A Short Tutorial on Artificial Intelligence, Deep Learning, and Probabilistic Circuits

Kristian Kersting

Thanks to Pedro Domingos, Christoph Lampert and Constantin Rothkopf for some of the slides

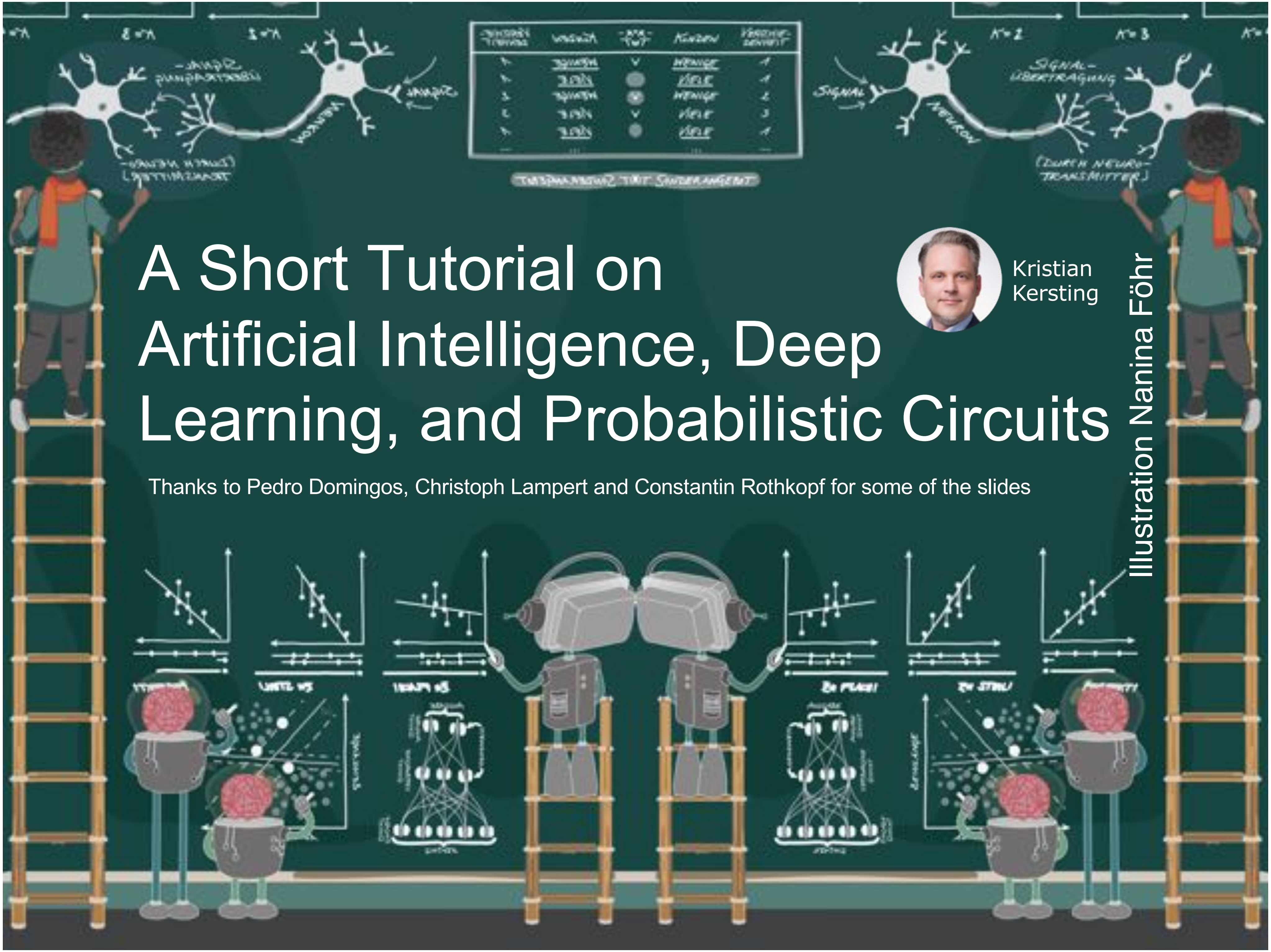
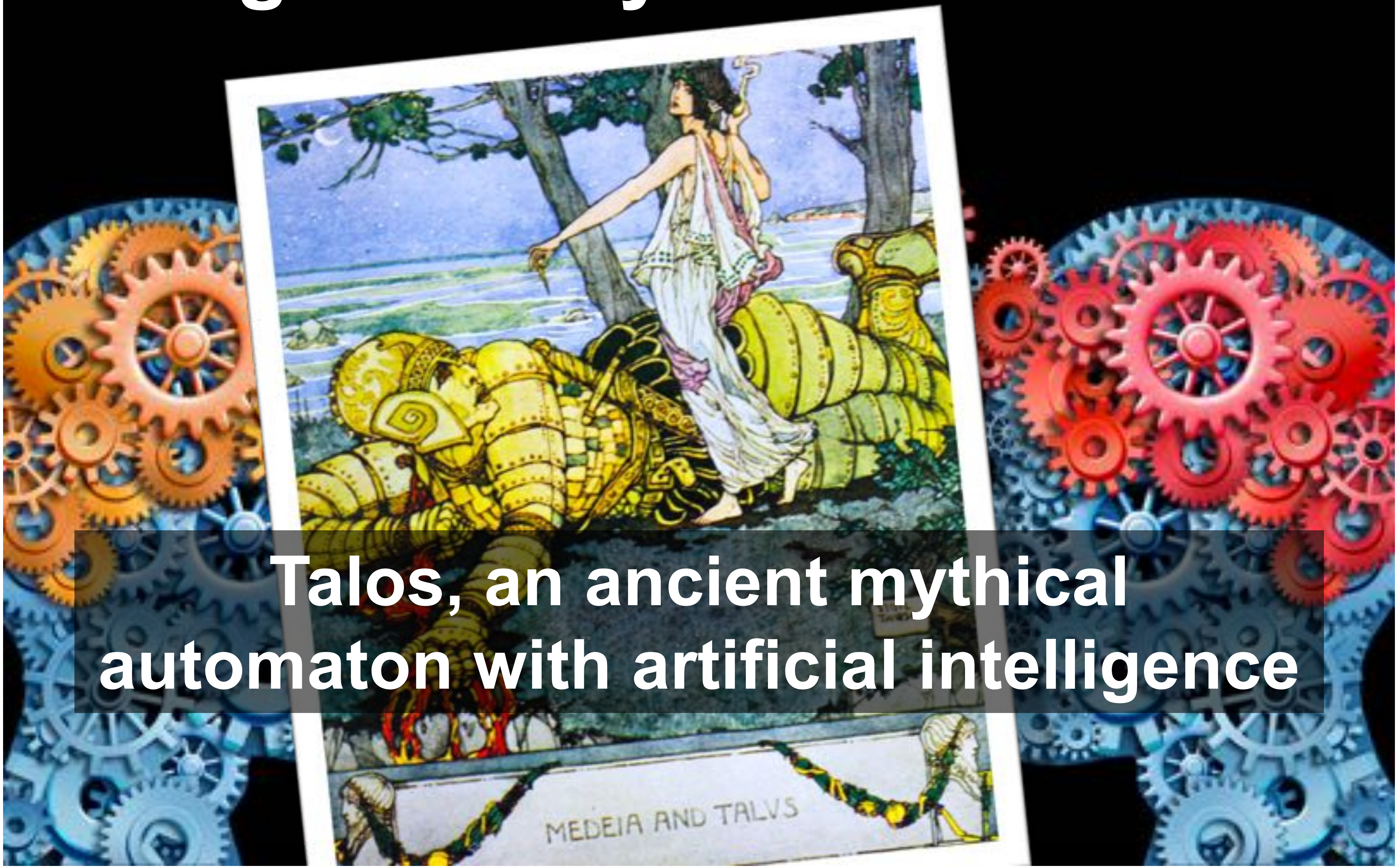


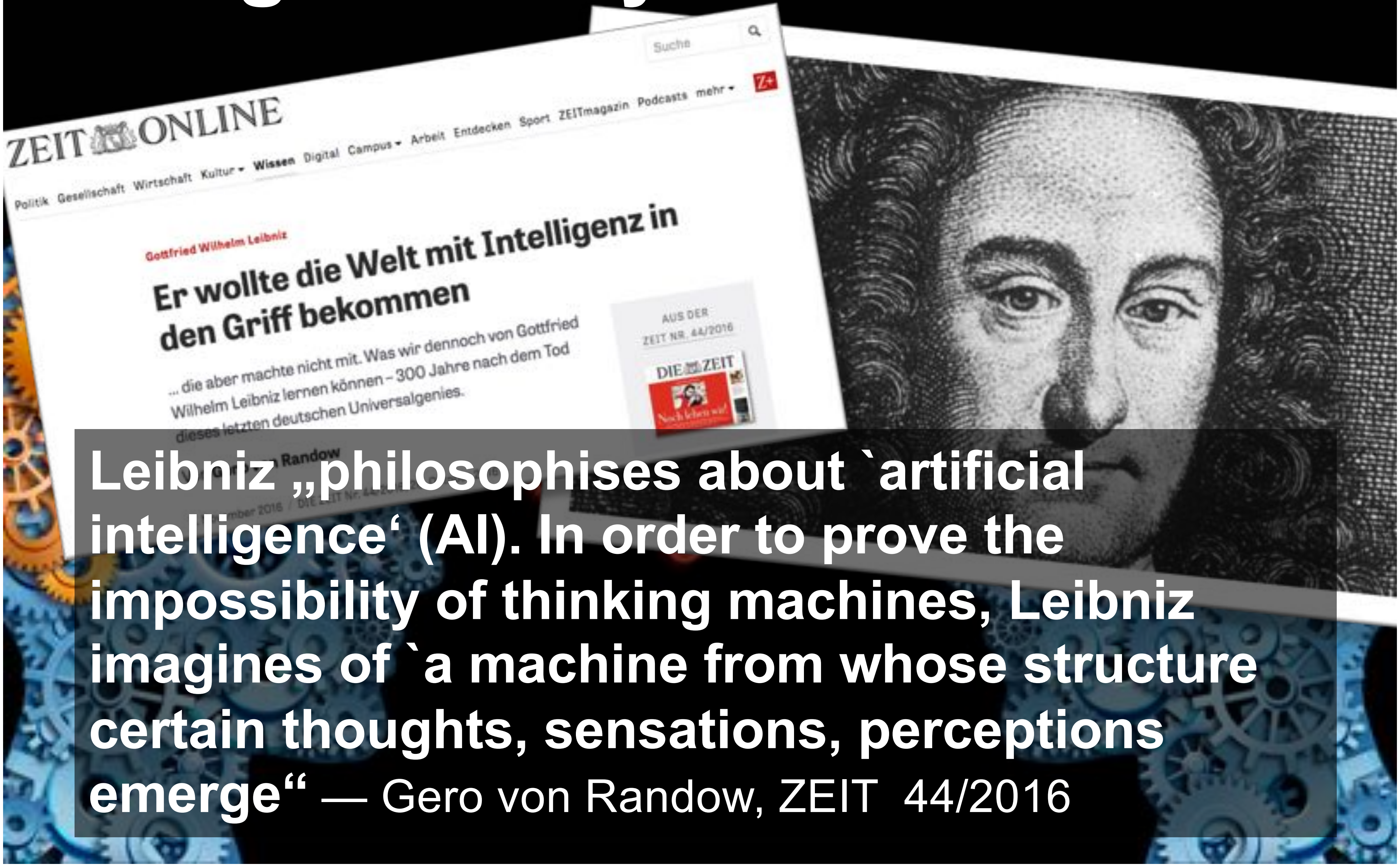
Illustration Nanina Föhr

The dream of an artificially intelligent entity is not new



Talos, an ancient mythical automaton with artificial intelligence

The dream of an artificially intelligent entity is not new



Leibniz „philosophises about `artificial intelligence‘ (AI). In order to prove the impossibility of thinking machines, Leibniz imagines of `a machine from whose structure certain thoughts, sensations, perceptions emerge“ — Gero von Randow, ZEIT 44/2016

AI today


THE ECONOMIC IMPACT OF
ARTIFICIAL INTELLIGENCE
NORTH AMERICA
$3.7 TR
NORTHERN EUROPE
$1.8 TR
CHINA
$7 TR
$0.7 TR
SOUTHERN EUROPE
LATIN AMERICA
$0.5 TR
REST OF WORLD
$1.2 TR
$0.9 TR
DEVELOPED ASIA
Projected Global Economic Effects of AI by 2030
Source: PwC

But, what exactly is AI?

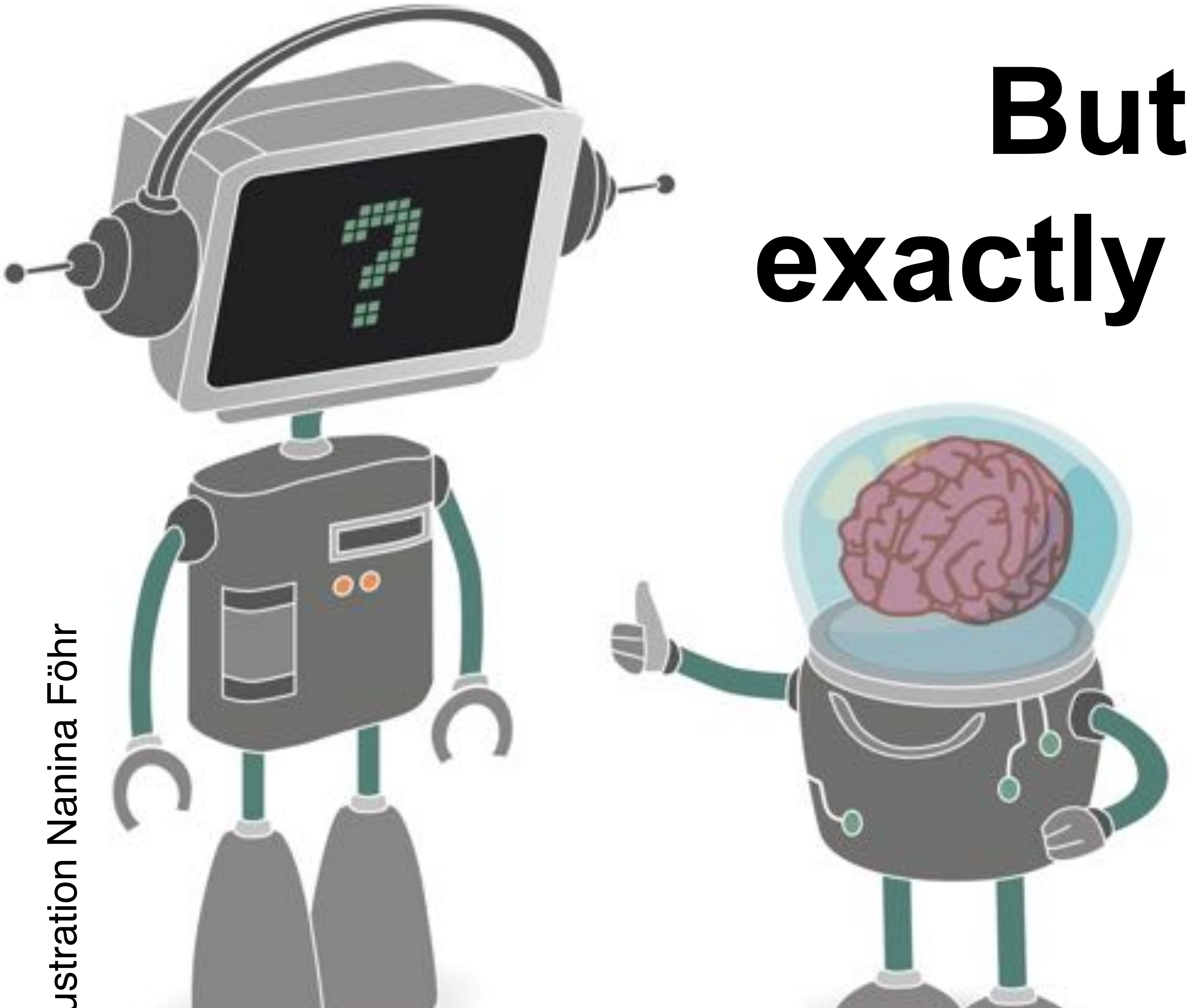


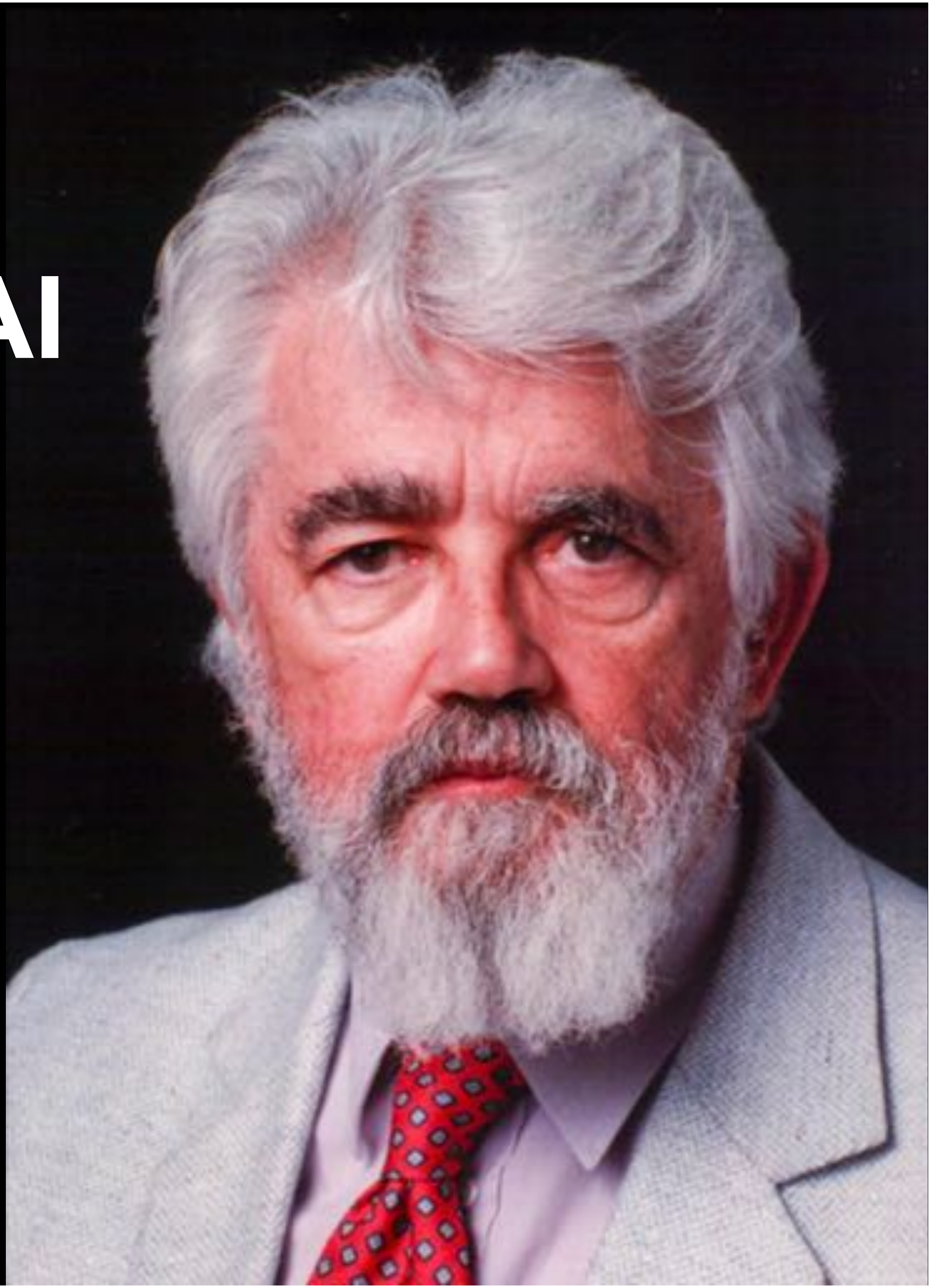
Illustration Nanina Föhr

The Definition of AI

„the science and engineering of making intelligent machines, especially intelligent computer programs.

