual understanding: assign appropriate weights to important elements in the sequence
  - Parallel computation: can be computed in parallel $\rightarrow$ efficient and scalable for large datasets.

Self-attention

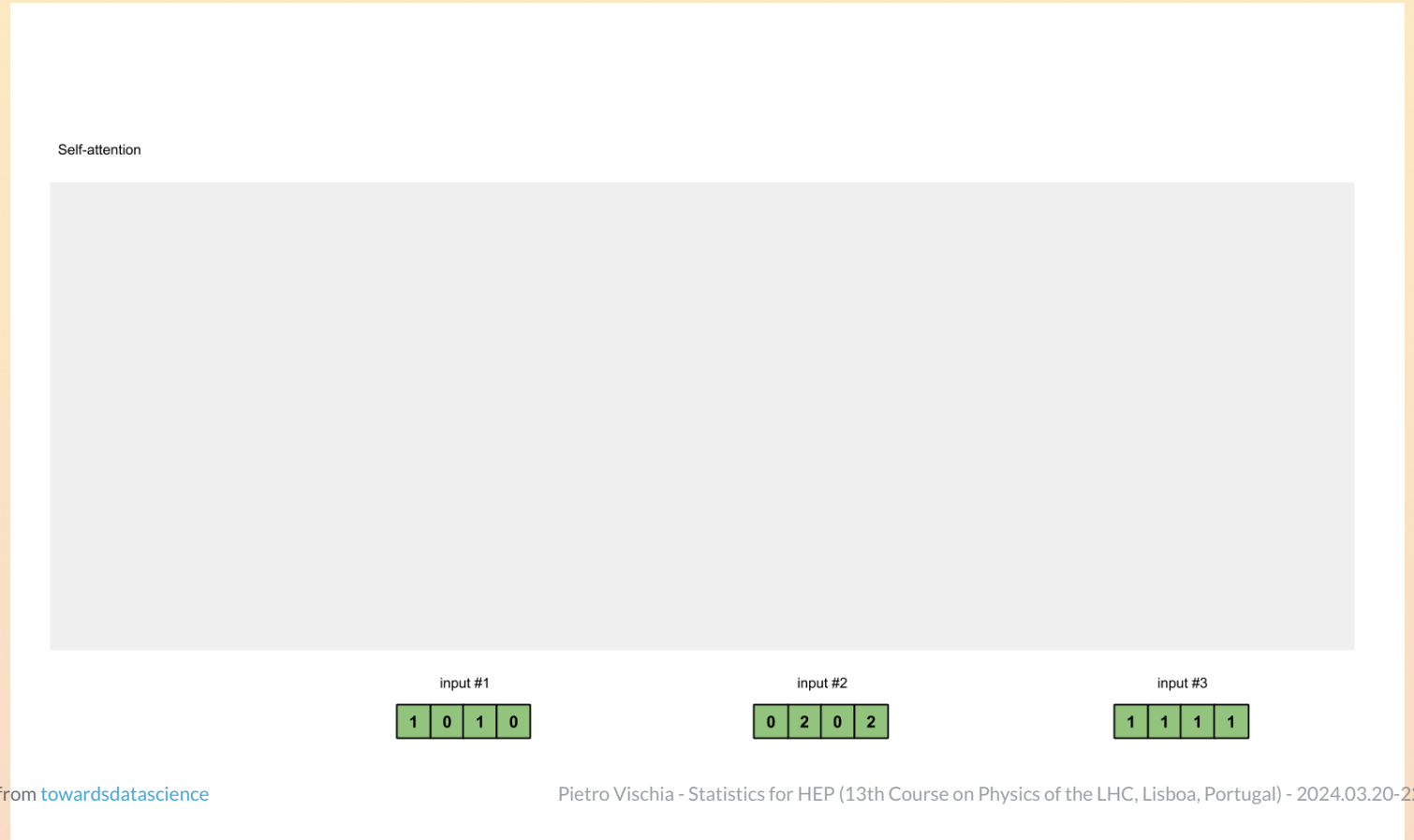
- Inputs (green) must be represented as: key (orange), query (red), value (purple)
  - Initially, by random reweighting of inputs themselves



Animation from towardsdatascience

Self-attention

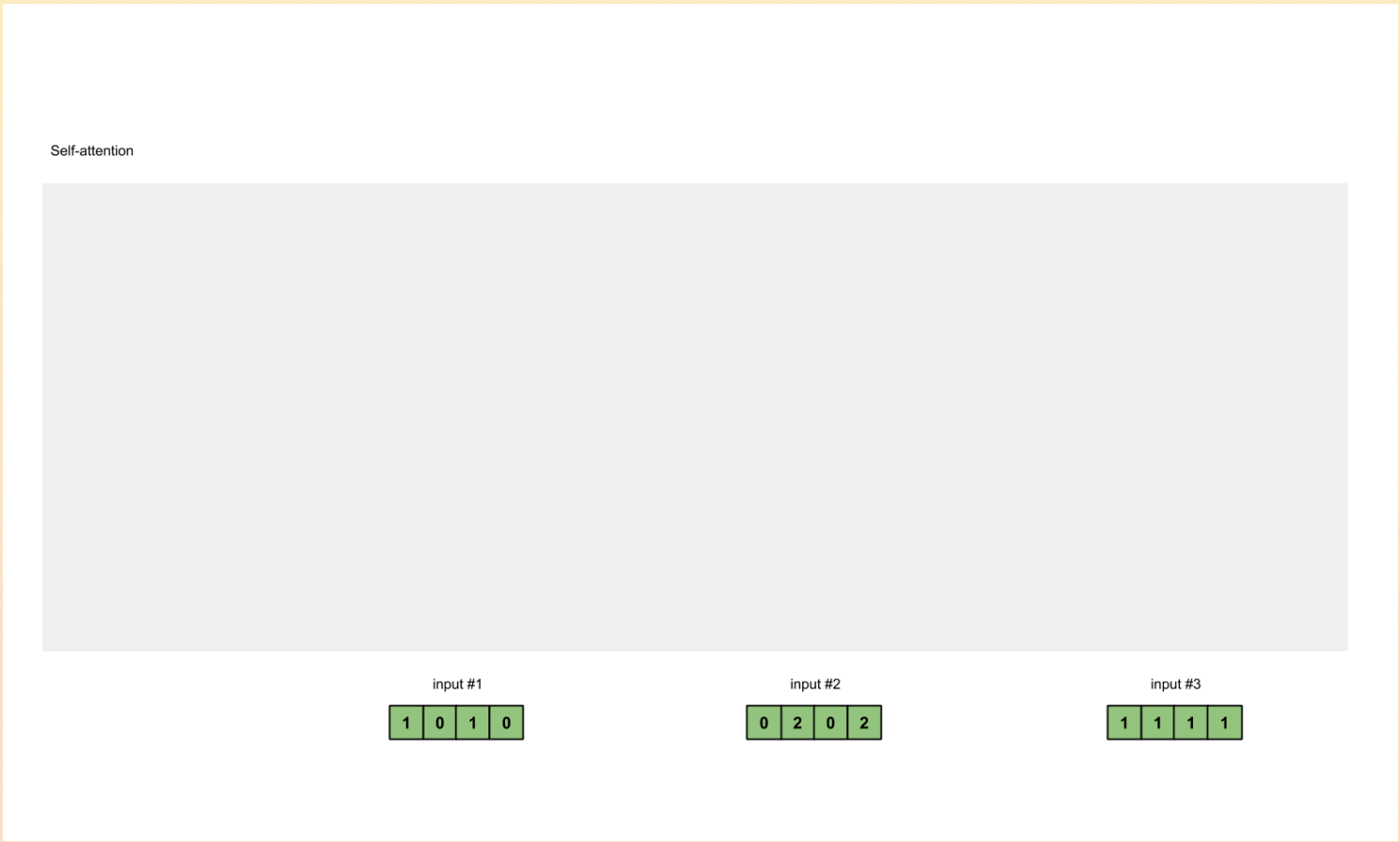
- Calculate attention score
  - Multiply (dot product) each query with all keys
  - For each query: N keys $\rightarrow$ N attention scores