It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.“

- John McCarthy, Stanford (1956), coined the term AI, Turing Awardee



Learning Thinking Planning

AI = Algorithms for …

Vision Behaviour Reading

Machine Learning

the science "concerned with the question of how to construct computer programs that automatically improve with experience"

- Tom Mitchell (1997) CMU

Deep Learning

a form of machine learning that makes use of artificial neural networks

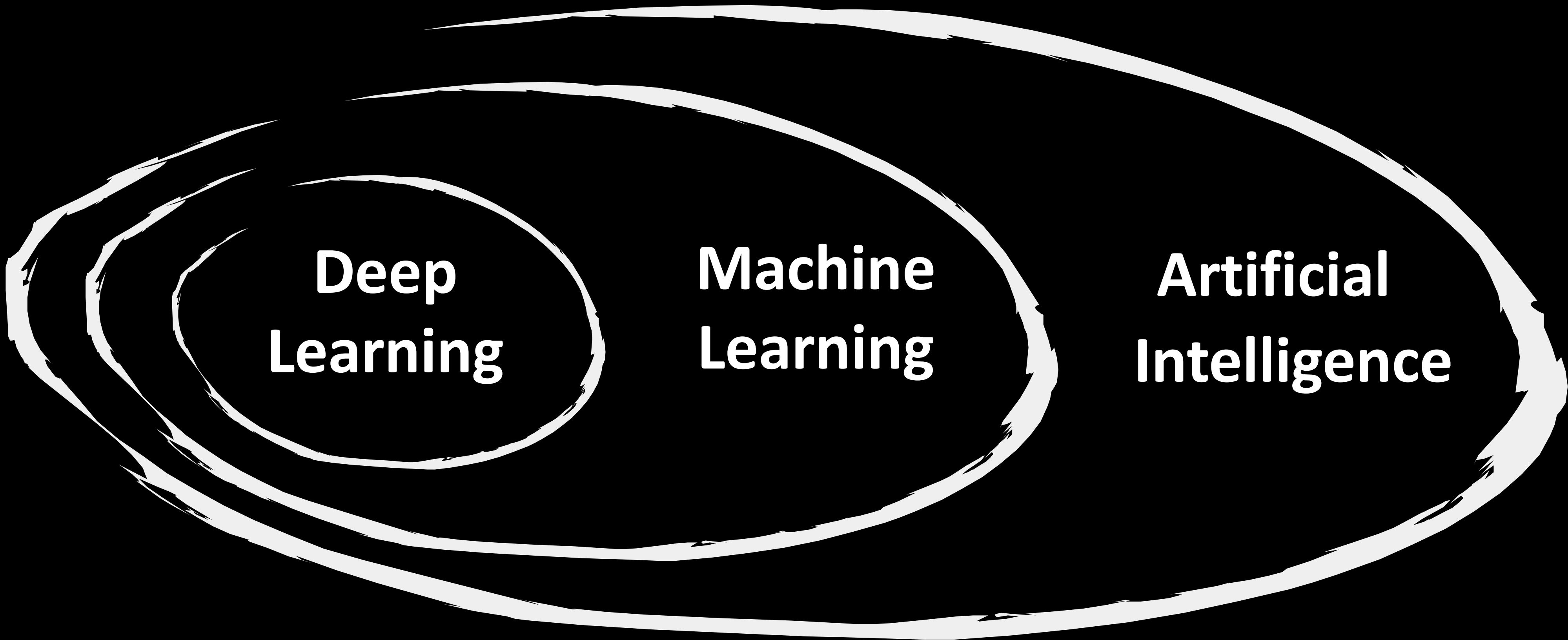
Geoffrey Hinton
Google
Univ. Toronto (CAN)

Yann LeCun
Facebook (USA)

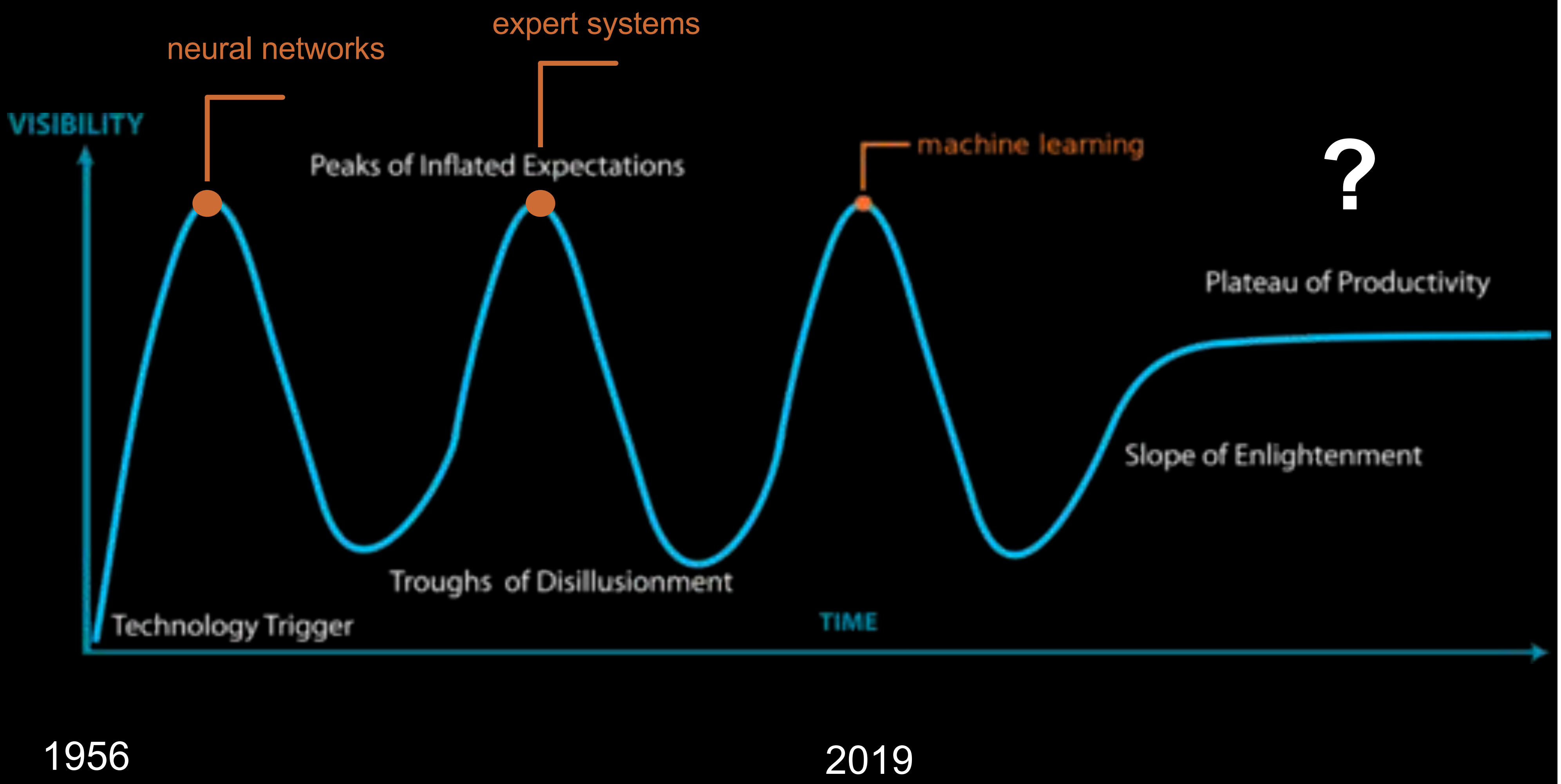
Yoshua Bengio
Univ. Montreal (CAN)

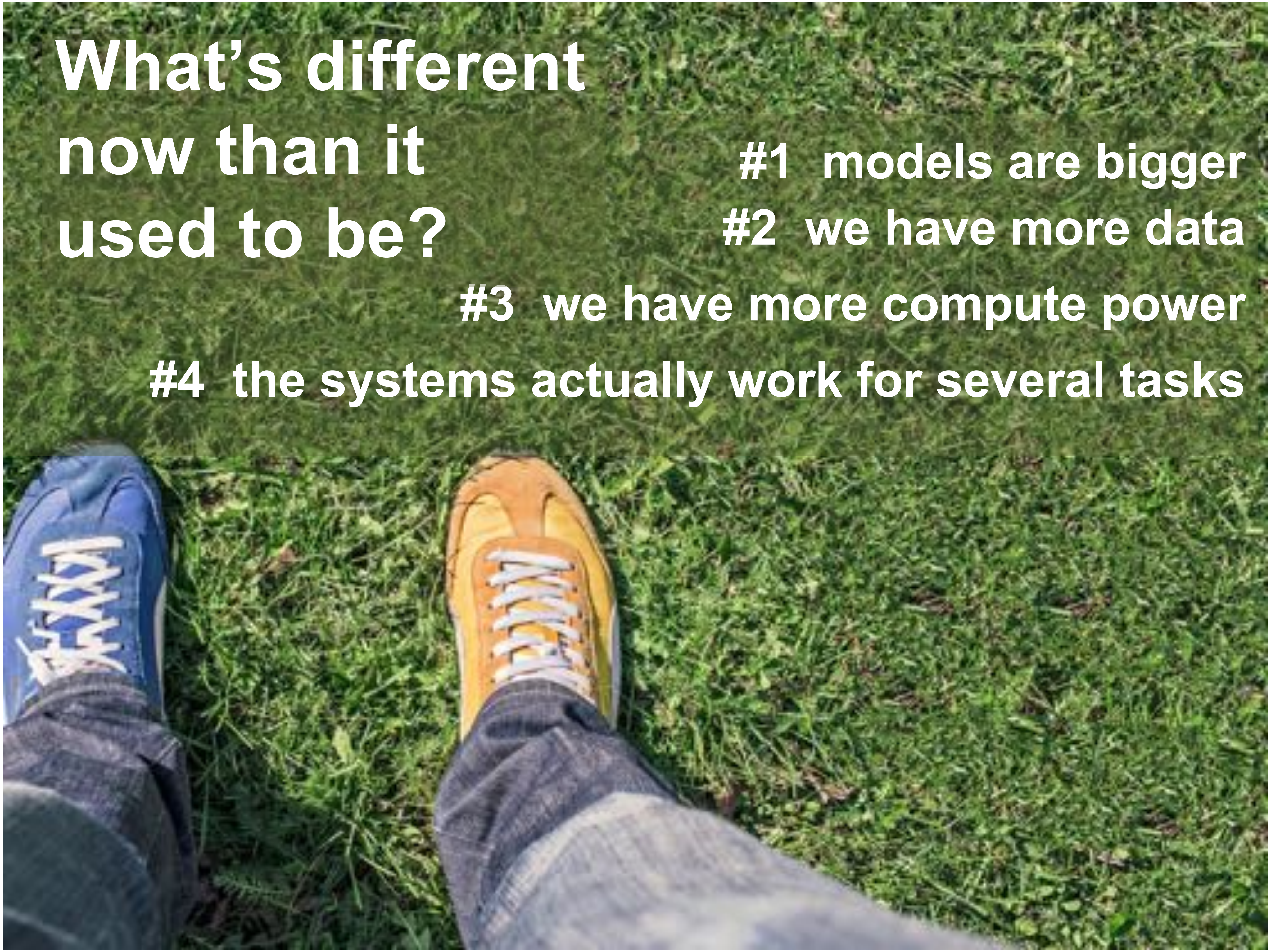
Turing Awardees 2019

Overall Picture



The Seasons of AI




What's different now than it used to be?
#1 models are bigger
#2 we have more data
#3 we have more compute power
#4 the systems actually work for several tasks

But, what exactly is Deep Learning?

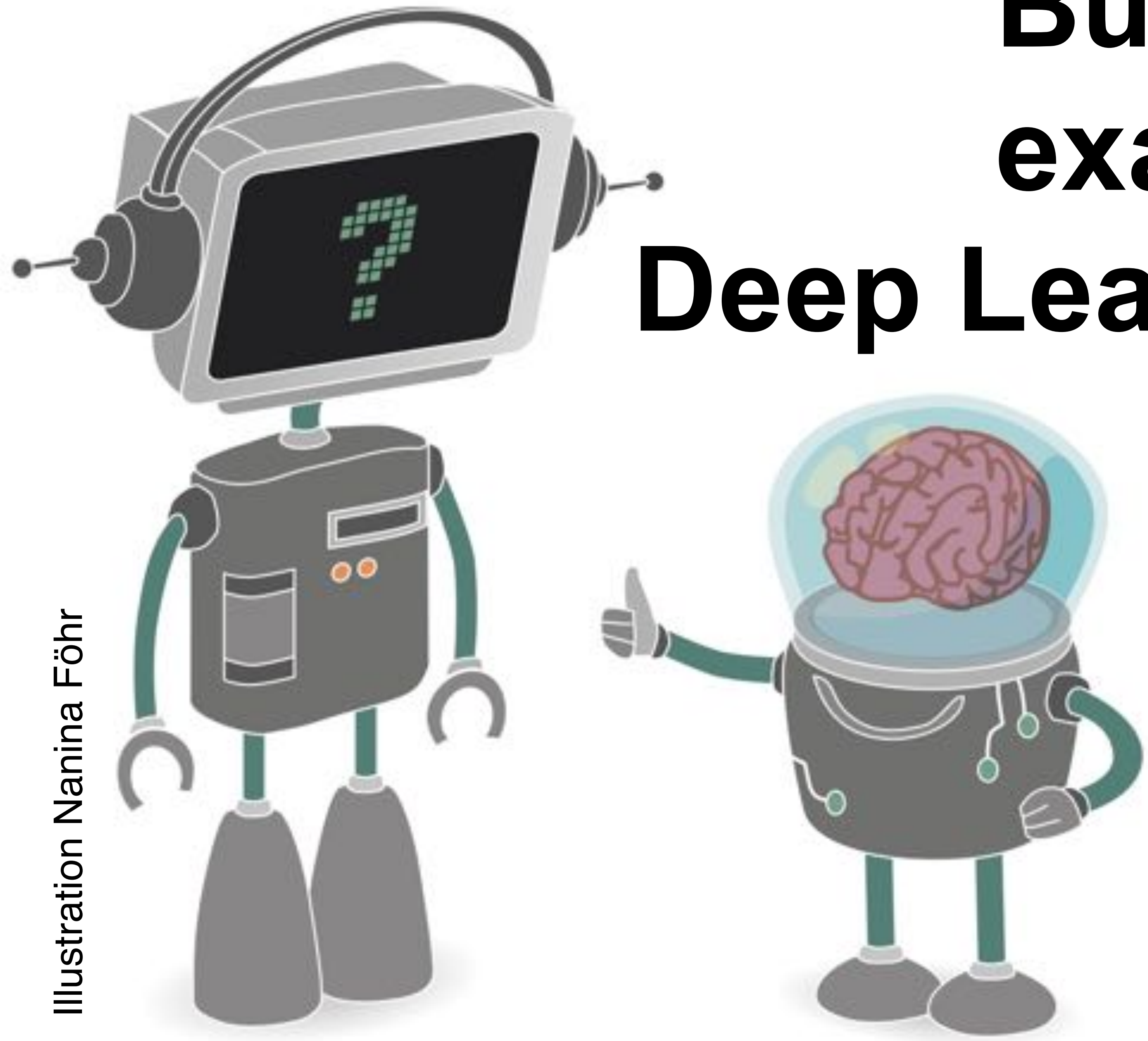
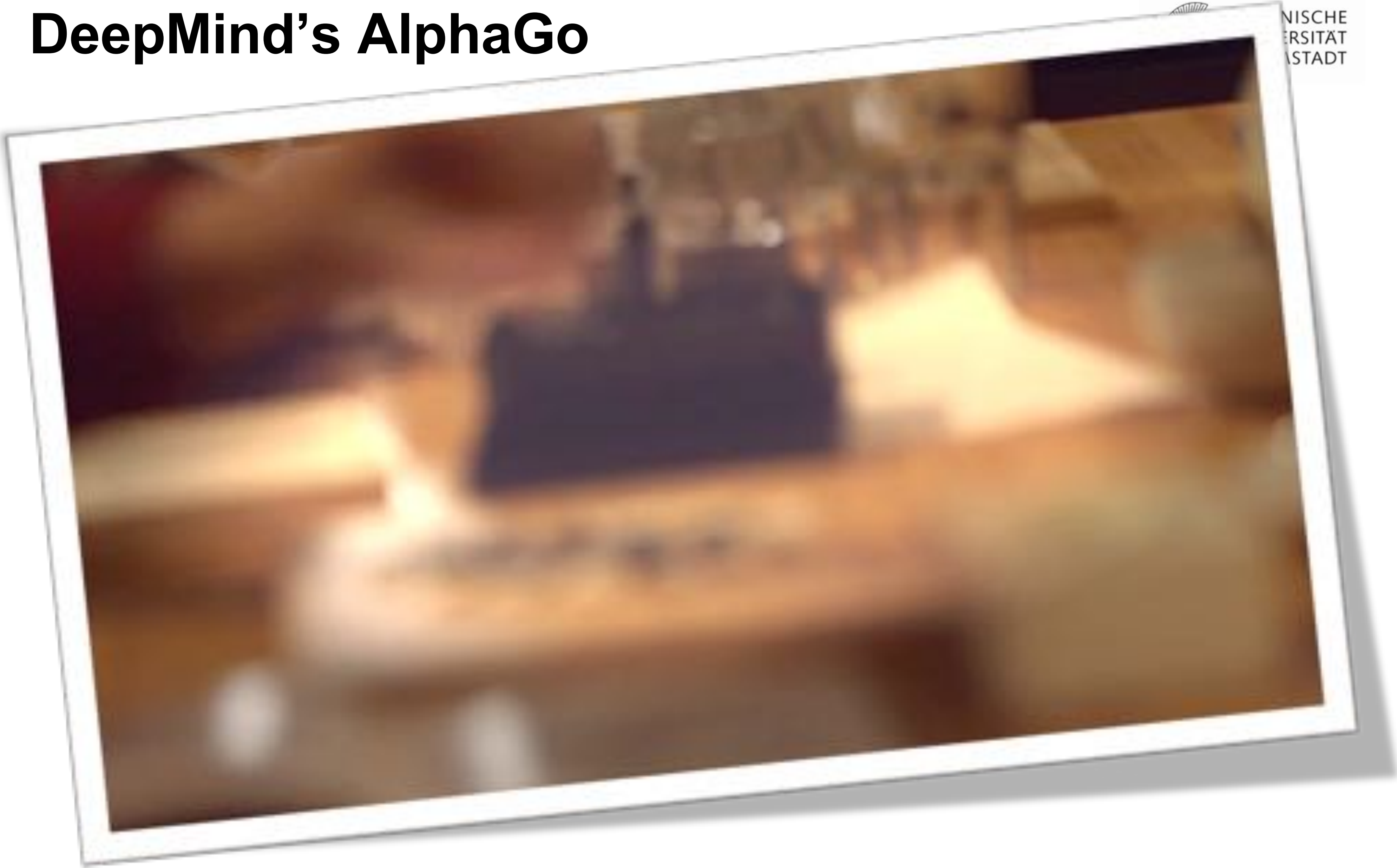


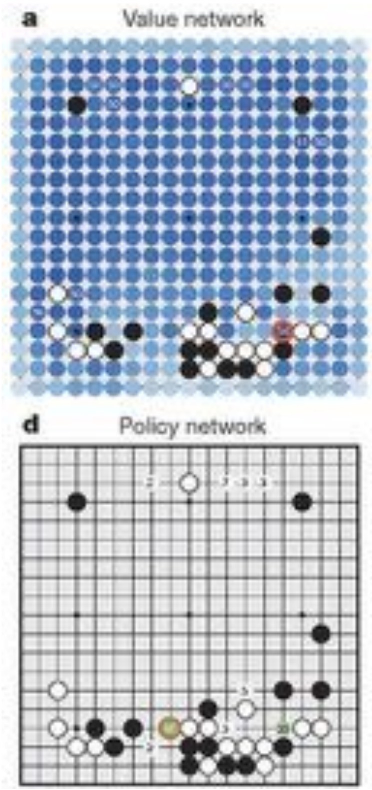
Illustration Nanina Föhr

DeepMind's AlphaGo



Watch NATURE video at https://www.youtube.com/watch?v=g-dKXOIsf98

DeepMind’s AlphaGo



Deep policy network is trained to produce probability map of promising moves. The deep value network is used to prune the search tree (monte-carlo tree search); so there is a lot of classical AI machinery around the deep part.

And yes, the machine may also learn to play other games

Goal of Deep Architectures

High-level semenatical representations

Edges, local shapes, object parts

Low level representation

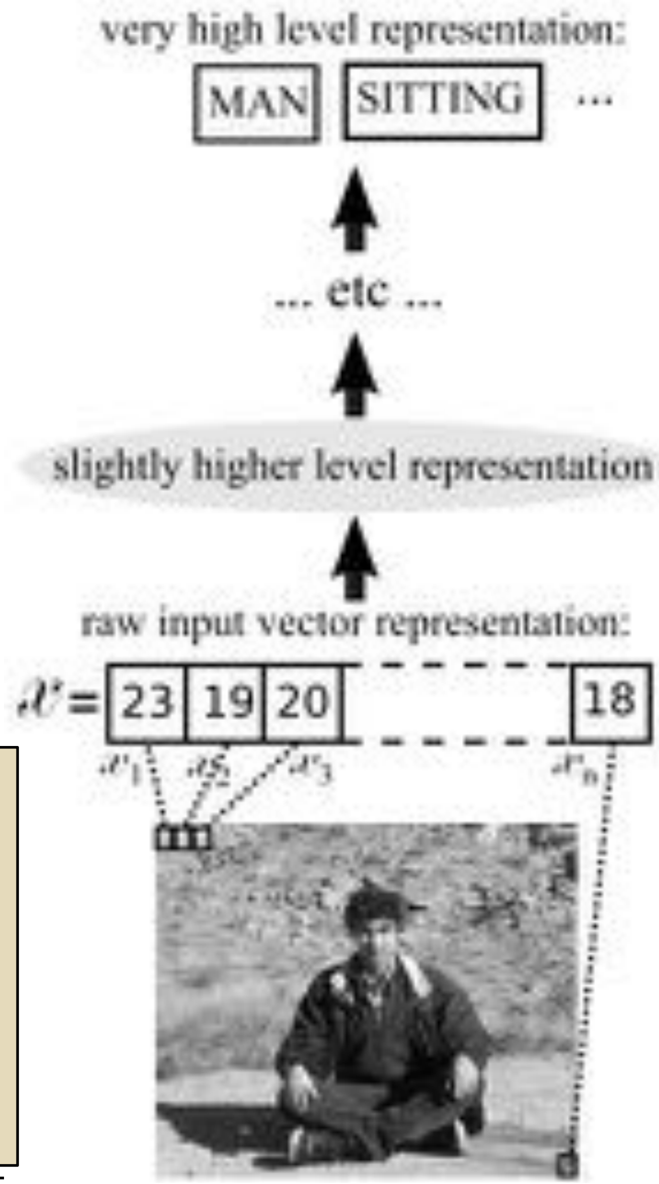


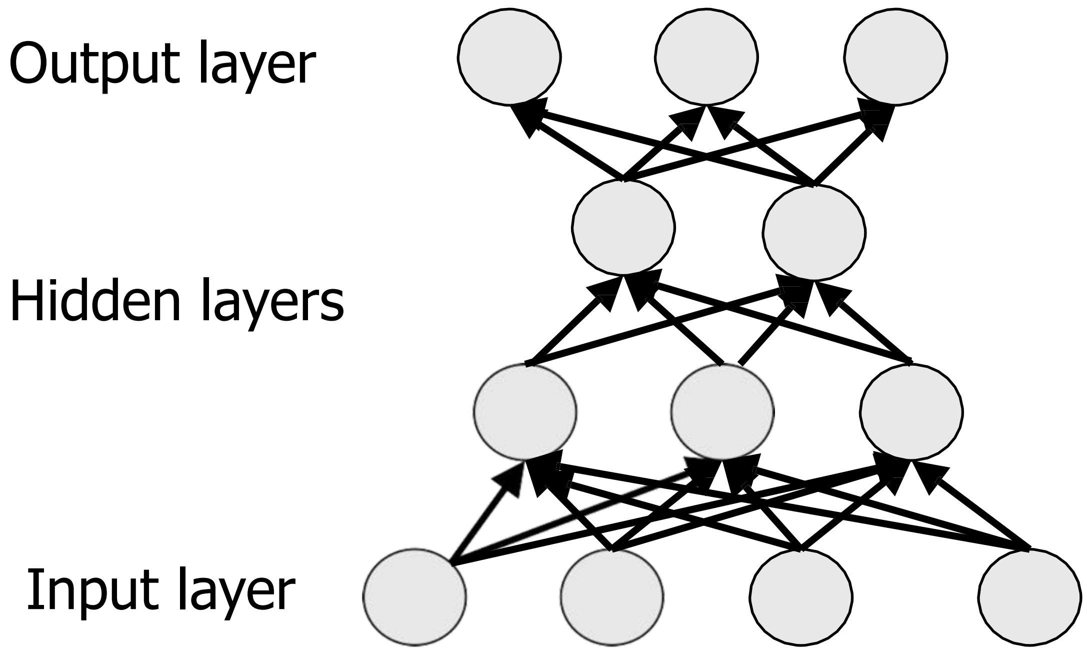
Figure is from Yoshua Bengio

Deep learning methods aim at

- **learning feature hierarchies**
- where features from higher levels of the hierarchy are formed by lower level features.

Deep Architectures

Deep architectures are composed of multiple levels of non-linear operations, such as neural nets with many hidden layers.



Examples of non-linear activations:

$$\tanh(x)$$

$$\sigma(x) = (1 + e^{-x})^{-1}$$

$$\max(0, x)$$

In practice, NN with multiple hidden layers work better than with a single hidden layer.

Artificial Neural Networks

Inspiration from the brain:

- **many small interconnected units (neurons)**
- **learning happens by changing the strength of connections (synapses)**
- **behavior of the whole is more than the sum of the parts**

Frank Rosenblatt (1928-1971)

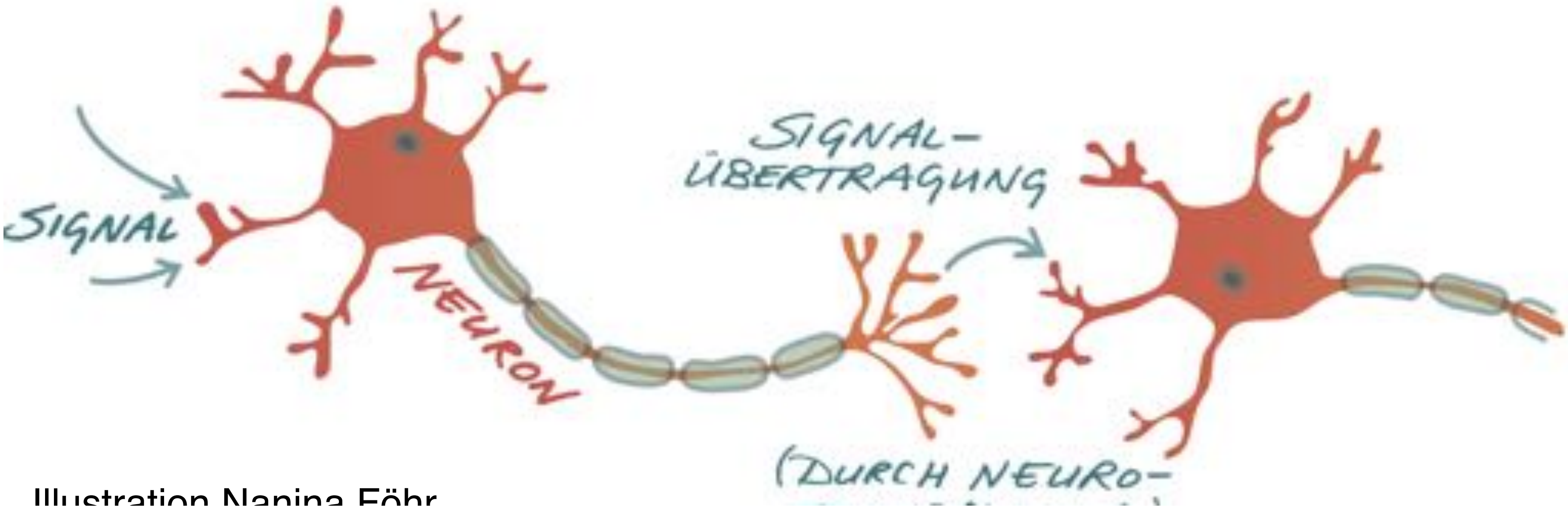
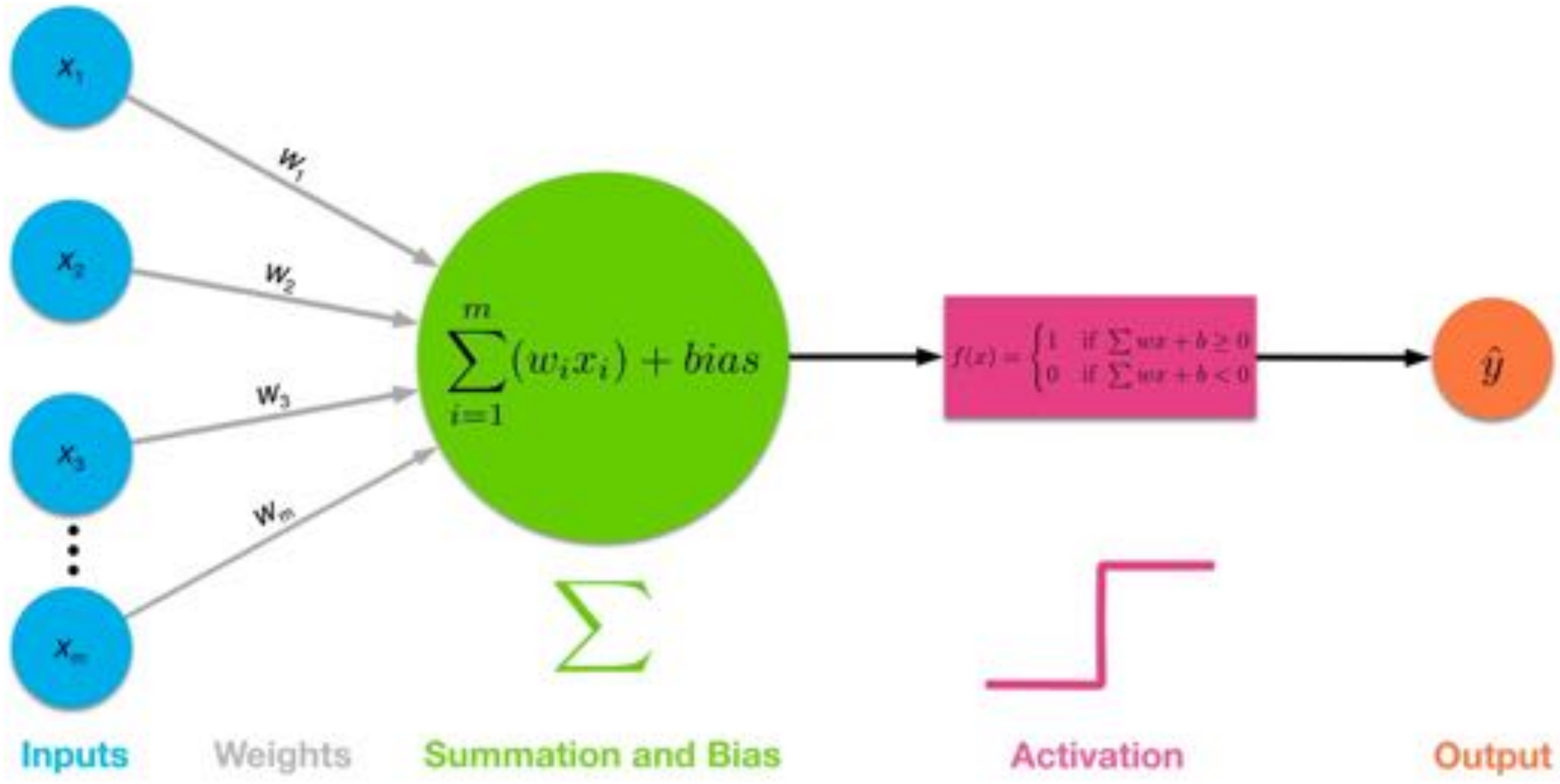
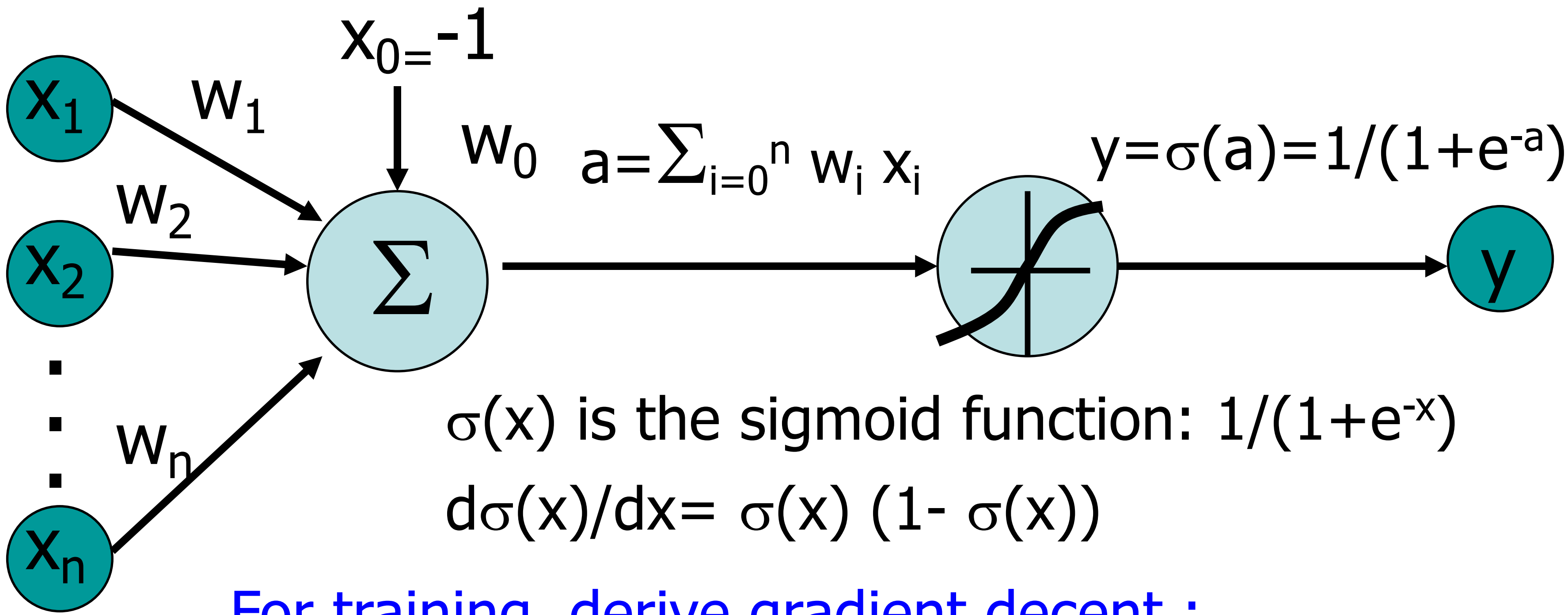


Illustration Nanina Föhr

Abstract Neural Unit



Commonly, neurons are encoded as **Sigmoid Unit (but other units are possible)**



$\sigma(x)$ is the sigmoid function: $1/(1+e^{-x})$

$d\sigma(x)/dx = \sigma(x)\,(1 - \sigma(x))$

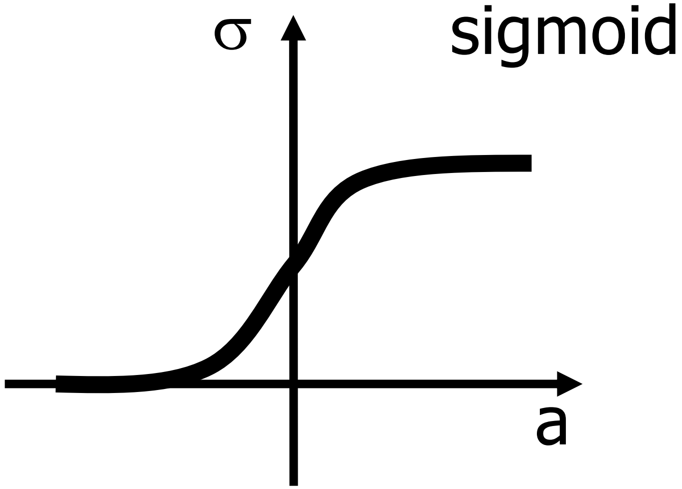
For training, derive gradient decent :