Animation from towardsdatascience

Self-attention

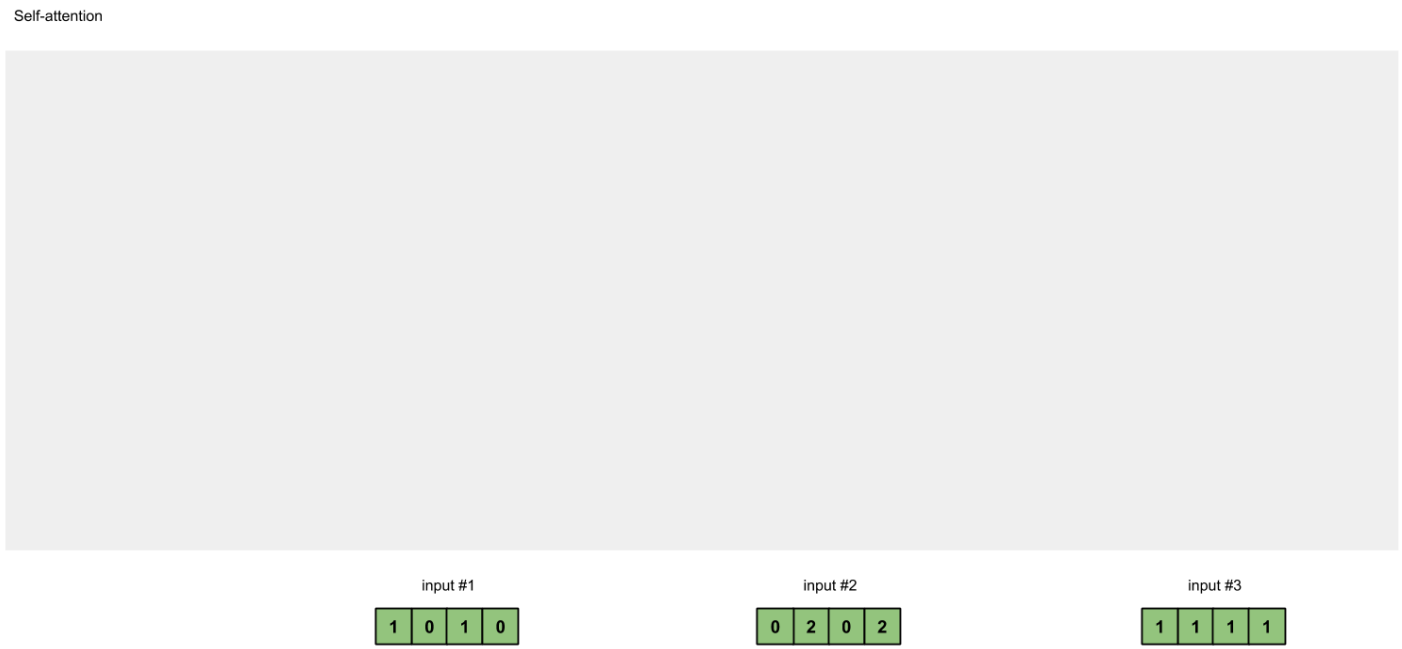
- Activation function (softmax) of attention scores



Animation from towardsdatascience

Self-attention

- Calculate alignment vectors (yellow), i.e. weighted values
  - Multiply each attention score (blue) by its value (purple)
- Sum alignment vectors to get input for output 1, repeat for 2 and 3



Animation from towardsdatascience

Transformers

- The engine behind GPT3

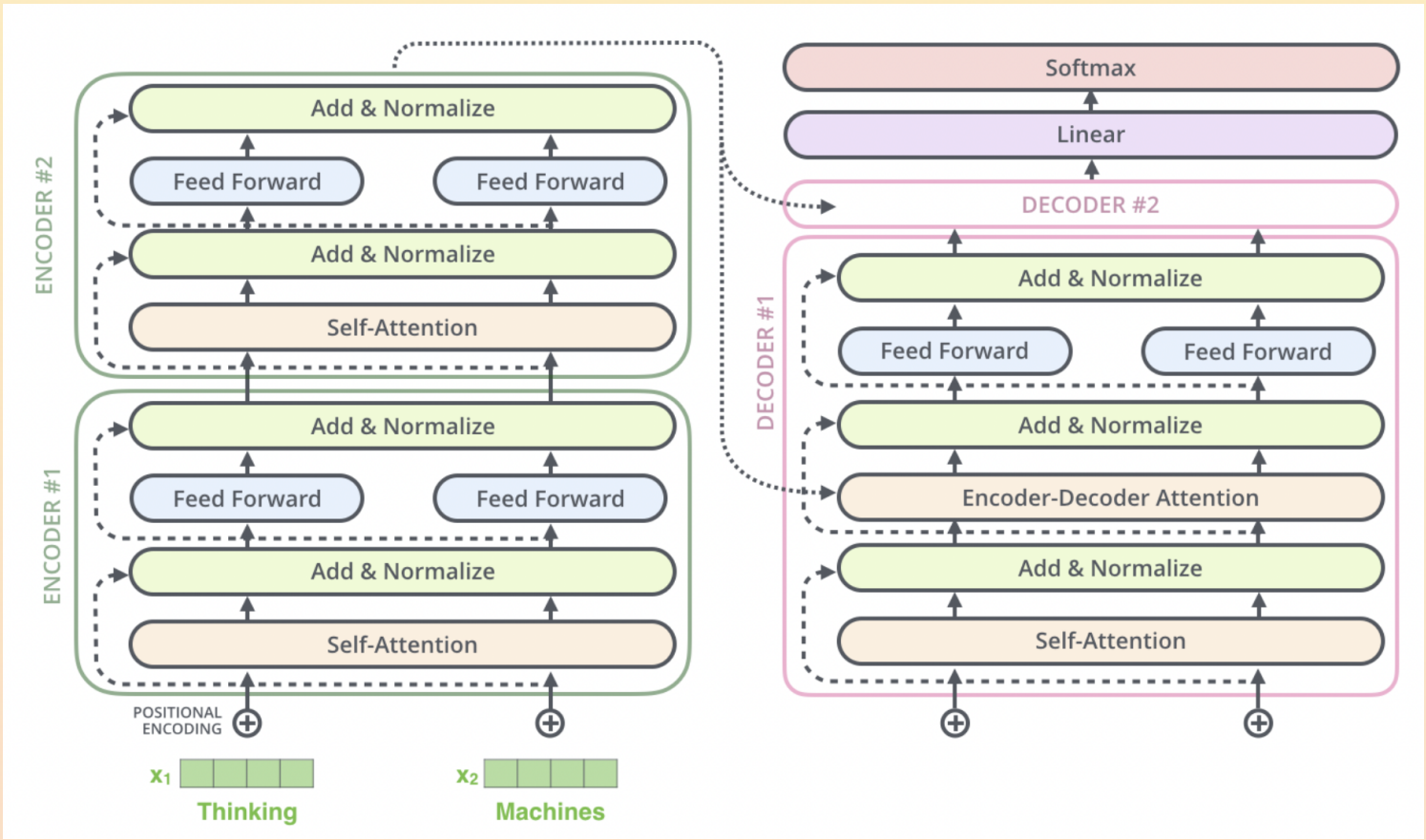


Image by Jay Alammar

Foundation Models

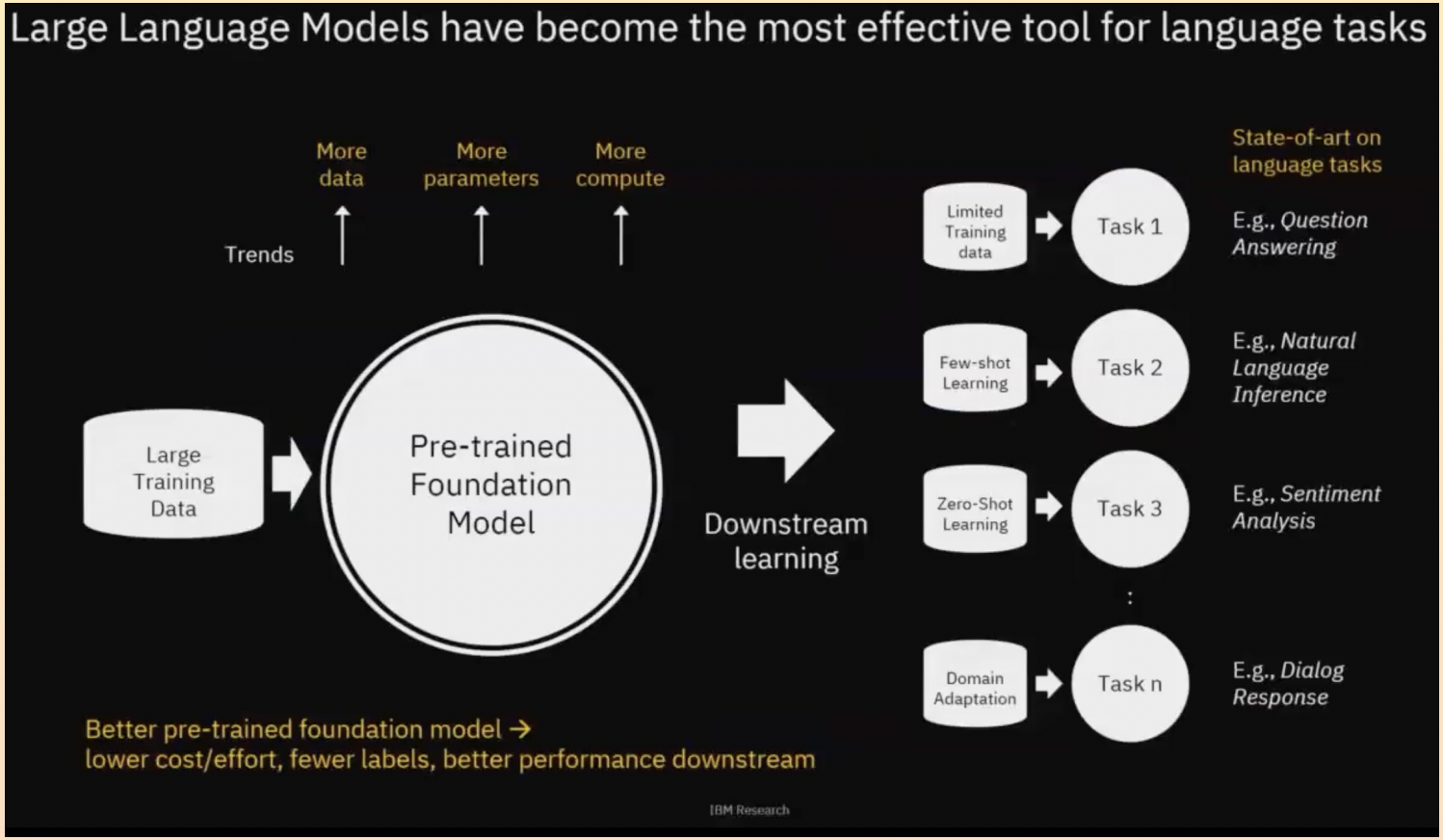
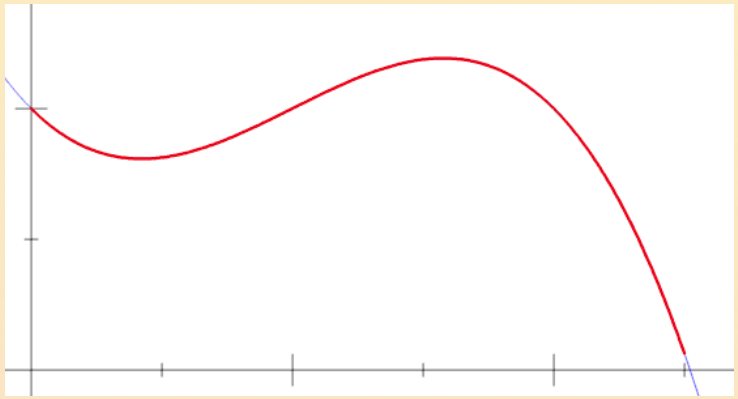


Image by John R. Smith (IBM Research)

Translate Problems into Solutions

- Symbolic integration: find the analytic formula for the area of the curve



	Integration (BWD)	ODE (order 1)	ODE (order 2)
Mathematica (30s)	84.0	77.2	61.6
Matlab	65.2	-	-
Maple	67.4	-	-
Beam size 1	98.4	81.2	40.8
Beam size 10	99.6	94.0	73.2
Beam size 50	99.6	97.0	81.0

Table by Lample, Charton (2019), image by Julia Inozemtseva

Reinforcement learning

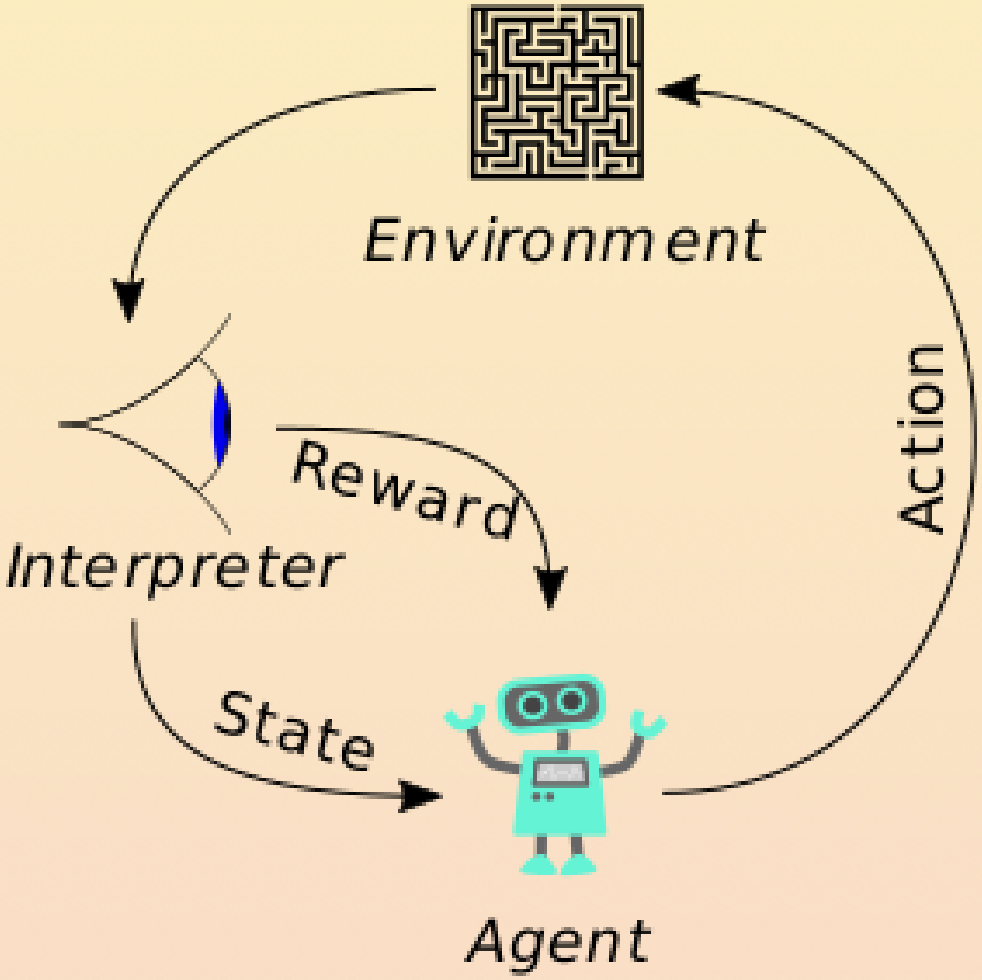


Image by Megajuice on Wikimedia

From Videogames...