- one sigmoid function

  $\partial E/\partial w_i = -\sum_p (t^p - y)\, y\, (1-y)\, x_i^p$

- Multilayer networks of sigmoid units use backpropagation

Gradient Descent Rule for Sigmoid Output Function

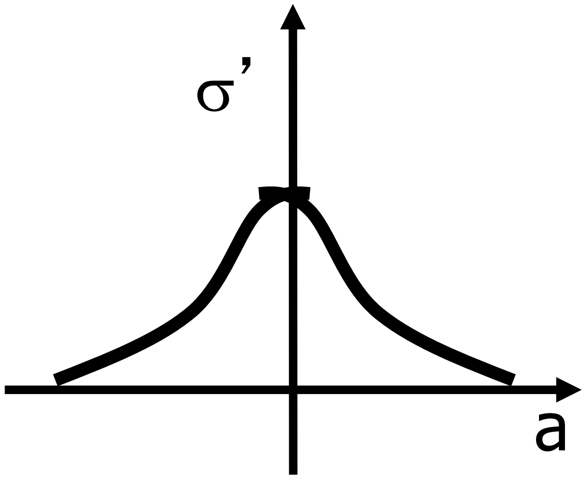


$$E^p[w_1,\ldots,w_n] = ½\,(t^p-y^p)^2$$

$$\partial E^p/\partial w_i = \partial/\partial w_i\ ½\,(t^p-y^p)^2$$

$$= \partial/\partial w_i\ ½\,(t^p- \sigma(\textstyle\sum_i w_i\, x_i^p))^2$$

$$= (t^p-y^p)\ \sigma\,'(\textstyle\sum_i w_i\, x_i^p)\ (-x_i^p)$$

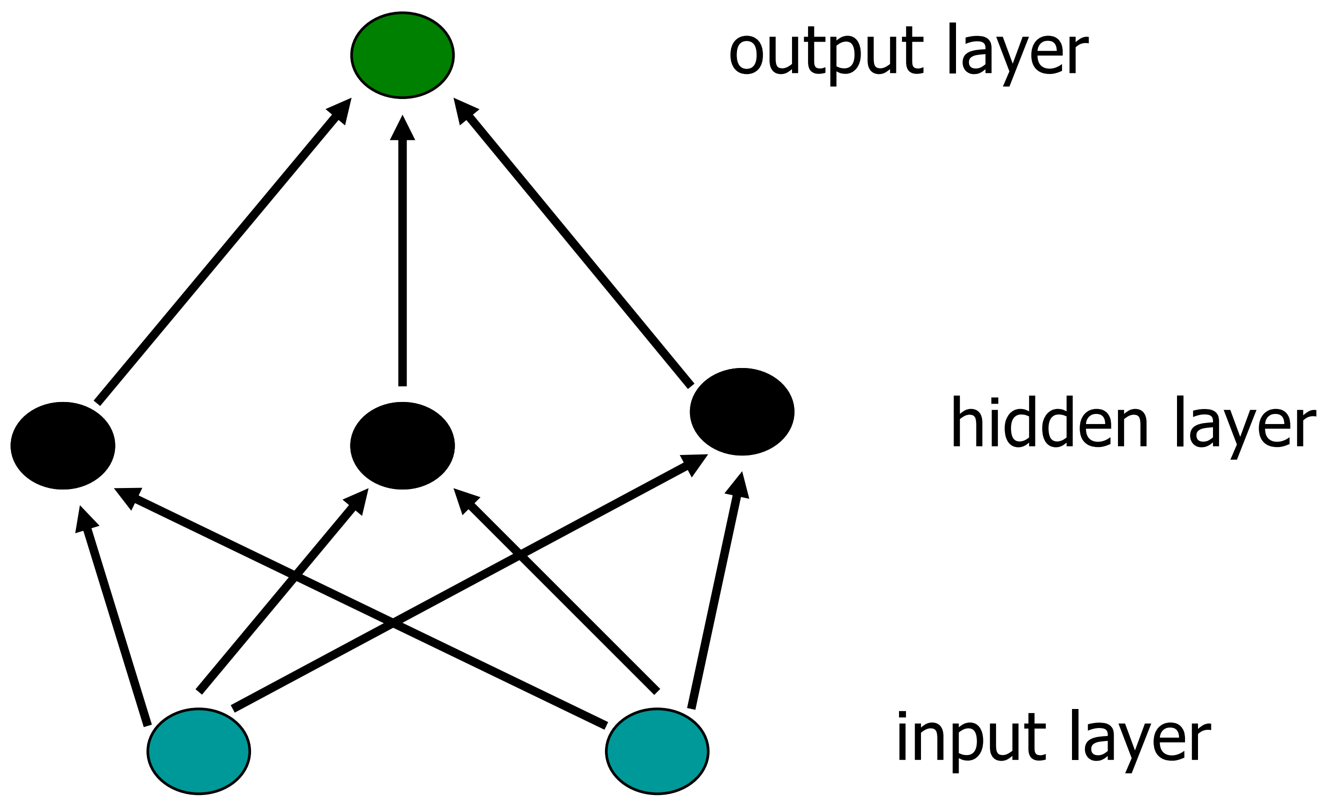


for $y=\sigma(a) = 1/(1+e^{-a})$

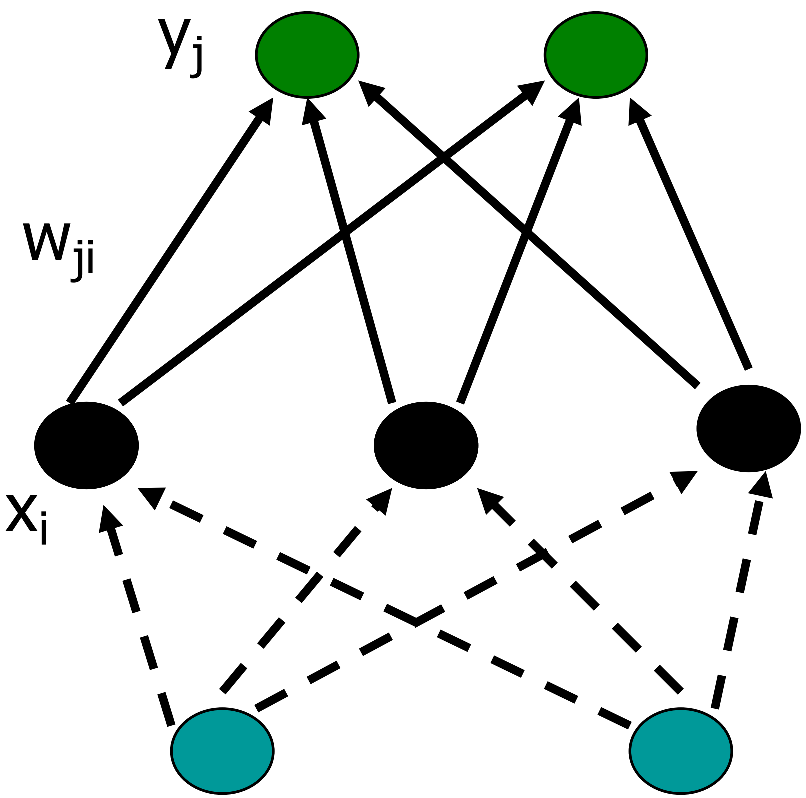
$\sigma'(a)= e^{-a}/(1+e^{-a})^2=\sigma(a)\ (1-\sigma(a))$

$$w'_i= w_i + \Delta w_i = w_i + \alpha\ y(1-y)(t^p-y^p)\ x_i^p$$

Build (feedforward) Multi-Layer Networks by sticking together units



Training-Rule for Weights to the Output Layer



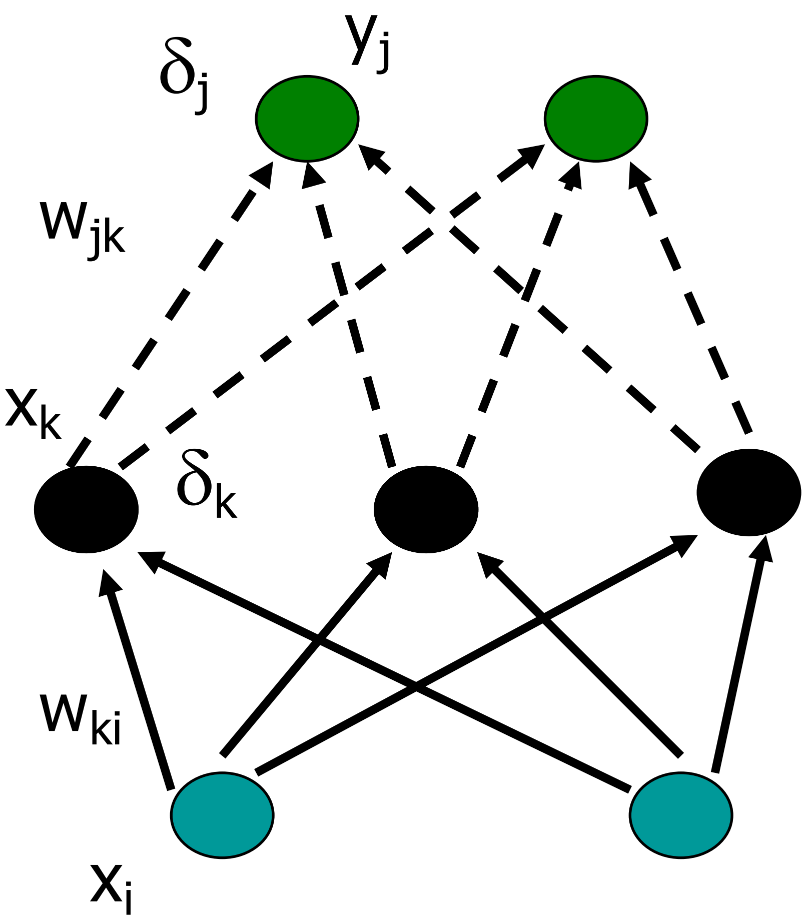
$$E^p[w_{ij}] = \tfrac{1}{2} \Sigma_j (t_j^p - y_j^p)^2$$

$$\partial E^p / \partial w_{ji} = \partial / \partial w_{ji} \tfrac{1}{2} \Sigma_j (t_j^p - y_j^p)^2$$
$$= \ldots$$
$$= - y_j^p (1 - y^p_j)(t^p_j - y^p_j) \; x_i^p$$

$$\Delta w_{ji} = \alpha \; y_j^p (1 - y_j^p) (t^p_j - y_j^p) \, x_i^p$$
$$= \alpha \; \delta_j^p \, x_i^p$$

with $\delta_j^p := y_j^p (1 - y_j^p) (t^p_j - y_j^p)$

Training-Rule for Weights to the Output Layer



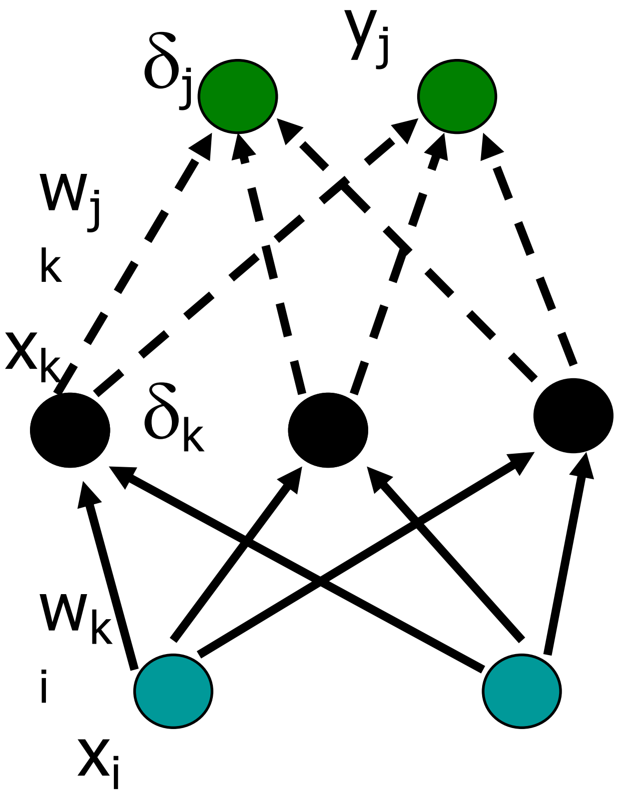
Credit assignment problem:
No target values t for hidden layer units.

Error for hidden units?

$$\delta_k = \Sigma_j \, w_{jk} \, \delta_j \, y_j \, (1-y_j)$$

$$\Delta w_{ki} = \alpha \;\; x_k^p(1-x_k^p) \, \delta_k^p \, x_i^p$$

Training-Rule for Weights to the Output Layer



$$E^p[w_{ki}] = \tfrac{1}{2} \Sigma_j (t_j^p - y_j^p)^2$$

$$\partial E^p/\partial w_{ki} = \partial/\partial w_{ki}\ \tfrac{1}{2} \Sigma_j (t_j^p - y_j^p)^2$$

$$= \partial/\partial w_{ki}\ \tfrac{1}{2}\Sigma_j (t_j^p - \sigma(\Sigma_k w_{jk}\ x_k^p))^2$$

$$= \partial/\partial w_{ki}\ \tfrac{1}{2}\Sigma_j (t_j^p - \sigma(\Sigma_k w_{jk}\ \sigma(\Sigma_i w_{ki}\ x_i^p)))^2$$

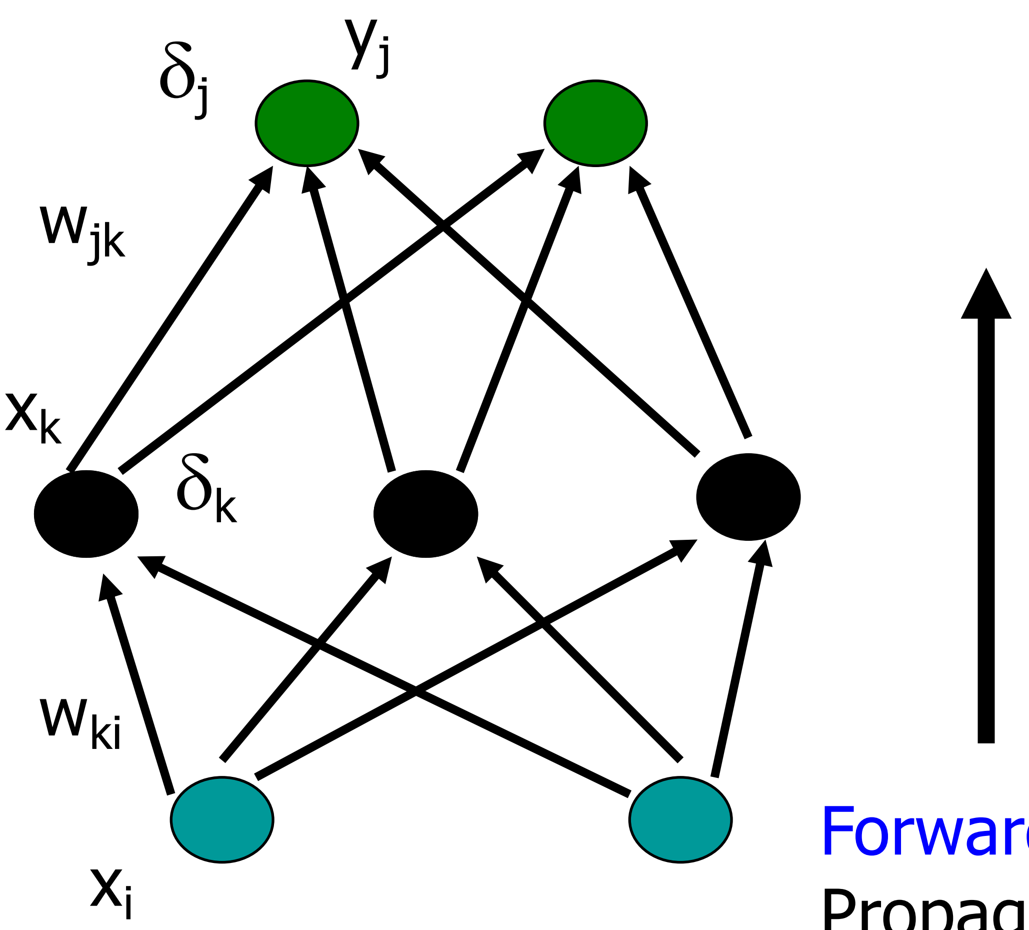
$$= -\Sigma_j (t_j^p - y_j^p)\ \sigma'_j(a)\ w_{jk}\ \sigma'_k(a)\ x_i^p$$

$$= -\Sigma_j \delta_j\ w_{jk}\ \sigma'_k(a)\ x_i^p$$

$$= -\Sigma_j \delta_j\ w_{jk}\ x_k\ (1-x_k)\ x_i^p$$

$$\Delta w_{ki} = \alpha\ \delta_k\ x_i^p \quad \text{with } \delta_k = \Sigma_j \delta_j\ w_{jk}\ x_k(1-x_k)$$

Backpropagation



Backward step:
propagate errors from
output to hidden layer

Forward step:
Propagate activation
from input to output layer

Deep Convolutional Networks CNNs

Compared to standard neural networks with similarly-sized layers,

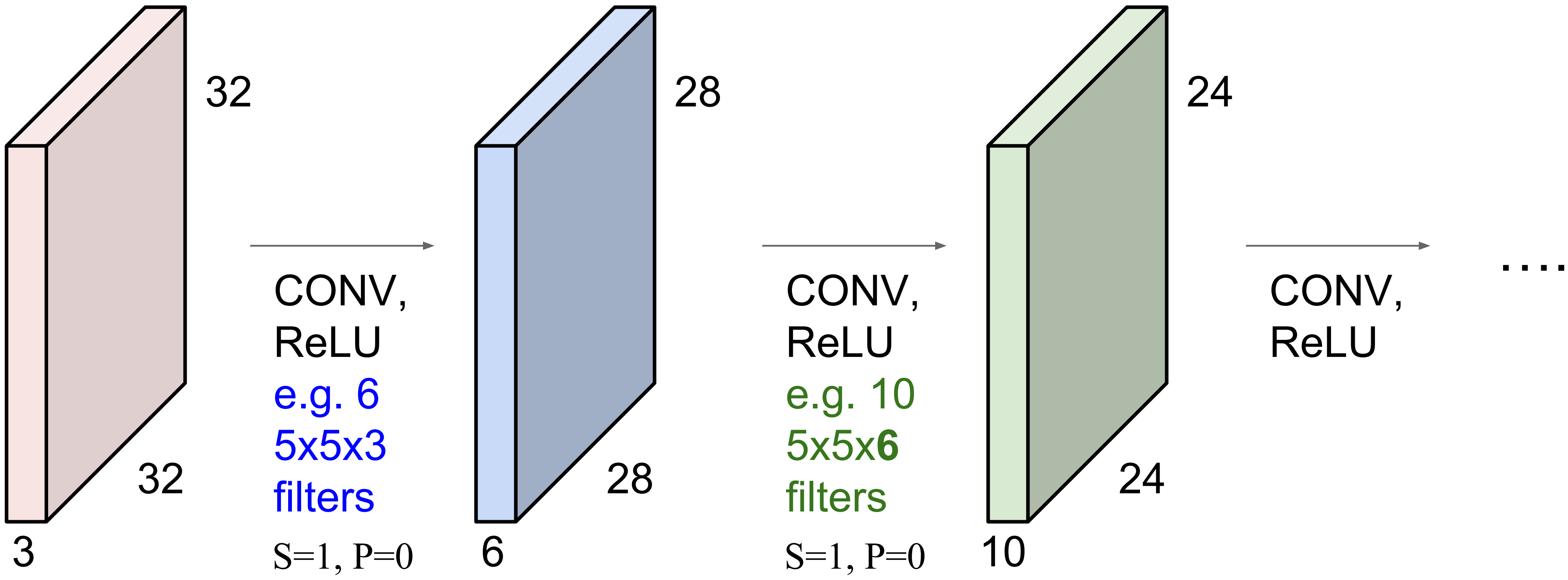
- CNNs have much fewer connections and parameters
- and so they are easier to train
- and typically have more than five layers (a number of layers which makes fully-connected neural networks almost impossible to train properly when initialized randomly)

LeNet, 1998 LeCun Y, Bottou L, Bengio Y, Haffner P: Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE

AlexNet, 2012 Krizhevsky A, Sutskever I, Hinton G: ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

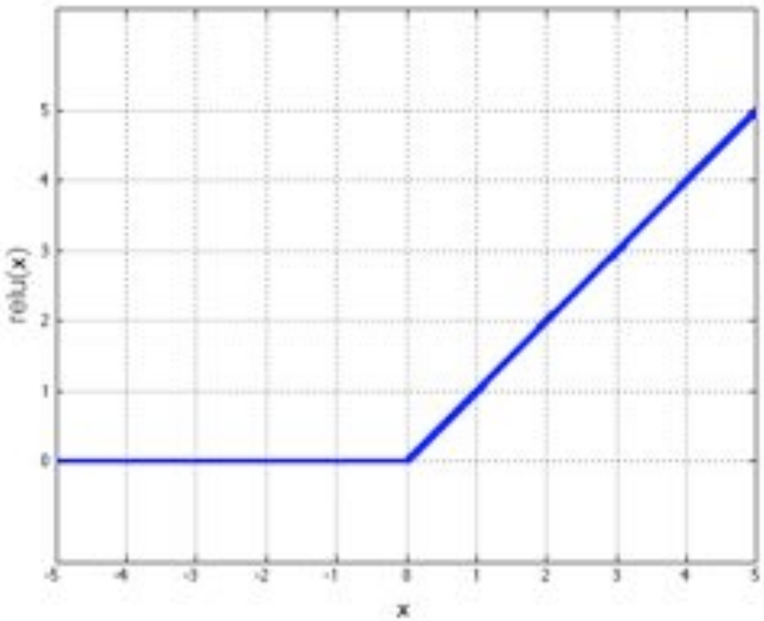
You start with convolutional layers

Preview: ConvNet is a sequence of Convolutional Layers, interspersed with activation functions



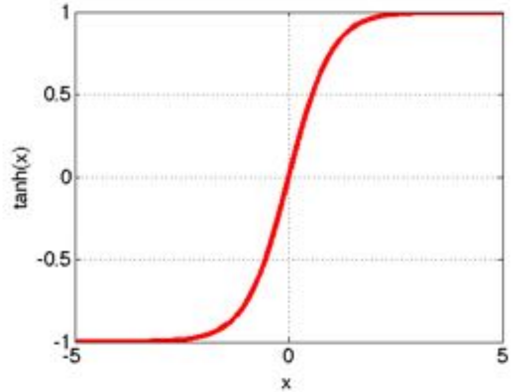
Where is ReLU?