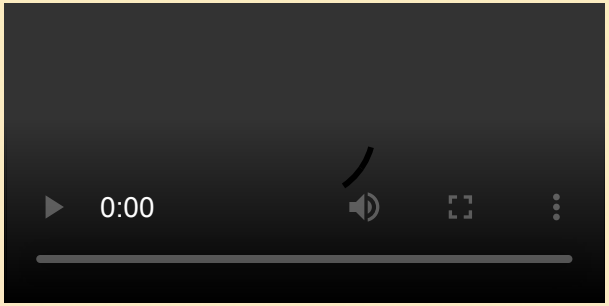


Image from YouTube video

...to Physics

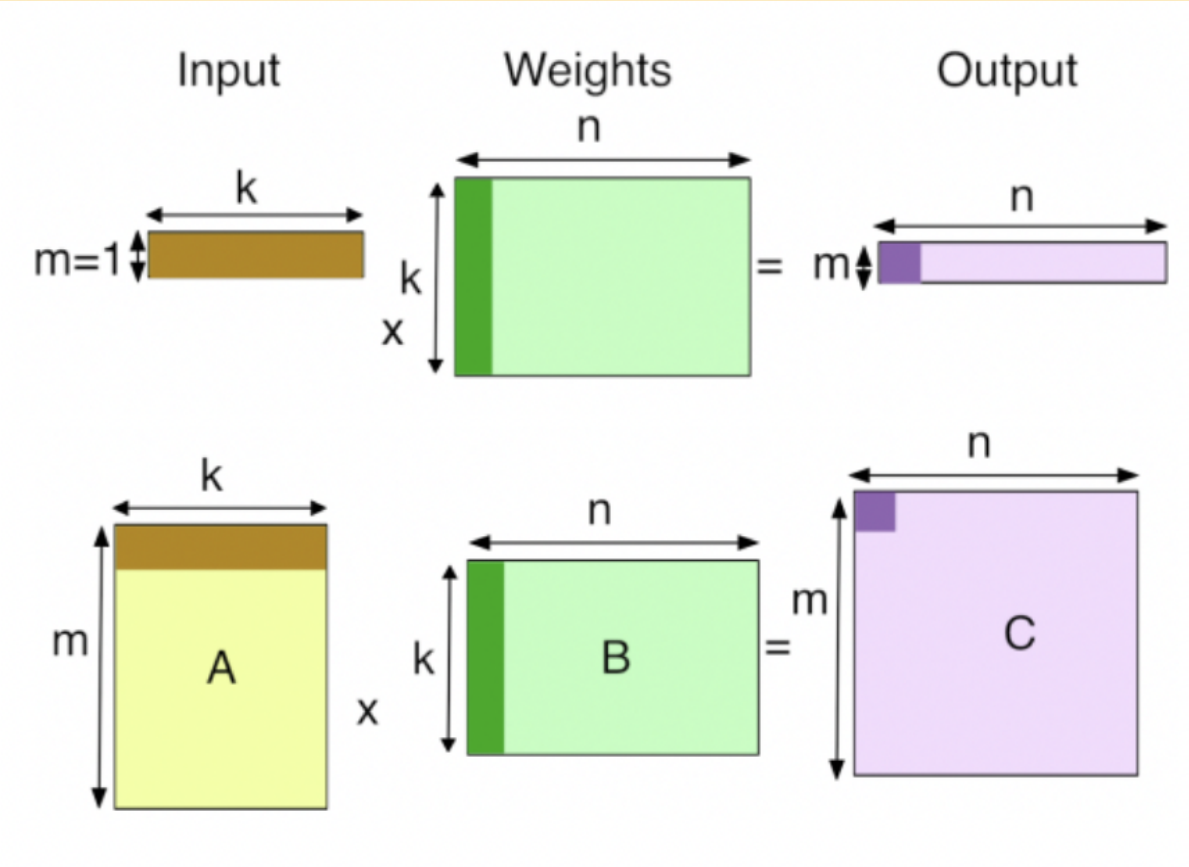
- Reward models consistent with the observed quark properties

charges	$\mathcal{Q} = \begin{array}{c\|ccc\|ccc\|ccc\|cc} & Q_1 & Q_2 & Q_3 & u_1 & u_2 & u_3 & d_1 & d_2 & d_3 & H & \phi \\ \hline q & 6 & 4 & 3 & -2 & 2 & 4 & -3 & -1 & -1 & 1 & 1 \end{array}$
$\mathcal{O}(1)$ coeff.	$(a_{ij}) \simeq \begin{pmatrix} -1.975 & 1.284 & -1.219 \\ 1.875 & -1.802 & -0.639 \\ 0.592 & 1.772 & 0.982 \end{pmatrix}$ $(b_{ij}) \simeq \begin{pmatrix} -1.349 & 1.042 & 1.200 \\ 1.632 & 0.830 & -1.758 \\ -1.259 & -1.085 & 1.949 \end{pmatrix}$
VEV, Value	$v_1 \simeq 0.224\,, \qquad \mathcal{V}(\mathcal{Q}) \simeq -0.598$
charges	$\mathcal{Q} = \begin{array}{ccc\|ccc\|ccc\|cc} Q_1 & Q_2 & Q_3 & u_1 & u_2 & u_3 & d_1 & d_2 & d_3 & H & \phi \\ \hline 1 & 2 & 0 & -1 & -3 & 1 & -3 & -5 & -4 & 1 & 1 \end{array}$
$\mathcal{O}(1)$ coeff.	$(a_{ij}) \simeq \begin{pmatrix} -0.601 & 1.996 & 0.537 \\ -0.976 & -1.498 & -1.156 \\ 1.513 & 1.565 & 0.982 \end{pmatrix}$ $(b_{ij}) \simeq \begin{pmatrix} 0.740 & -1.581 & -1.664 \\ -1.199 & -1.383 & 0.542 \\ 0.968 & 0.679 & -1.153 \end{pmatrix}$
VEV, value	$v_1 \simeq 0.158\,, \qquad \mathcal{V}(\mathcal{Q}) \simeq -0.621$

Table from 2103.04759

From perceptron-based networks...

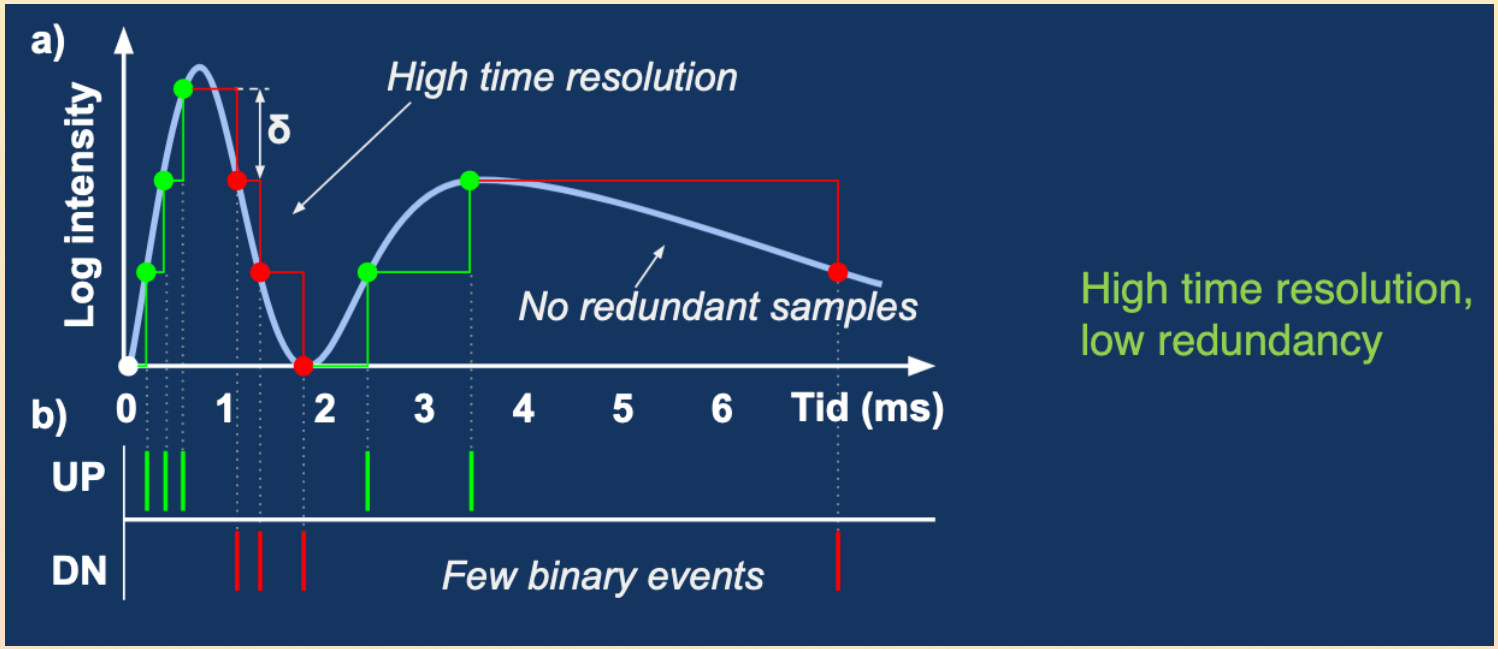
- Matrix multiplication



Figures by Gilles Louppe

...to spiking neural networks

- Event-driven computations
  - "when a spike occurs, compute something"