- Non-linear activation function are applied per-element
- Rectified linear unit (ReLU):
  - max(0,x)
  - makes learning faster (in practice x6)
  - avoids saturation issues (unlike sigmoid, tanh)
  - simplifies training with backpropagation
  - preferred option (works well)

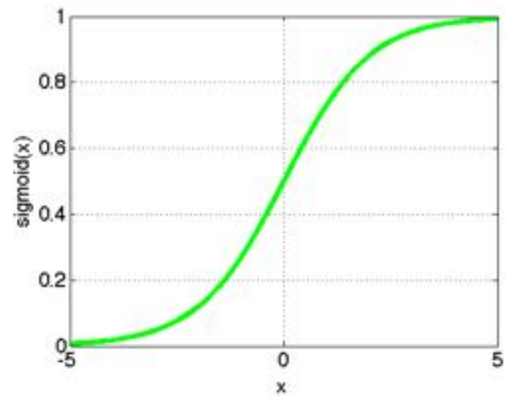


Other examples:

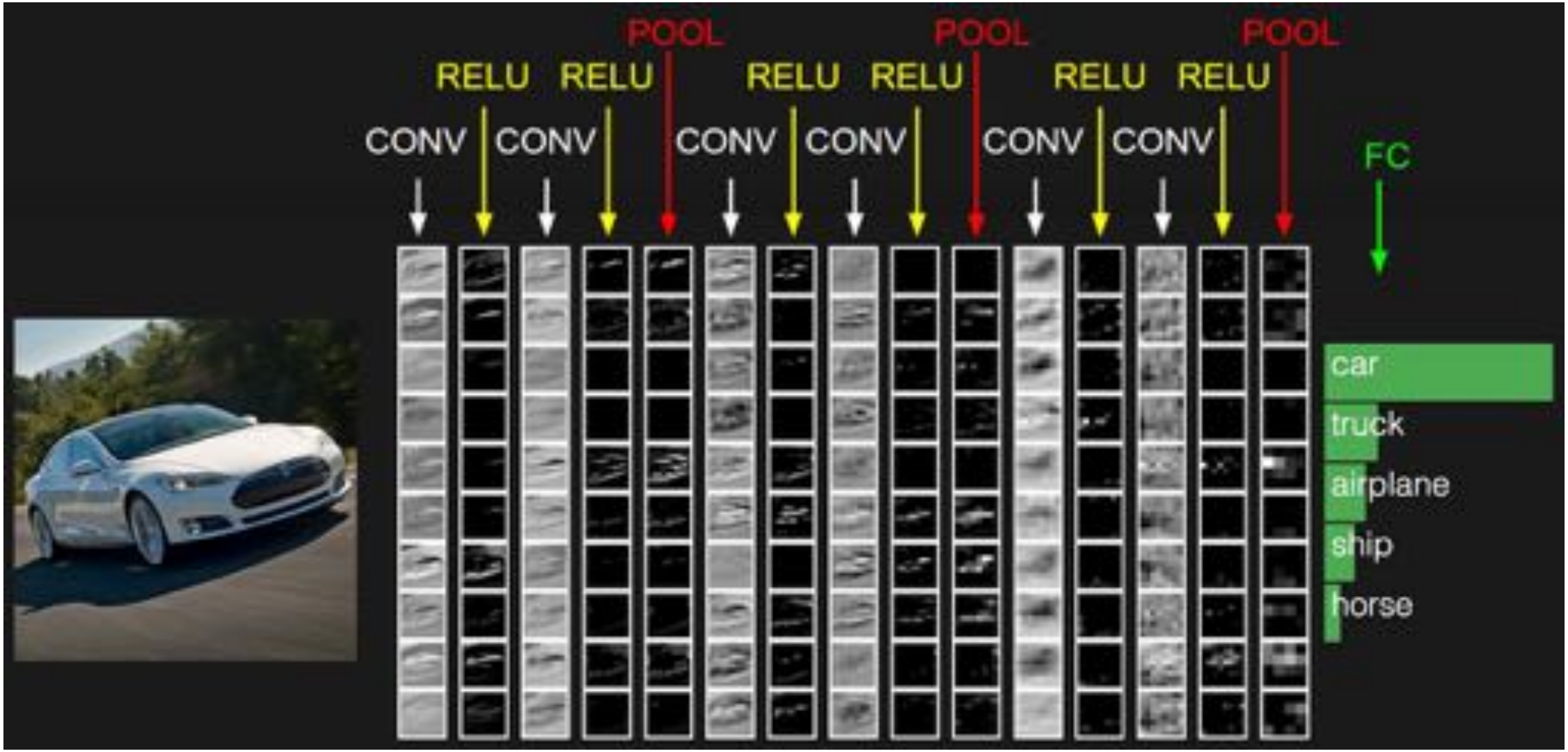
tanh(x)



sigmoid(x)=$(1+e^{-x})^{-1}$

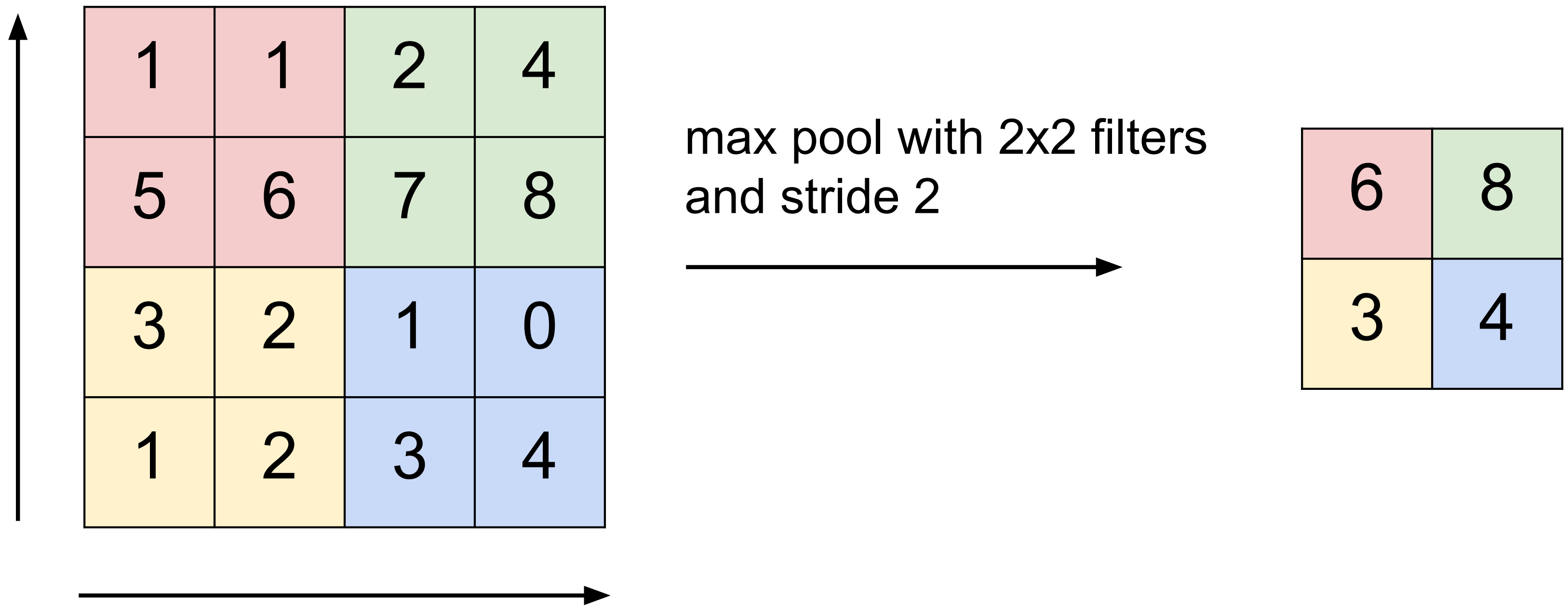


Then you pool to reduce complexity



Max Pooling

Single activation map

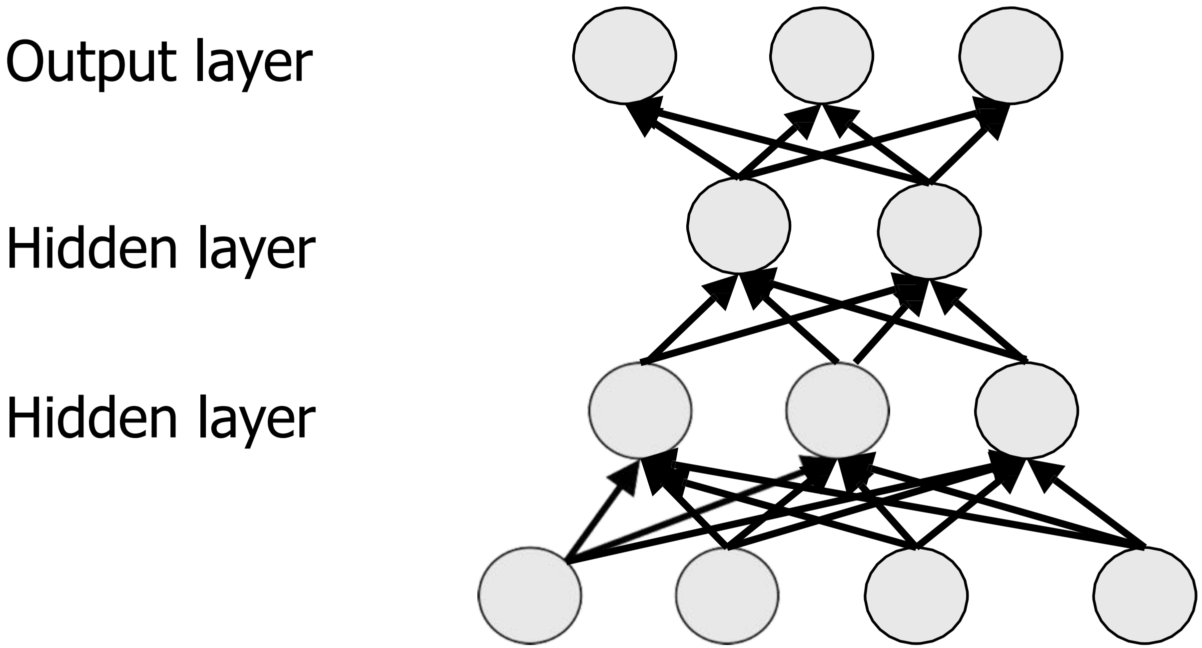


Alternatives:

- sum pooling
- overlapping pooling

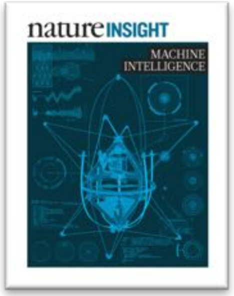
Finally some fully connected layers

Contains neurons that connect to the entire input volume, as in ordinary Neural Networks:



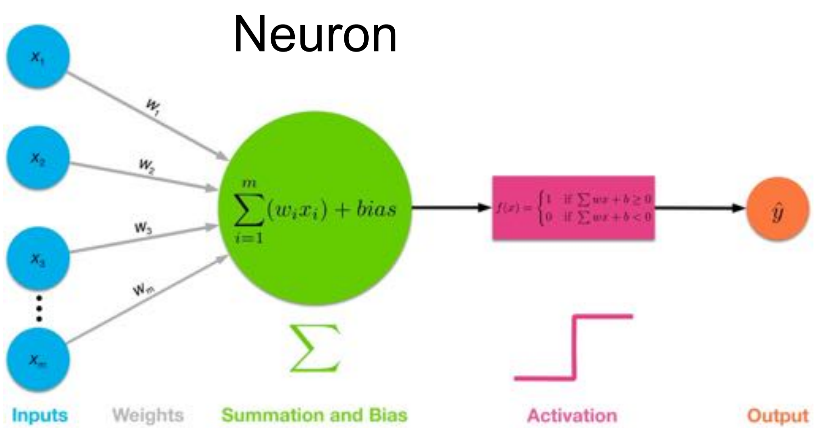
neurons between two adjacent layers are fully pairwise connected, but neurons within a single layer share no connections

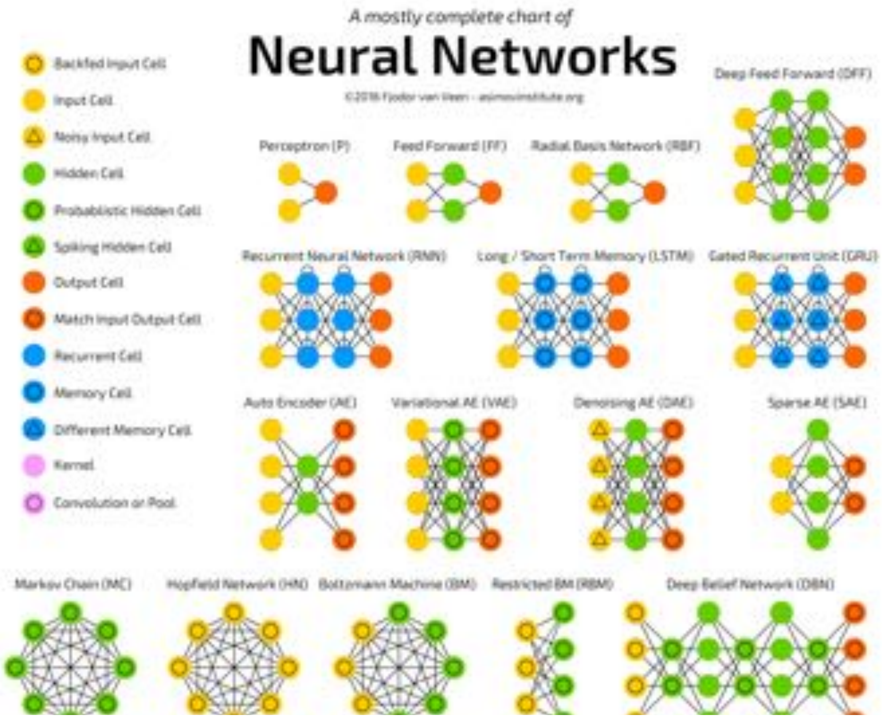
Deep Neural Networks



Potentially much more powerful than shallow architectures, represent computations

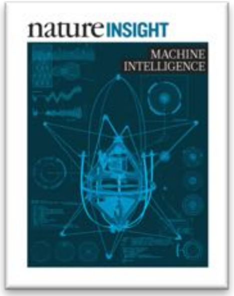
[LeCun, Bengio, Hinton Nature 521, 436–444, 2015]





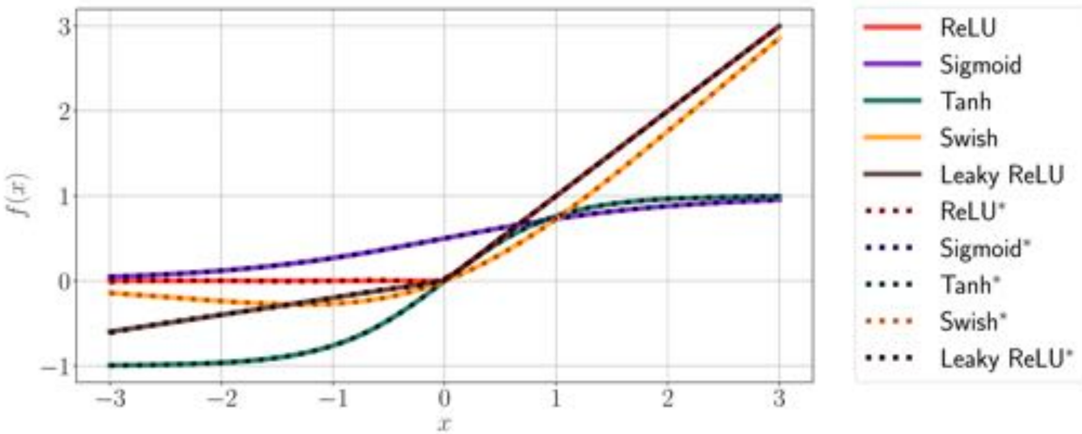
Differentiable Programming

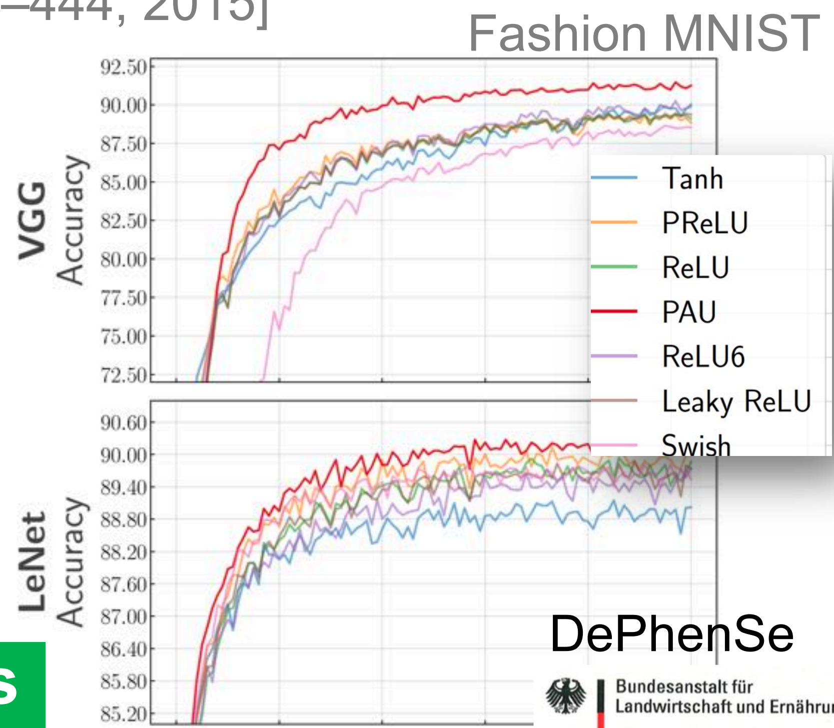
Deep Neural Networks



Potentially much more powerful than shallow architectures, represent computations