Figures by Fredrik Sandin

The energy advantage

- Perceptron-based networks: matrix multiplication
  - Sparsity doesn't affect much the throughput and energy consumption
- Spiking neural networks: event-driven computations
  - Sparser inputs require less computations, therefore less time and energy

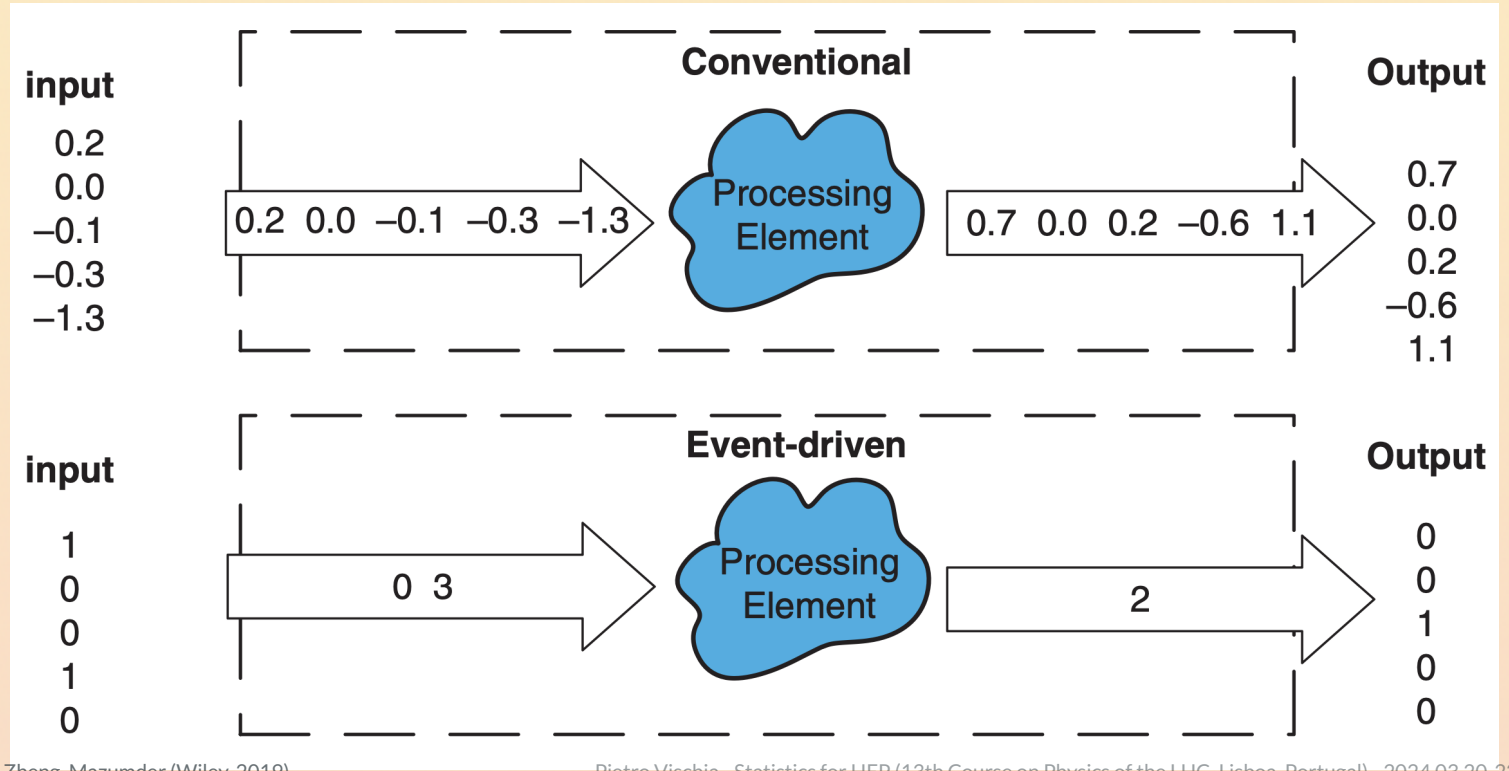


Image from Zheng, Mazumder (Wiley, 2019)

Encode information with Qubits

- Random bit (Bernoulli random variable) whose description is not governed by classical probability theory but by quantum mechanics
- Not only "because it can take real values in $[0, 1]$": complex numbers as coefficients α and β create interference
  - Interference is not reproducible with classical bits

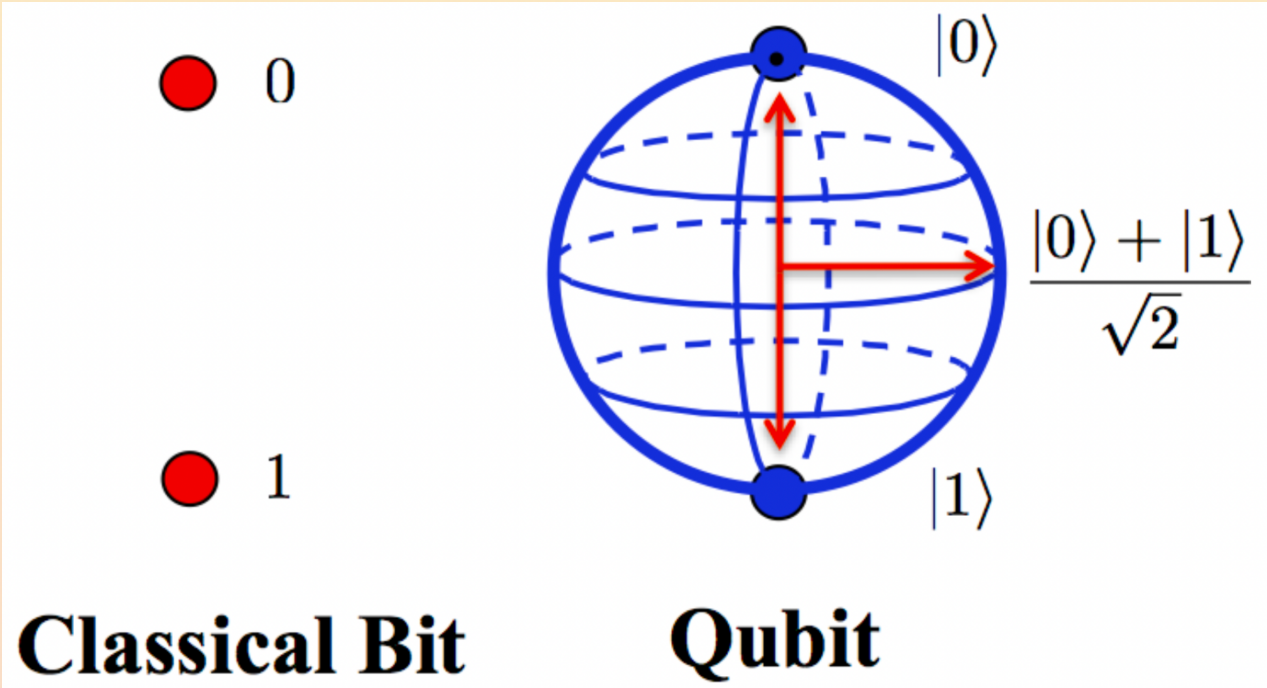
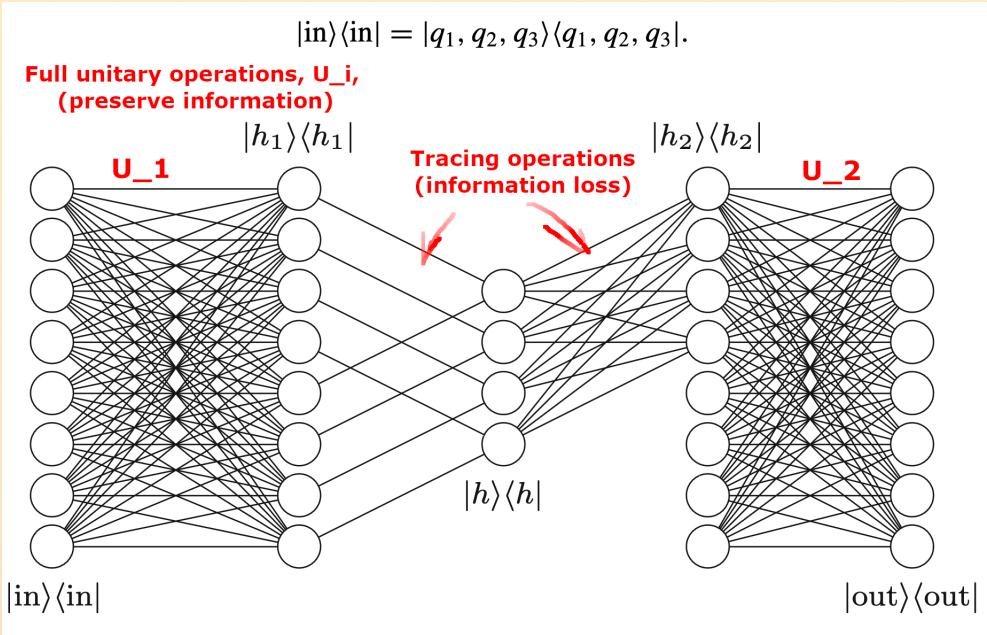