[LeCun, Bengio, Hinton Nature 521, 436–444, 2015]





https://github.com/ml-research/pau

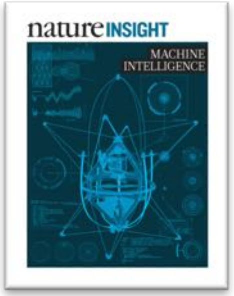
E2E-Learning Activation Functions

[Molina, Schramowski, Kersting arxiv:1901.03704 2019]

DePhenSe

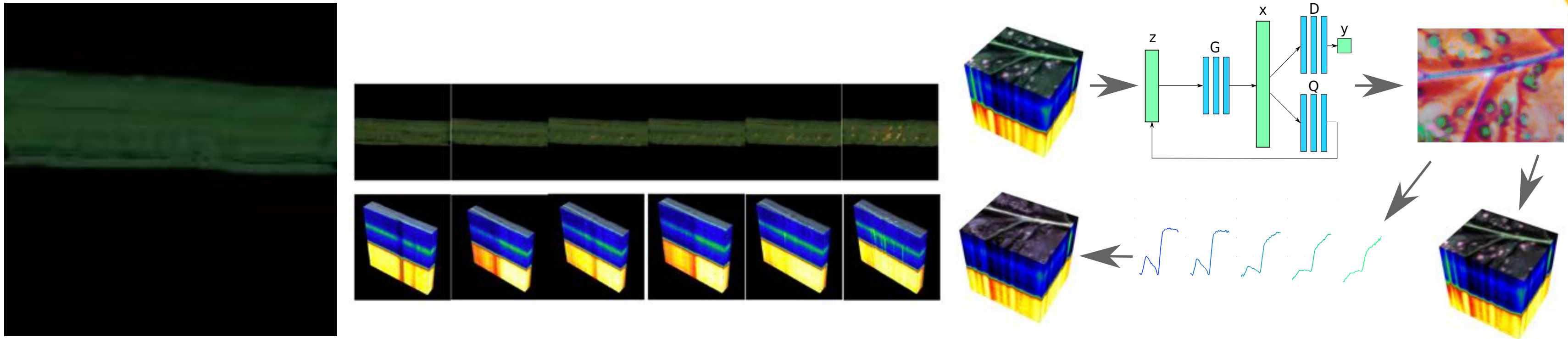
Bundesanstalt für Landwirtschaft und Ernährung

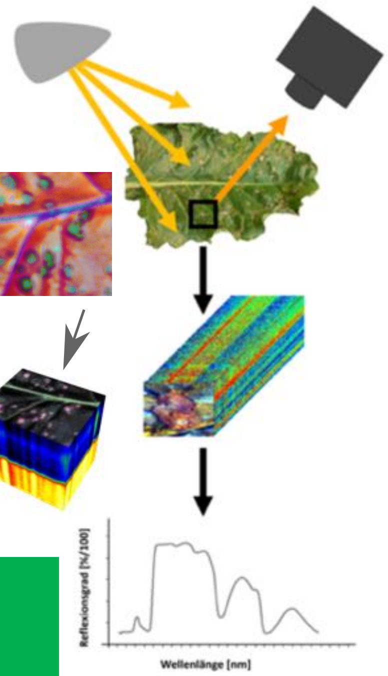
Deep Neural Networks



Potentially much more powerful than shallow architectures, represent computations

[LeCun, Bengio, Hinton Nature 521, 436–444, 2015]





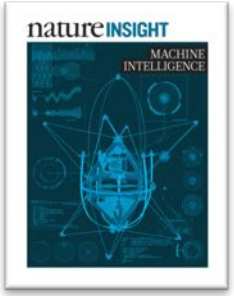
They "develop intuition" about complicated biological processes and generate scientific data

[Schramowski, Brugger, Mahlein, Kersting 2019]

DePhenSe

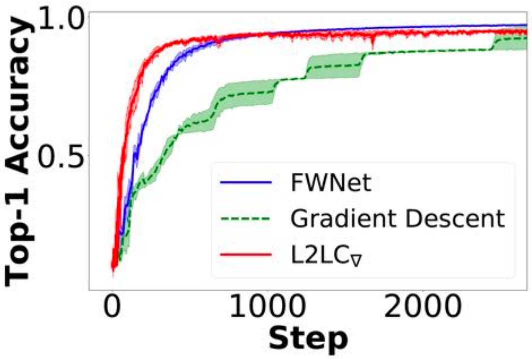
Bundesanstalt für Landwirtschaft und Ernährung

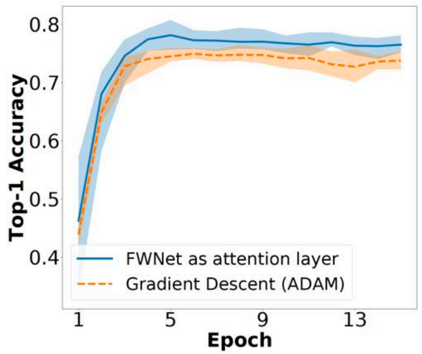
Deep Neural Networks

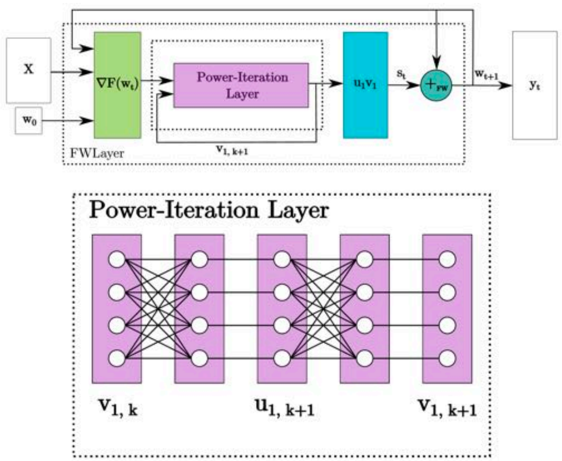


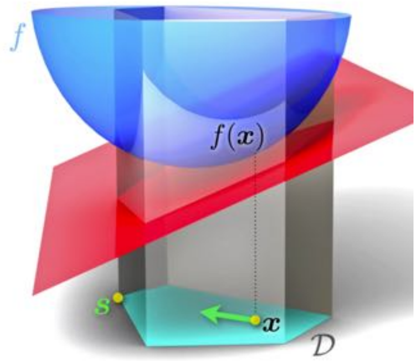
Potentially much more powerful than shallow architectures, represent computations

[LeCun, Bengio, Hinton Nature 521, 436–444, 2015]









They “invent” constrained optimizers

DePhenSe

Bundesanstalt für Landwirtschaft und Ernährung

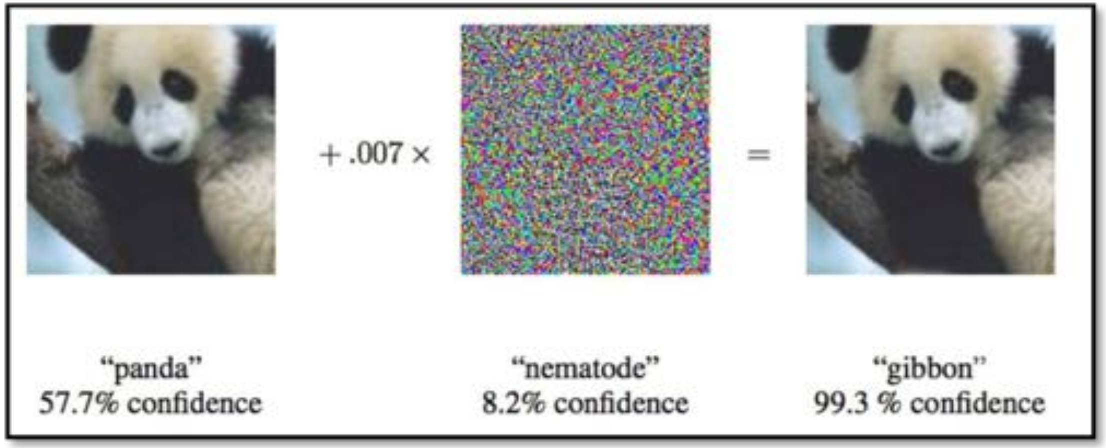
[Schramowski, Bauckhage, Kersting arXiv:1803.04300, 2018]

They are not human!

Fundamental Differences

Sharif et al., 2015

Brown et al. (2017)



Google, 2015

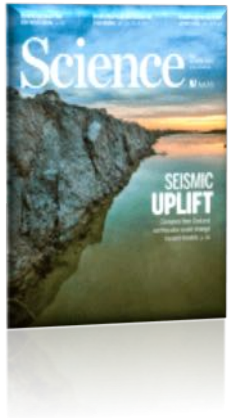
REPORTS | PSYCHOLOGY

Semantics derived automatically from language corpora contain human-like biases

Aylin Caliskan[1,*], Joanna J. Bryson[1,2,*], Arvind Narayanan[1,*]

+ See all authors and affiliations

Science 14 Apr 2017:
Vol. 356, Issue 6334, pp. 183-186
DOI: 10.1126/science.aal4230



However, they can also help us on the quest for a „good“ AI

How could an AI programmed by humans, with no more moral expertise than us, recognize (at least some of) our own civilization’s ethics as moral progress as opposed to mere moral instability?

„The Ethics of Artificial Intelligence“ Cambridge Handbook of Artificial Intelligence, 2011

Nick Bostrom

Future of Humanity Institute

UNIVERSITY OF OXFORD

Eliezer Yudkowsky

MIRI

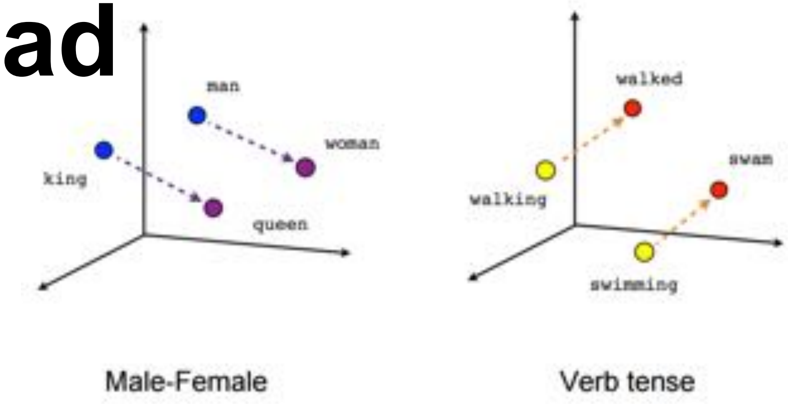
MACHINE INTELLIGENCE RESEARCH INSTITUTE

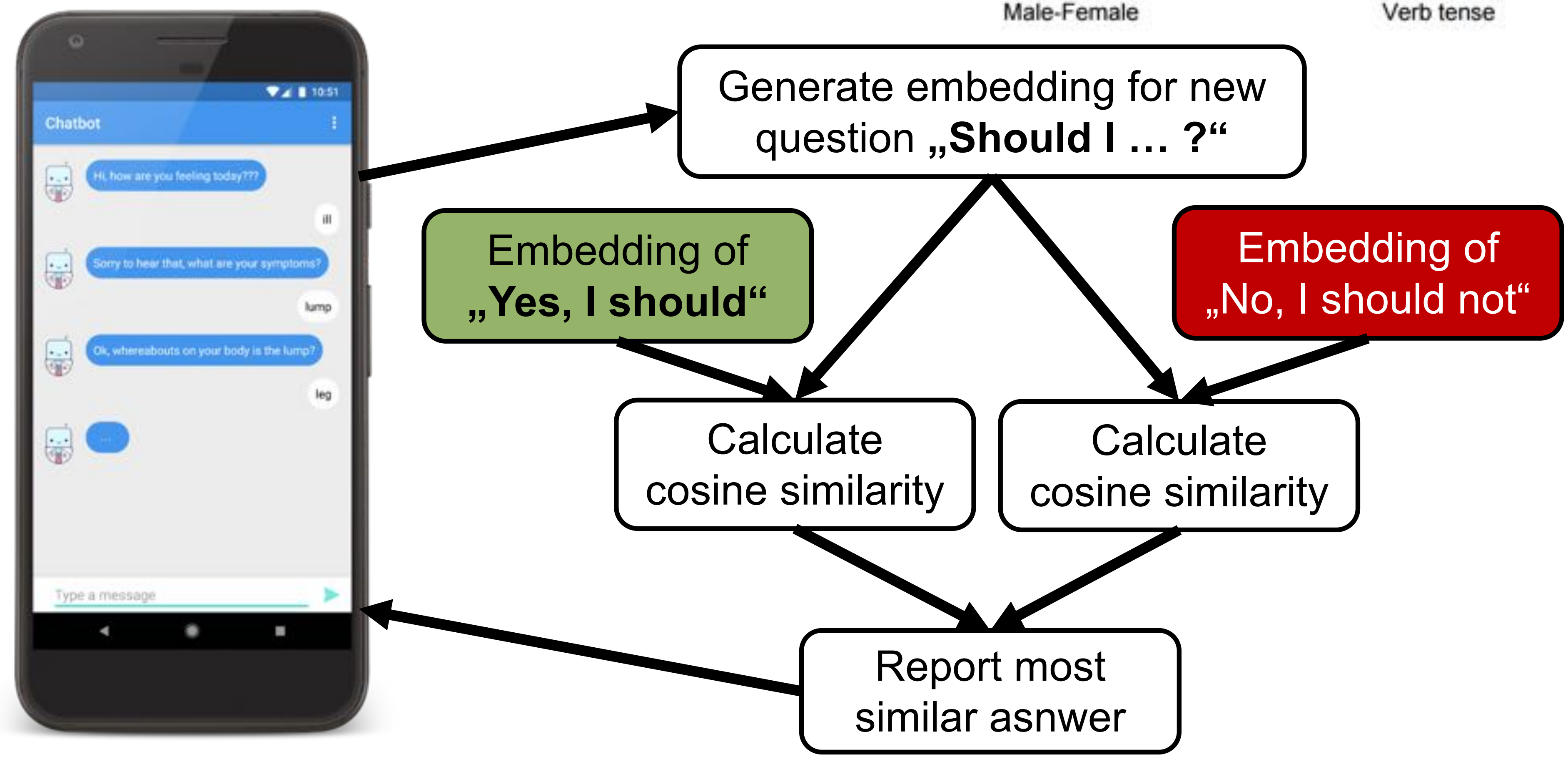
The Moral Choice Machine

Not all stereotypes are bad

[Jentzsch, Schramowski, Rothkopf, Kersting AIES 2019]

AAAI / ACM conference on ARTIFICIAL INTELLIGENCE, ETHICS, AND SOCIETY





The Moral Choice Machine

Not all stereotypes are bad

[Jentzsch, Schramowski, Rothkopf, Kersting AIES 2019]

AAAI / ACM conference on
ARTIFICIAL INTELLIGENCE, ETHICS, AND SOCIETY

TECHNISCHE UNIVERSITÄT DARMSTADT

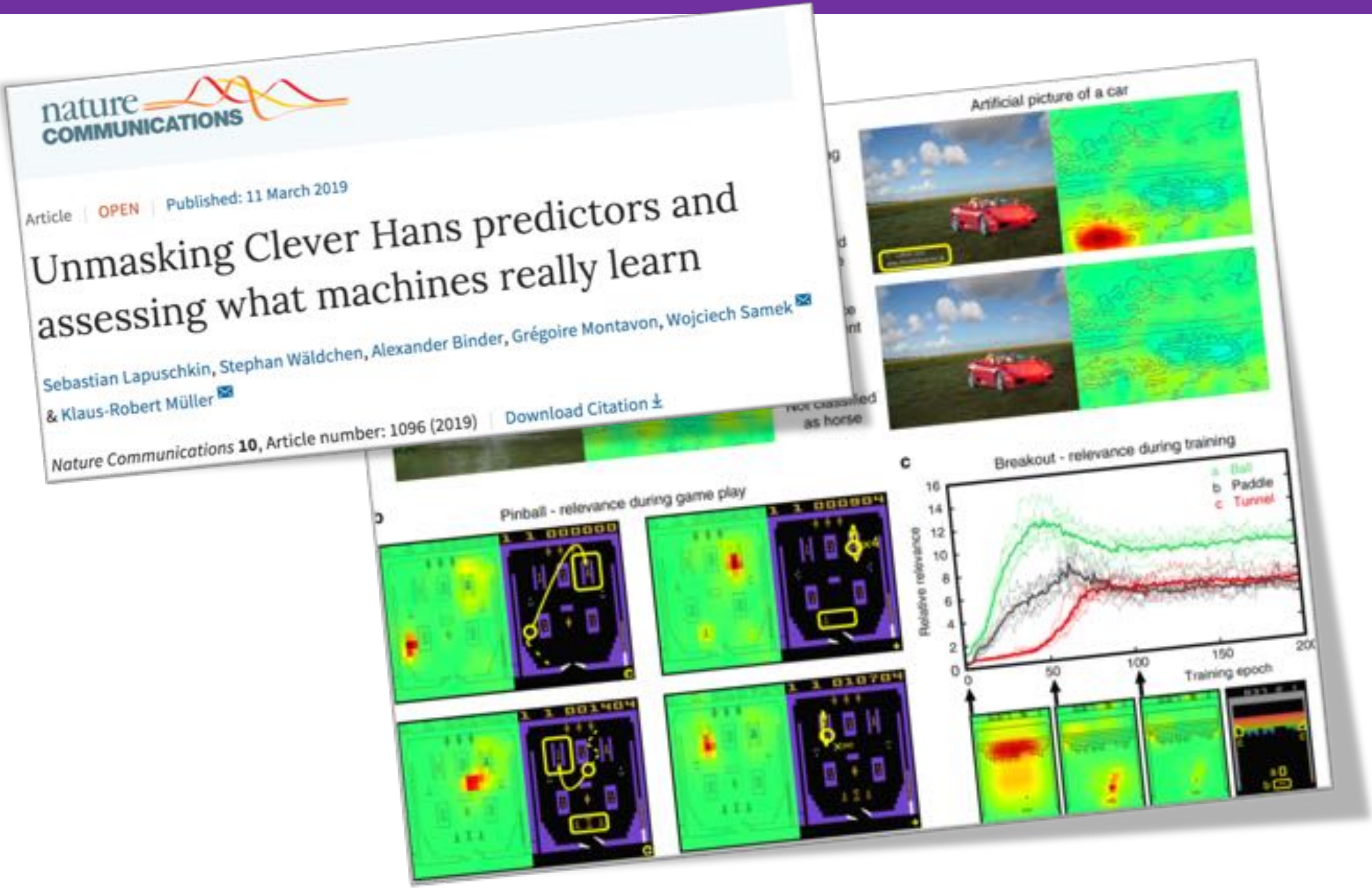
https://www.hr-fernsehen.de/sendungen-a-z/hauptsache-kultur/sendungen/hauptsache-kultur,sendung-56324.html

Video 05:10 Min.