Image by prateekvjoshi

Represent neural networks

- Qubit operations can represent rather naturally neural networks



- Gradient descent exploits intrinsic analytic differentiability of quantum circuits

$$
\begin{aligned}
\partial_\mu \langle\psi(x,\theta)|\sigma_z|\psi(x,\theta)\rangle &= \langle 0|\ldots\partial_\mu e^{-i\mu\sigma}\ldots\sigma_z\ldots e^{i\mu\sigma}\ldots|0\rangle \\
&\quad + \langle 0|\ldots e^{-i\mu\sigma}\ldots\sigma_z\ldots\partial_\mu e^{i\mu\sigma}\ldots|0\rangle \\
&= \langle 0|\ldots(-i\sigma)e^{-i\mu\sigma}\ldots\sigma_z\ldots e^{i\mu\sigma}\ldots|0\rangle \\
&\quad + \langle 0|\ldots e^{-i\mu\sigma}\ldots\sigma_z\ldots(i\sigma)e^{i\mu\sigma}\ldots|0\rangle \\
&= \langle 0|\ldots(1-i\sigma)e^{-i\mu\sigma}\ldots\sigma_z\ldots(1+i\sigma)e^{i\mu\sigma}\ldots|0\rangle \\
&\quad + \langle 0|\ldots(1+i\sigma)e^{-i\mu\sigma}\ldots\sigma_z\ldots(1-i\sigma)e^{i\mu\sigma}\ldots|0\rangle
\end{aligned}
$$

Images from M. Schuld, F. Petruccione (2018)

Need for new paradigma

- If you are interested in Neuromorphic computing or Quantum computing, drop me a line!

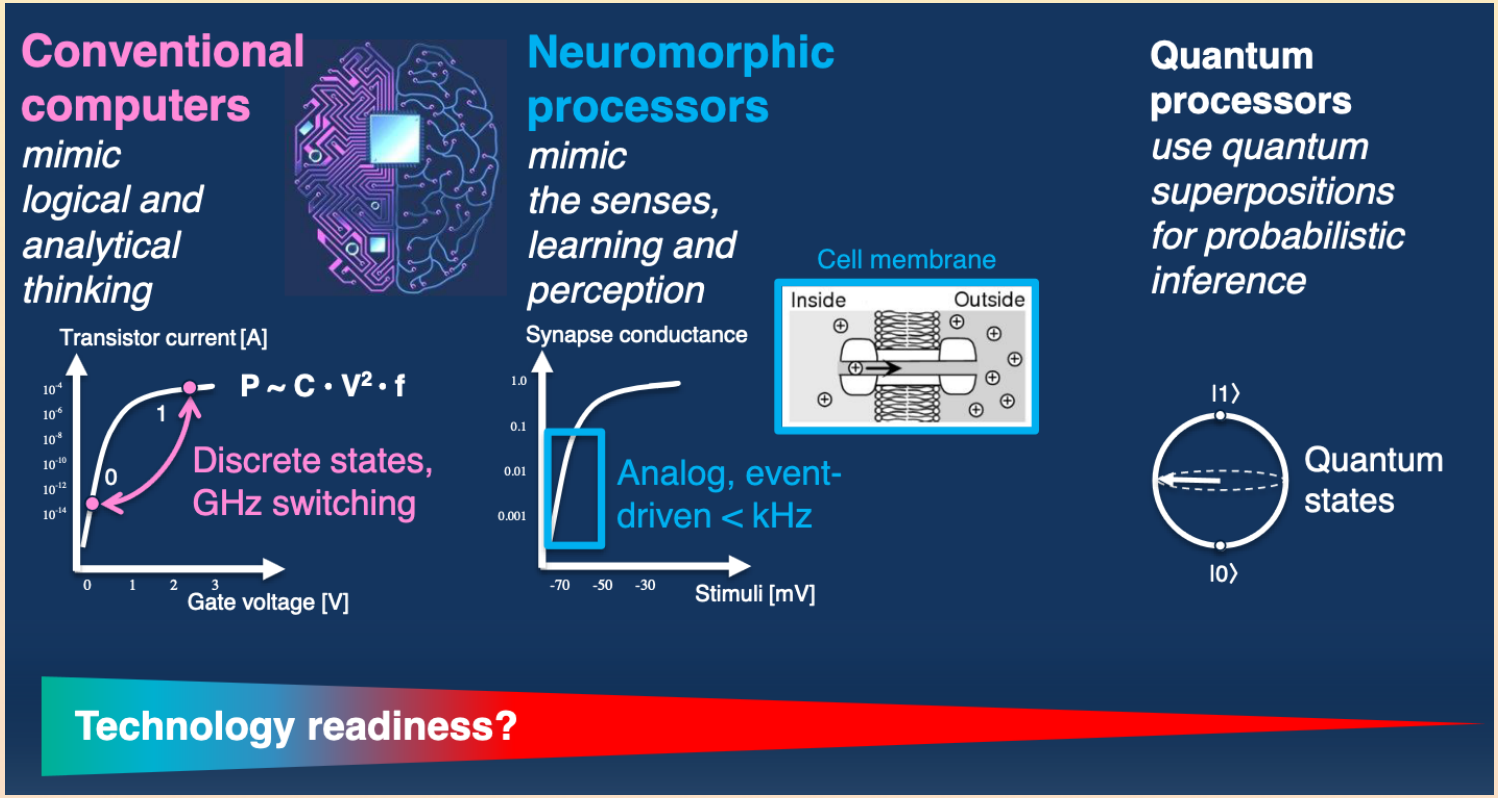
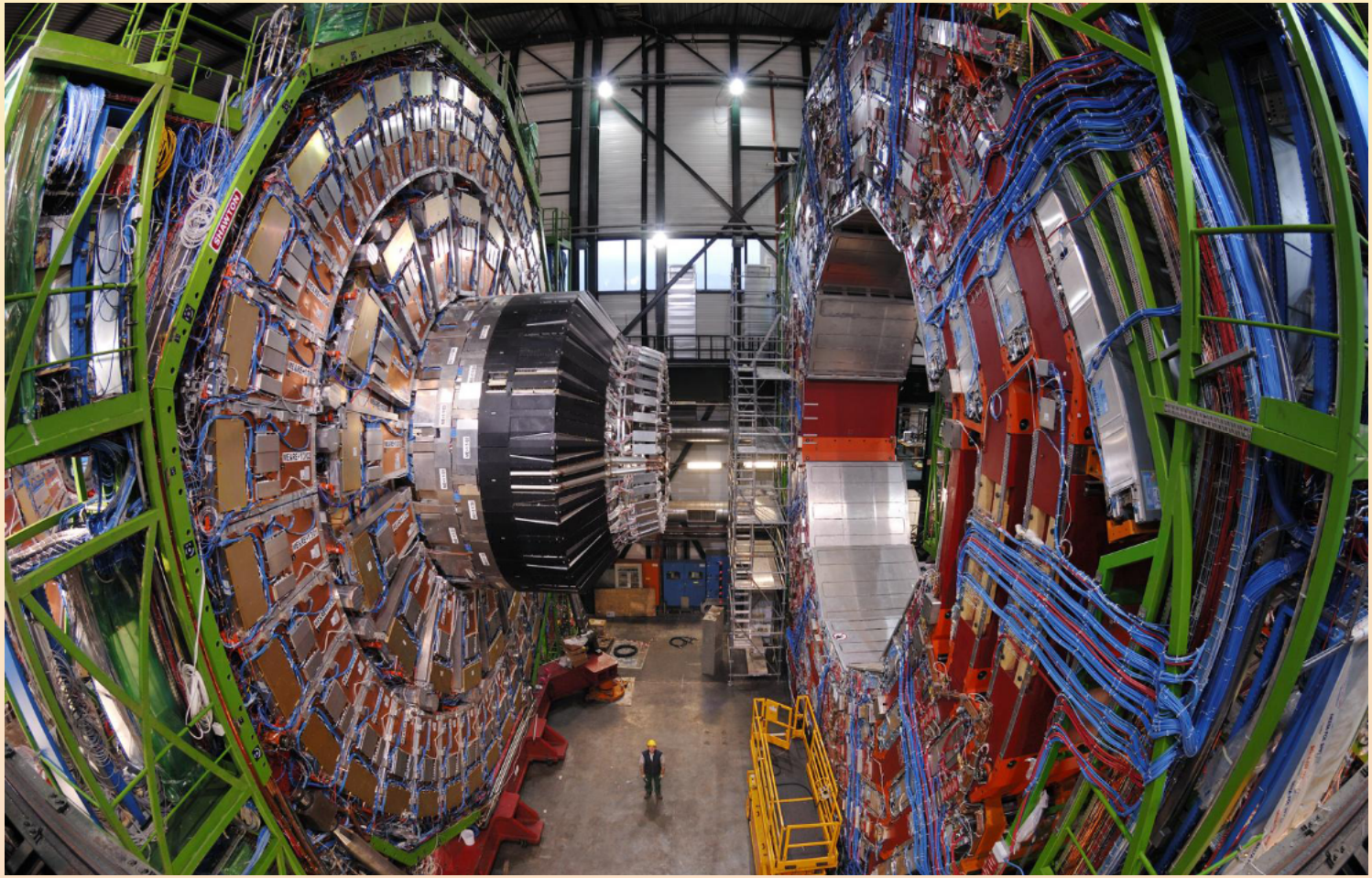


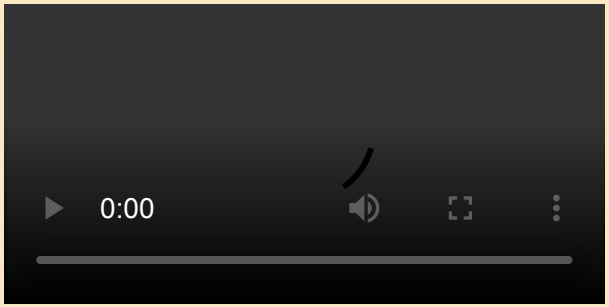
Image by Fredrik Sandin, Lulea University

The Detectors of the Present



The Detectors of the Future

- Optimize/learn by finding the minimum of a function $\mathcal{L} : \mathbb{R}^n \to \mathbb{R}$
- Output represents performance on a physics goal or a constraint (e.g. cost)



Animation by Alec Radford

The Detectors of the Future

- Joint optimization yields in general different solution than optimization of individual features

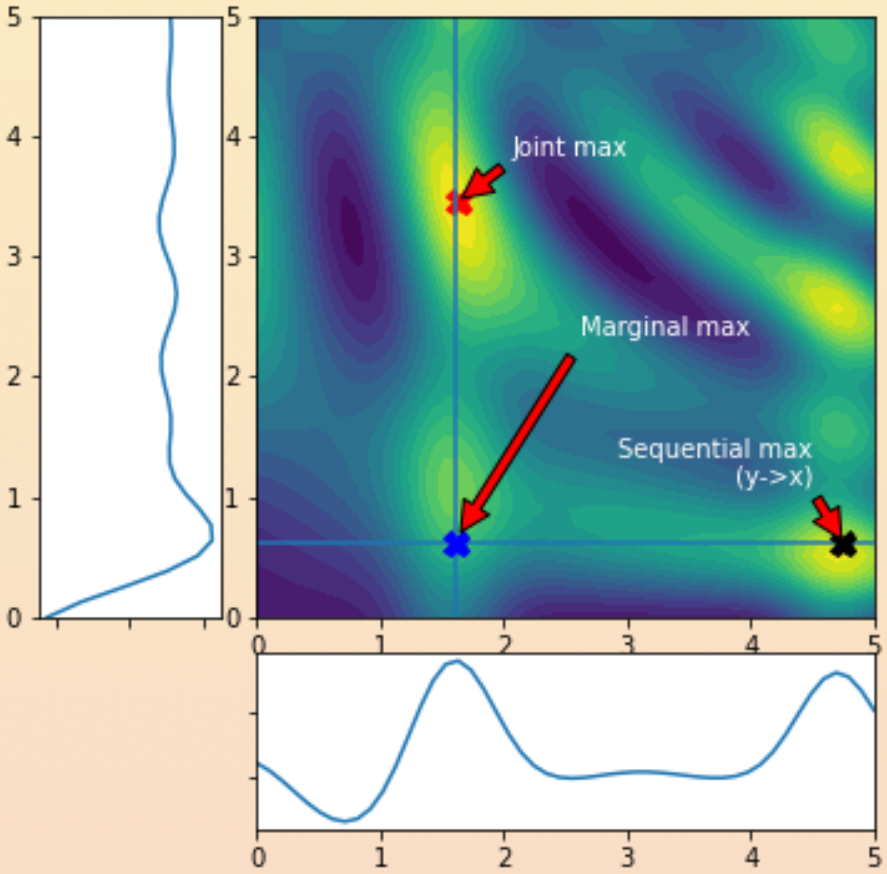


Illustration (c) P. Vischia, book in preparation (Springer Nature)