Der Hamster gehört nicht in den Toaster – Wie Forscher von der TU Darmstadt versuchen, Maschinen ... [Videoseite]

hauptsache kultur | 14.03.19, 22:45 Uhr

Can we trust deep neural networks?


nature COMMUNICATIONS

Article | OPEN | Published: 11 March 2019

Unmasking Clever Hans predictors and assessing what machines really learn

Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek & Klaus-Robert Müller

Nature Communications 10, Article number: 1096 (2019) | Download Citation

DNNs often have no probabilistic semantics. They are not calibrated joint distributions.

$$P(Y|X) \neq P(Y,X)$$

MNIST

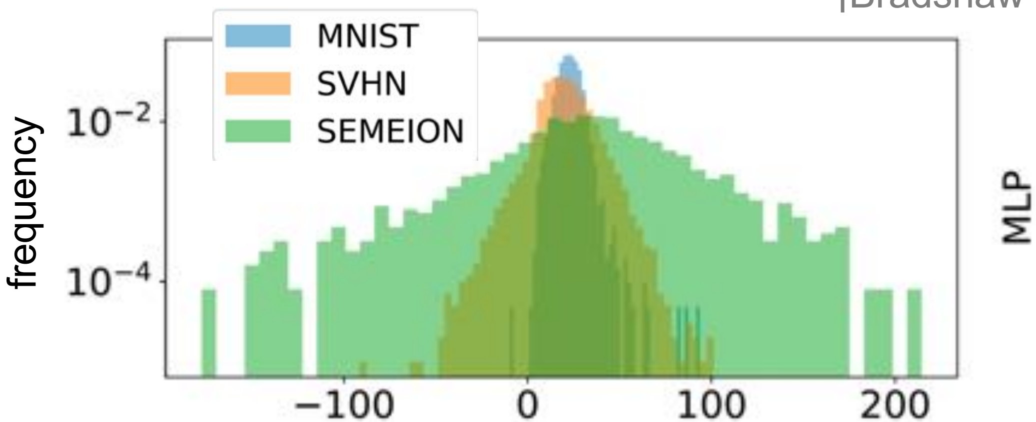
Train & Evaluate

SVHN

SEMEION

Transfer Testing

[Bradshaw et al. arXiv:1707.02476 2017]



Input log „likelihood“ (sum over outputs)

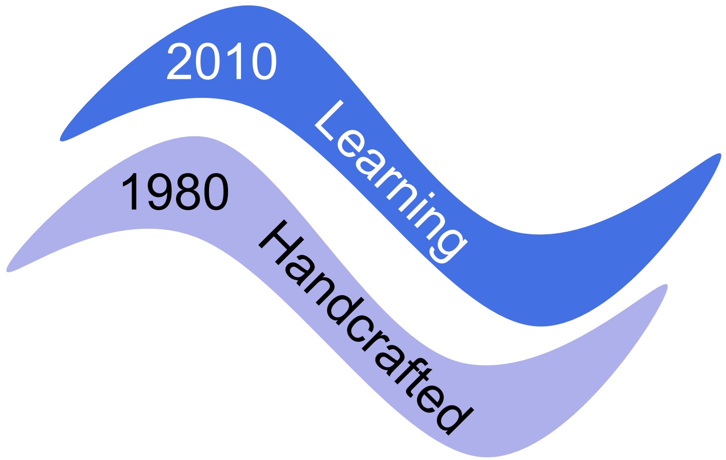
Many DNNs cannot distinguish the datasets

[Peharz, Vergari, Molina, Stelzner, Trapp, Kersting, Ghahramani UAI 2019]

Second wave of AI

Data are now ubiquitous; there is great value from understanding this data, building models and making predictions

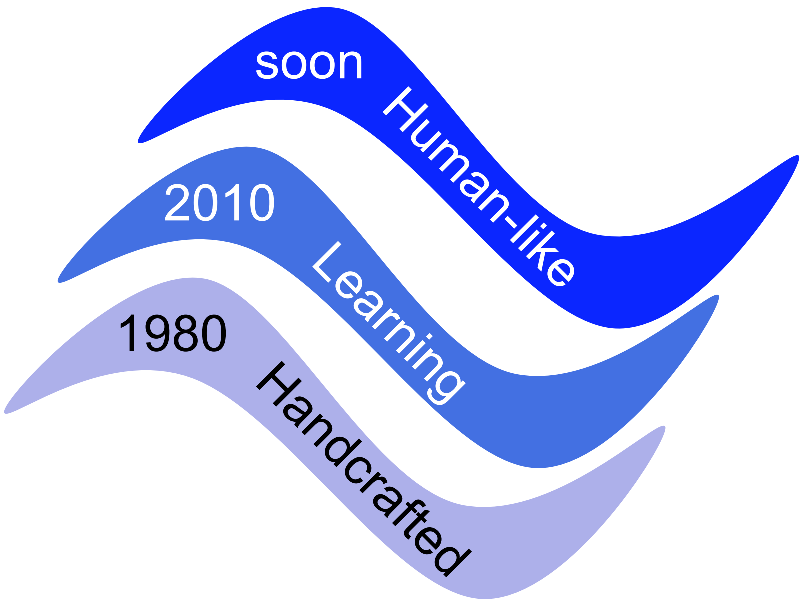
However, data is not everything



Third wave of AI

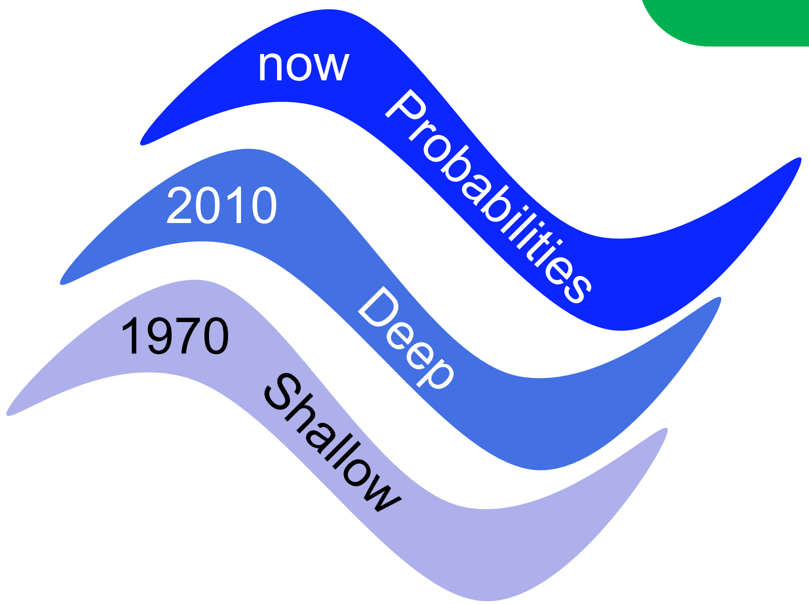
Data are now ubiquitous; there is great value from under-standing this data, building models and making predictions

However, data is not everything

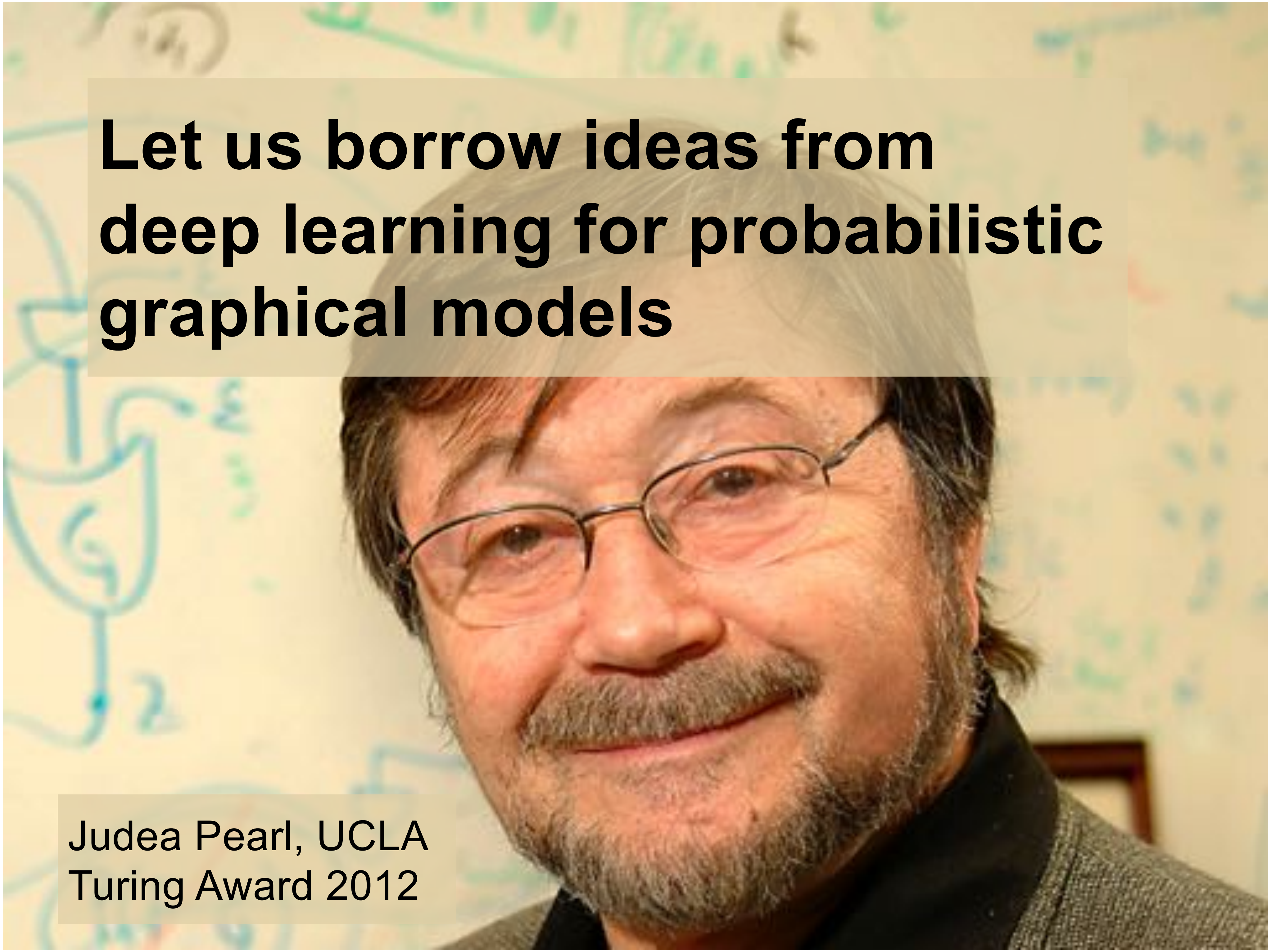


AI systems that can acquire human-like communication and reasoning capabilities, with the ability to recognise new situations and adapt to them.

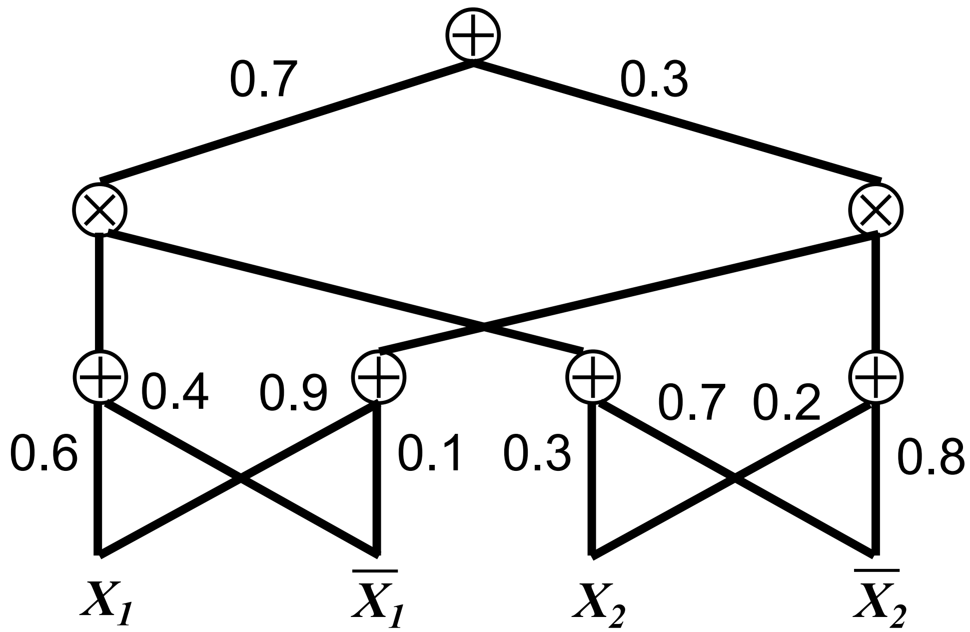
The third wave of deep learning



Getting deep systems that know when they do not know and, hence, recognise new situations


Let us borrow ideas from deep learning for probabilistic graphical models
Judea Pearl, UCLA
Turing Award 2012

Sum-Product Networks
a deep probabilistic learning framework



Computational graph (kind of TensorFlow graphs) that encodes how to compute probabilities

Inference is linear in size of network

Alternative Representation: Graphical Models as (Deep) Networks

X_1	X_2	$P(X)$
1	1	0.4
1	0	0.2
0	1	0.1
0	0	0.3

$$
\begin{aligned}
P(X) = {} & 0.4 \cdot \mathrm{I}[X_1{=}1] \cdot \mathrm{I}[X_2{=}1] \\
& + 0.2 \cdot \mathrm{I}[X_1{=}1] \cdot \mathrm{I}[X_2{=}0] \\
& + 0.1 \cdot \mathrm{I}[X_1{=}0] \cdot \mathrm{I}[X_2{=}1] \\
& + 0.3 \cdot \mathrm{I}[X_1{=}0] \cdot \mathrm{I}[X_2{=}0]
\end{aligned}
$$

Alternative Representation: Graphical Models as (Deep) Networks

X_1	X_2	$P(X)$
1	**1**	**0.4**
1	0	0.2
0	1	0.1
0	0	0.3

$$\begin{aligned} P(X) = &\ \mathbf{0.4 \cdot I[X_1{=}1] \cdot I[X_2{=}1]} \\ &+ 0.2 \cdot I[X_1{=}1] \cdot I[X_2{=}0] \\ &+ 0.1 \cdot I[X_1{=}0] \cdot I[X_2{=}1] \\ &+ 0.3 \cdot I[X_1{=}0] \cdot I[X_2{=}0] \end{aligned}$$

Shorthand using Indicators

X_1	X_2	$P(X)$
1	1	0.4
1	0	0.2
0	1	0.1
0	0	0.3

$$\begin{aligned} P(X) = {} & 0.4 \cdot X_1 \cdot X_2 \\ & + 0.2 \cdot X_1 \cdot \overline{X}_2 \\ & + 0.1 \cdot \overline{X}_1 \cdot X_2 \\ & + 0.3 \cdot \overline{X}_1 \cdot \overline{X}_2 \end{aligned}$$

Summing Out Variables

Let us say, we want to compute $P(X_1 = 1)$

X_1	X_2	$P(X)$
1	**1**	**0.4**
1	**0**	**0.2**
0	1	0.1
0	0	0.3

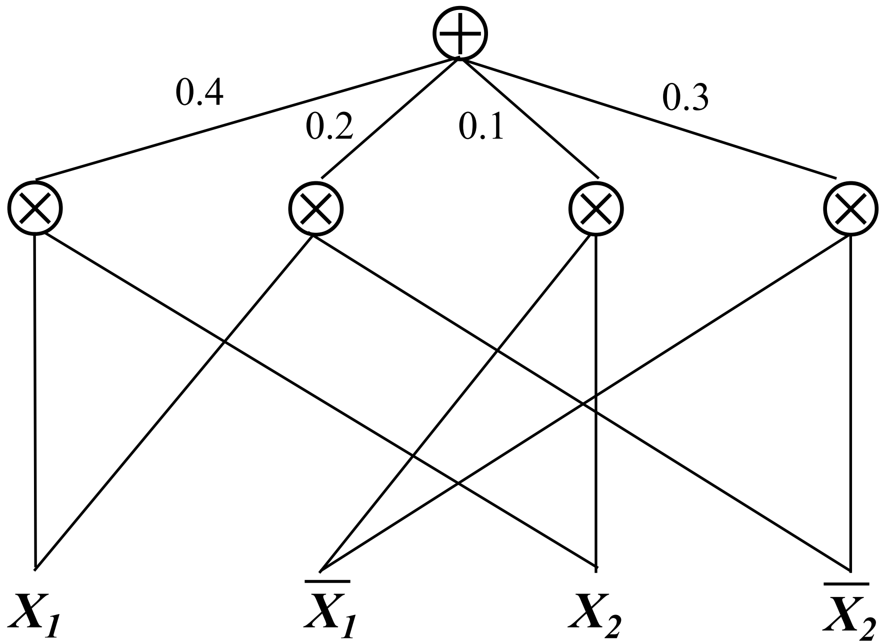
$$\begin{aligned} P(e) = &\ \mathbf{0.4 \cdot X_1 \cdot X_2} \\ &+ \mathbf{0.2 \cdot X_1 \cdot \overline{X}_2} \\ &+ 0.1 \cdot \overline{X}_1 \cdot X_2 \\ &+ 0.3 \cdot \overline{X}_1 \cdot \overline{X}_2 \end{aligned}$$