The Detectors of the Future

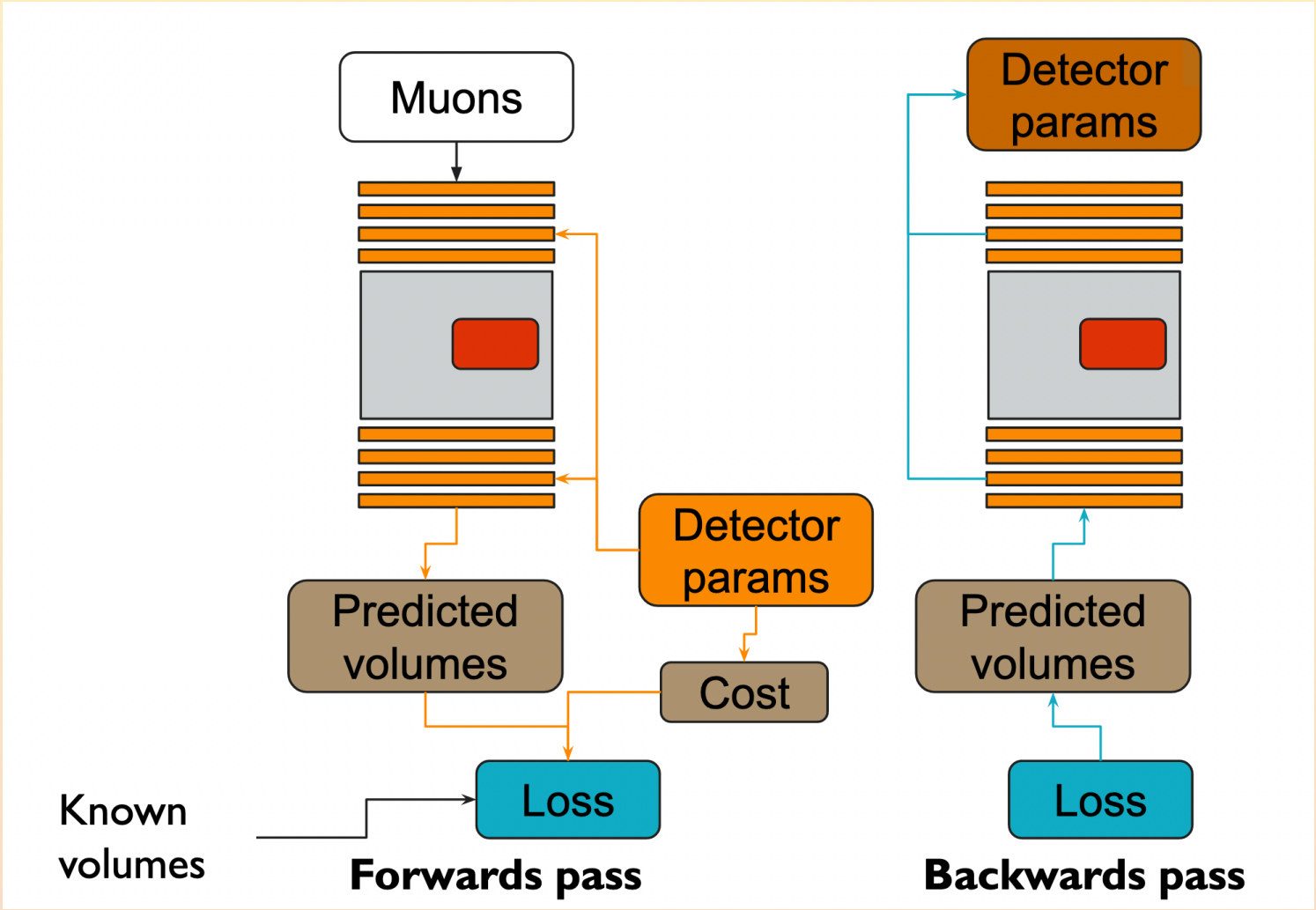


Illustration from the TomOpt paper draft

MODE

- Creation (11.2020) and rapid expansion of the MODE Collaboration

  https://mode-collaboration.github.io/

- Joint effort of particle physicists, nuclear physicists, astrophysicists, and computer scientists

At INFN and Universitá of Padova Dr. **Tommaso Dorigo**, Dr. **Pablo De Castro Manzano**, Dr. **Lukas Layer**, Dr. **Giles Strong**, Dr. **Mia Tosi**, and Dr. **Hevjin Yarar**
At Université catholique de Louvain Dr. **Andrea Giammanco**, Prof. **Christophe Delaere**, Mr. **Maxime Lagrange**, and Dr. **Pietro Vischia**
At Université Clermont Auvergne, Prof. **Julien Donini**, and Mr. **Federico Nardi** (joint with Universitá di Padova)
At the Higher School of Economics of Moscow, Prof. **Andrey Ustyuzhanin**, Dr. **Alexey Boldyrev**, Dr. **Denis Derkach**, and Dr. **Fedor Ratnikov**
At the Instituto de Física de Cantabria, Dr. **Pablo Martínez Ruíz del Árbol**
At CERN, Dr. **Jan Kieseler**
At University of Oxford Dr. **Atilim Gunes Baydin**
At New York University Prof. **Kyle Cranmer**
At Université de Liège Prof. **Gilles Louppe**
At GSI Dr. **Anastasios Belias**
At Rutgers University Dr. **Claudius Krause**
At Uppsala Universitet Prof. **Christian Glaser**
At TU-München, Prof. **Lukas Heinrich** and Mr. **Max Lamparth**
At Durham University Dr. **Patrick Stowell**
At Lebanese University Prof. **Haitham Zaraket**

2022
28 people
15 institutions

2019
10 people
4 institutions

Expression of Interest

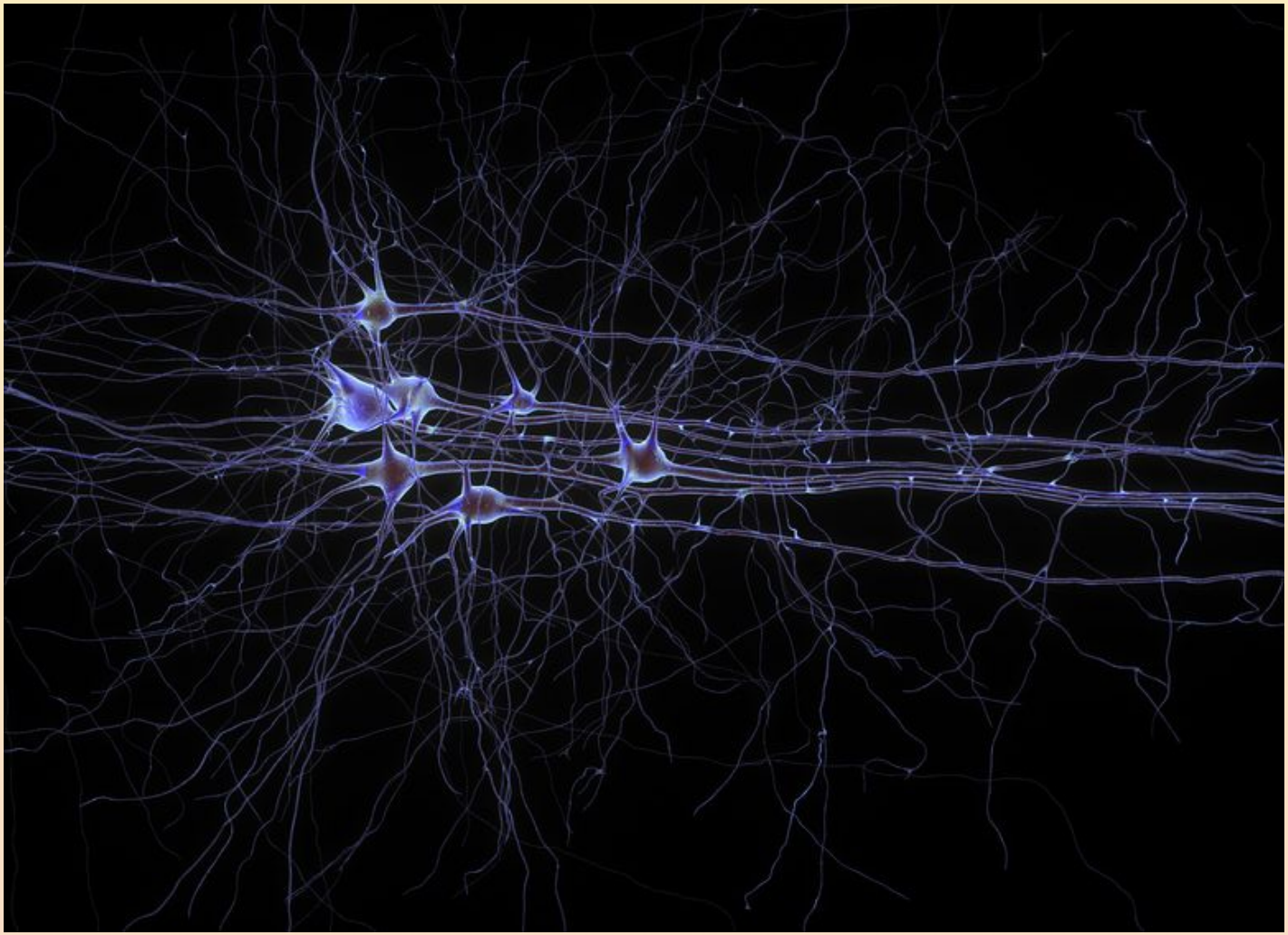
for a synergic research plan of potential interest of the JENAS group

T. Dorigo, D. Boumediene, C. Delaere, D. Derkach, J. Donini,
A. Giammanco, R. Rossin, M. Tosi, A. Ustyuzhanin, P. Vischia,

December 3, 2019

Coordinator
Steering Board

Artificial Brain



—
Image from pinimg.com

Artificial General Intelligence?

Source of the images lost in the sands of time

Not yet.

My efforts in the next years

- Exploration of high-dimensional spaces via gradient descent, eventually powered by quantum algorithms
- Realistic neurons (spiking networks on neuromorphic circuits)
- Applications to experiment design and to heart diseases
- If you are interested in collaborating, drop me an email!