Set $X_1 = 1, \overline{X}_1 = 0, \boxed{X_2 = 1, \overline{X}_2 = 1}$

Easy: Set both indicators of X2 to 1

This can be represented as a computational graph

X_1	X_2	$P(X)$
1	1	0.4
1	0	0.2
0	1	0.1
0	0	0.3

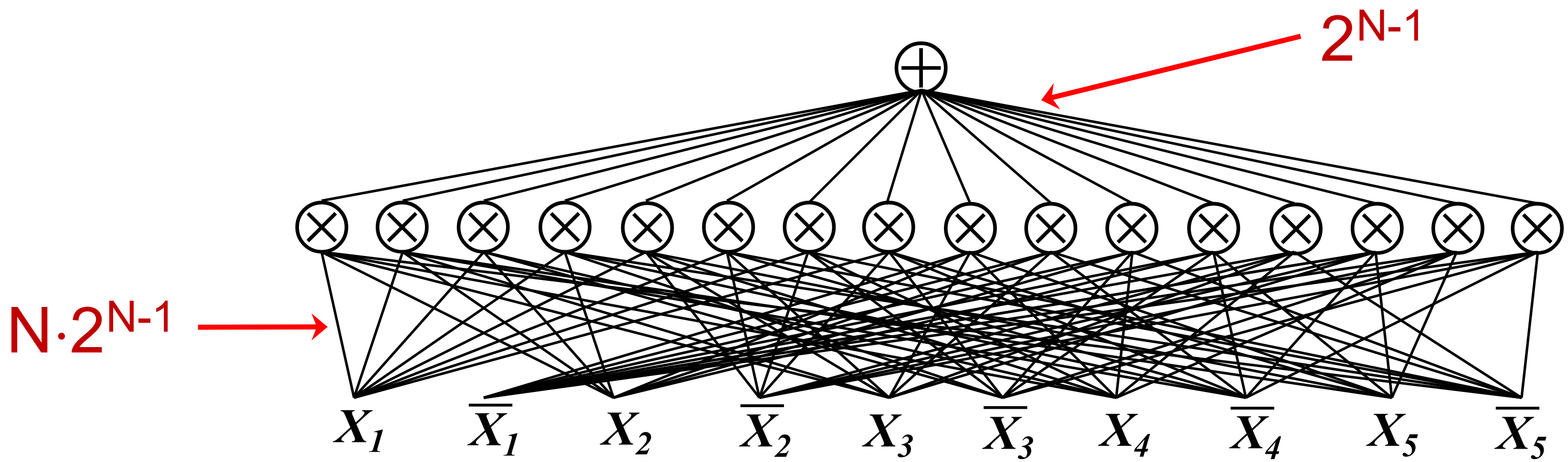


network polynomial

However, the network polynomial of a distribution might be exponentially large

Example: Parity

Uniform distribution over states with even number of 1's



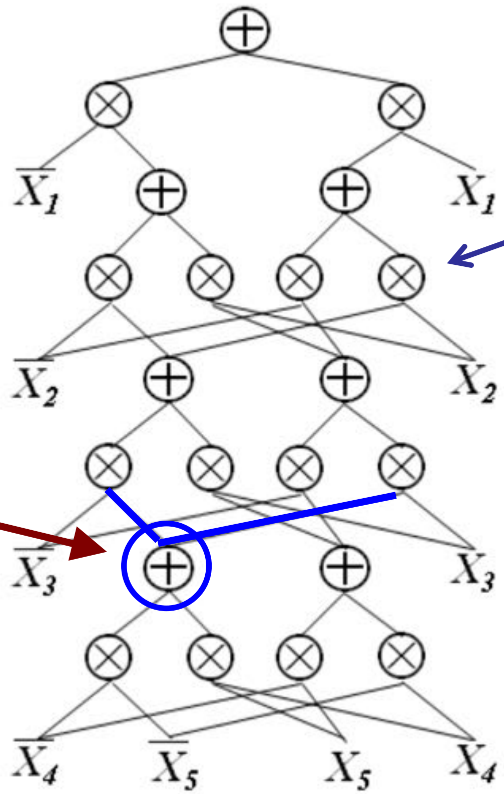
Make the computational graphs deep

Example: Parity

Uniform distribution over states with even number of 1's

Induce many hidden layers

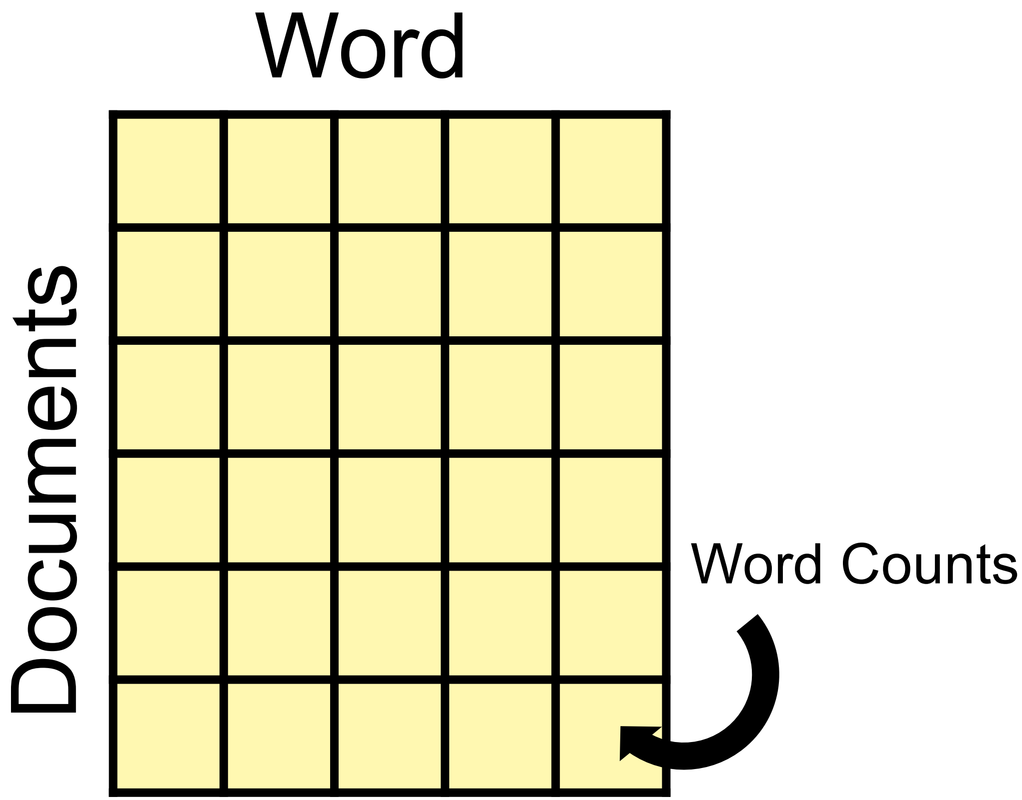
Reuse partial computation



[Poon, Domingos UAI'11; Molina, Natarajan, Kersting AAAI´17]

Principled approach to selecting (Tree-)SPNs

Testing independence using a (non-parametric) independency test



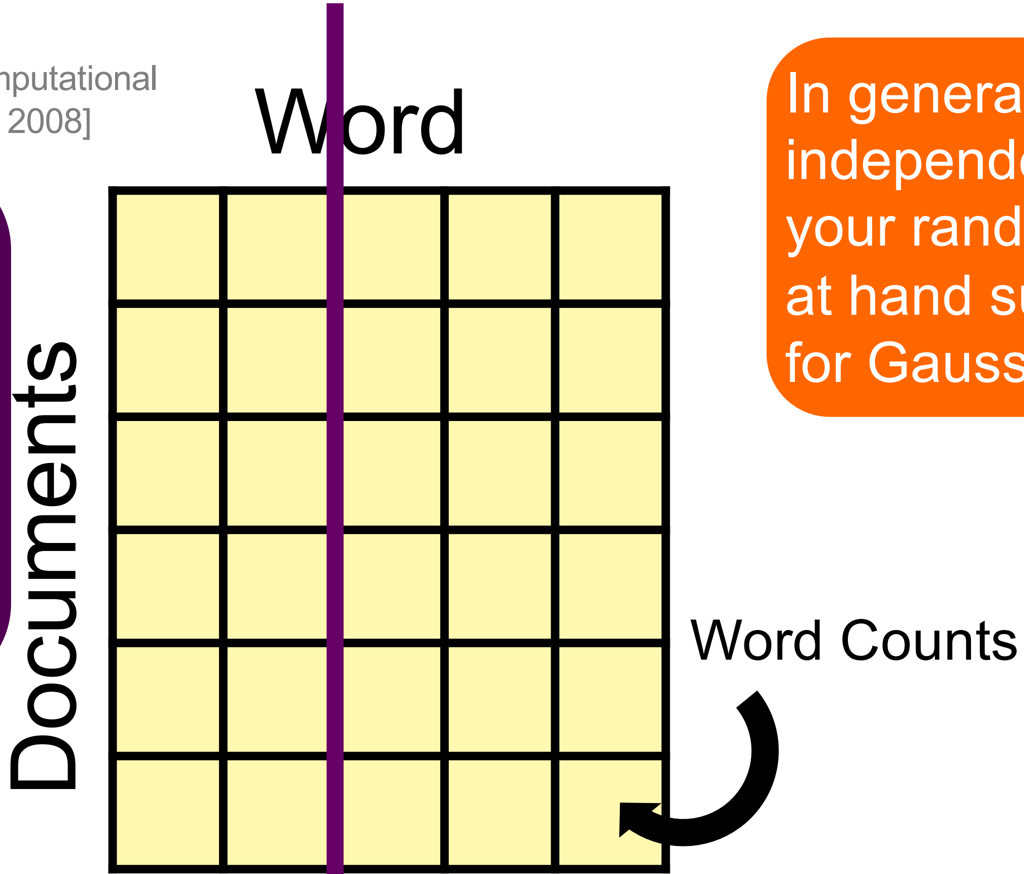
[Poon, Domingos UAI’11; Molina, Natarajan, Kersting AAAI´17]

Principled approach to selecting (Tree-)SPNs

Testing independence using a (non-parametric) independency test

[Zeileis, Hothorn, Hornik Journal of Computational And Graphical Statistics 17(2):492–514 2008]

E.g. for Poisson RVs: Learn Poisson model trees for P(x|V-x) and P(y|V-y). Check whether X resp. Y is significant in P(y|V-x) resp. P(x|V-y)



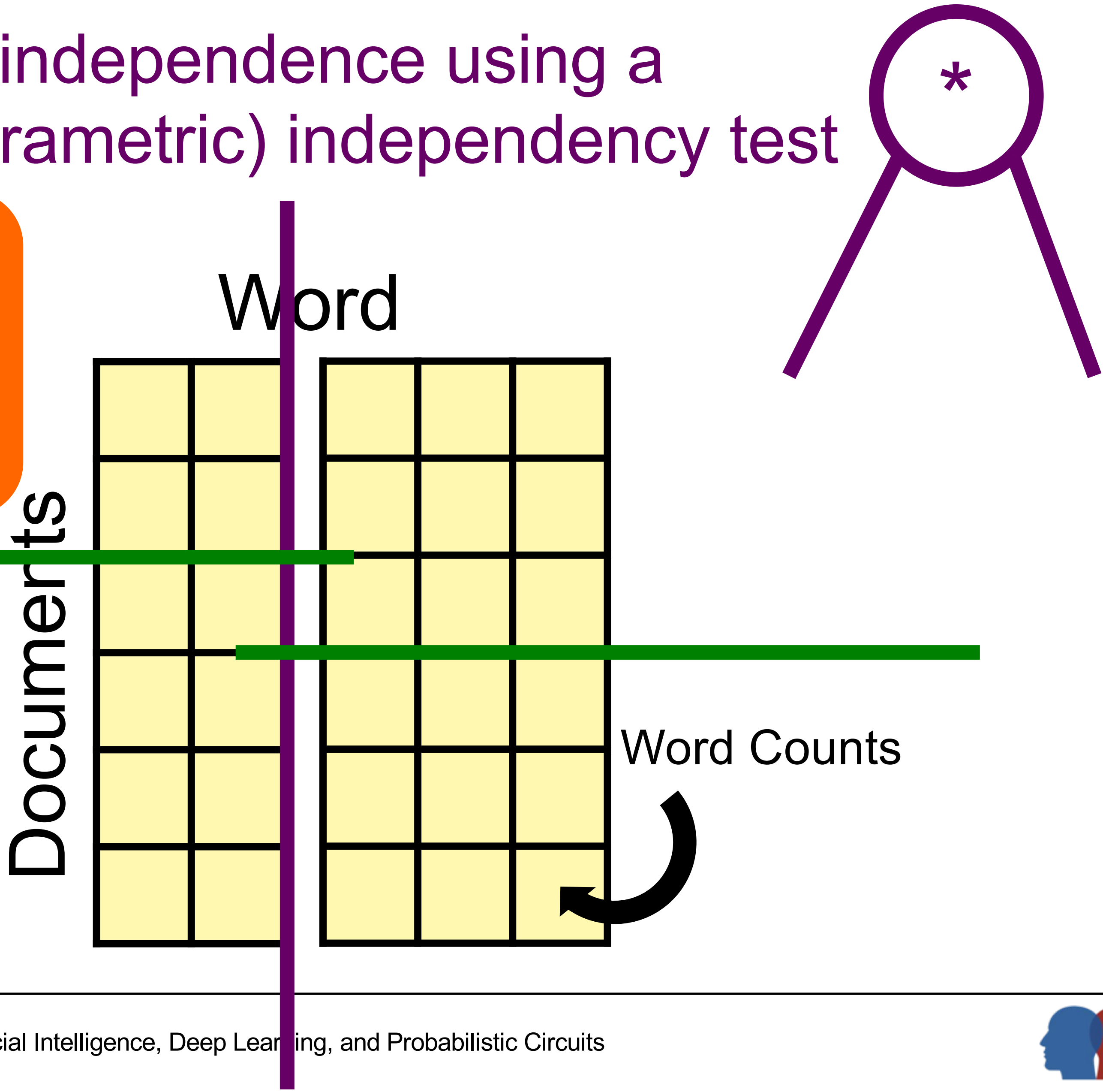
In general use the independency test for your random variables at hand such as g-test for Gaussians

Principled approach to selecting (Tree-)SPNs

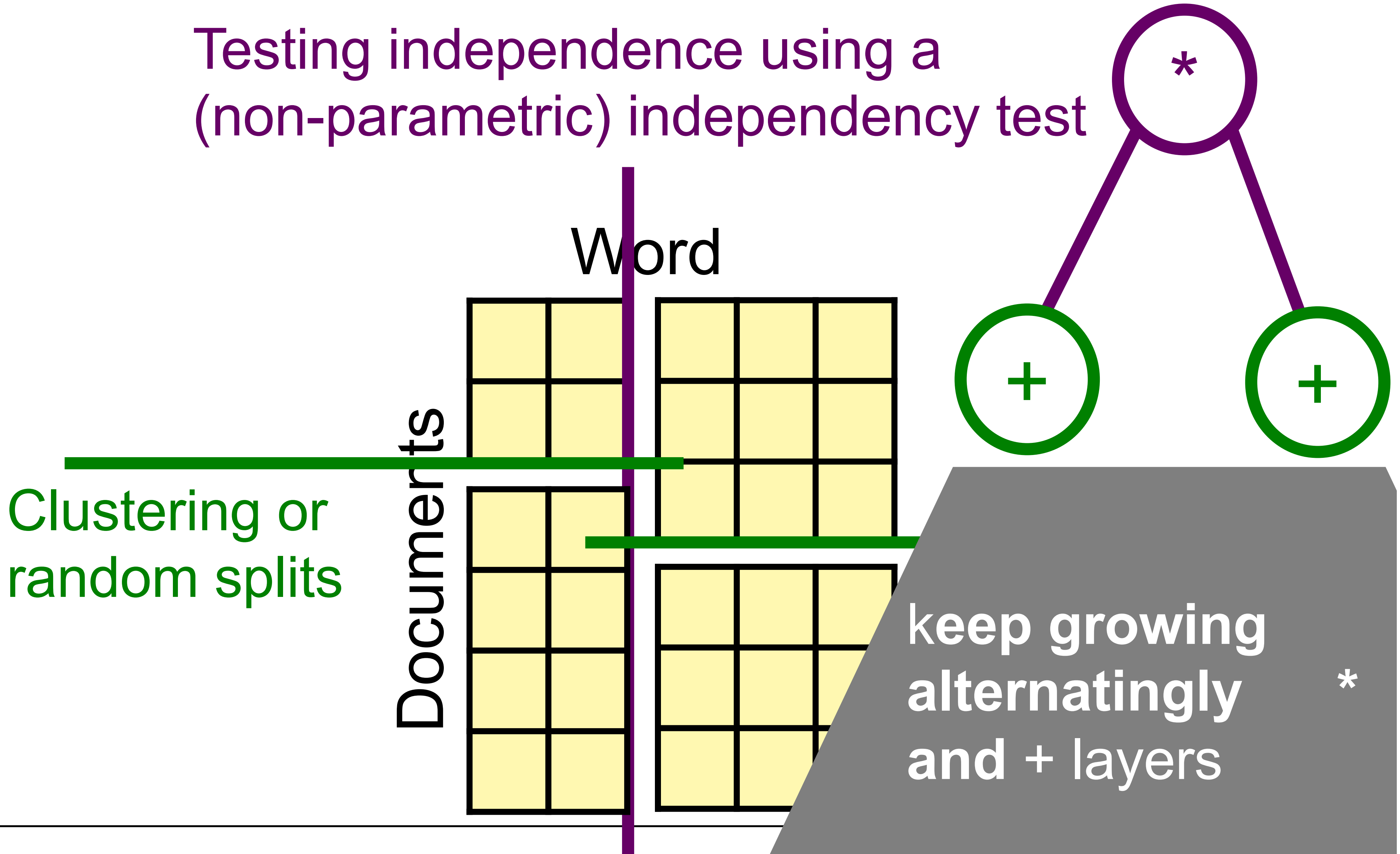
Testing independence using a (non-parametric) independency test

In general some clustering for your random variables at hand such as kMeans for Gaussians

Mixture of Poisson Dependency Networks or random splits



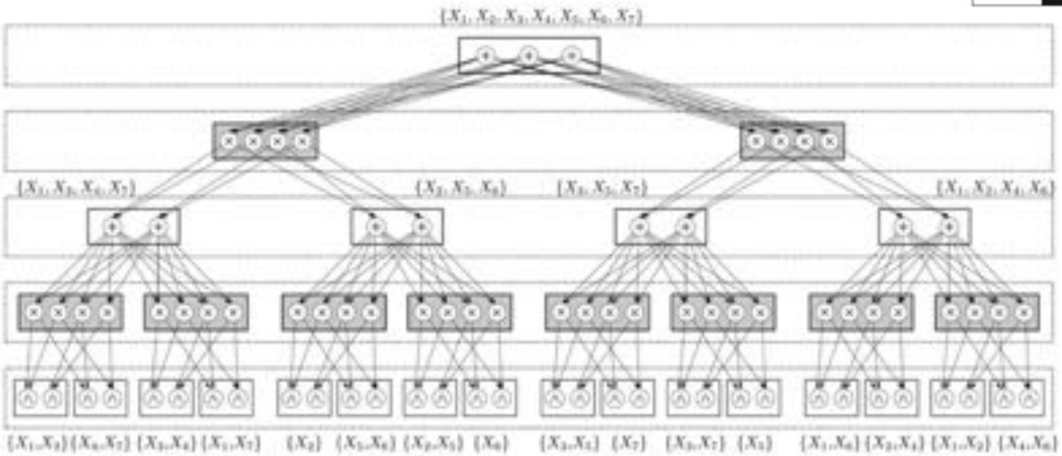
Principled approach to selecting (Tree-)SPNs



Random sum-product networks

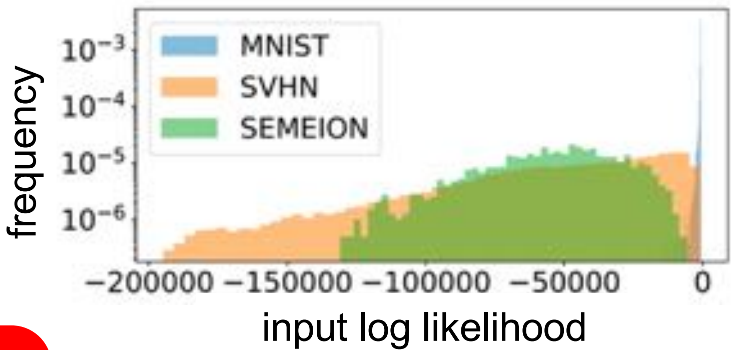
[Peharz, Vergari, Molina, Stelzner, Trapp, Kersting, Ghahramani UAI 2019]

Conference on Uncertainty in Artificial Intelligence
Tel Aviv, Israel
July 22 - 25, 2019
uai2019



Similar to Random Forests, build a random SPN structure. This can be done in an informed way or completely at random

		RAT-SPN	MLP	vMLP
Accuracy	MNIST	98.19 (8.5M)	98.32 (2.64M)	98.09 (5.28M)
	F-MNIST	89.52 (0.65M)	90.81 (9.28M)	89.81 (1.07M)
	20-NG	47.8 (0.37M)	49.05 (0.31M)	48.81 (0.16M)
Cross-Entropy	MNIST	0.0852 (17M)	0.0874 (0.82M)	0.0974 (0.22M)
	F-MNIST	0.3525 (0.65M)	0.2965 (0.82M)	0.325 (0.29M)
	20-NG	1.6954 (1.63M)	1.6180 (0.22M)	1.6263 (0.22M)



outliers
prototypes
outliers
prototypes

SPNs can have similar predictive performances as (simple) DNNs

SPNs can distinguish the datasets

SPNs know when they do not know by design

[Poon, Domingos UAI'11; Molina, Natarajan, Kersting AAAI'17; Vergari, Peharz, Di Mauro, Molina, Kersting, Esposito AAAI '18; Molina, Vergari, Di Mauro, Esposito, Natarajan, Kersting AAAI '18, Peharz et al. UAI 2019, Stelzner, Peharz, Kersting iCML 2019]

SPFlow: An Easy and Extensible Library for Sum-Product Networks

[Molina, Vergari, Stelzner, Peharz, Subramani, Poupart, Di Mauro, Kersting arXiv:1901.03704, 2019]

TECHNISCHE UNIVERSITÄT DARMSTADT · UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO · UNIVERSITY OF WATERLOO · CAML · MADESI · Max Planck Institute for Intelligent Systems · UNIVERSITY OF CAMBRIDGE · VECTOR INSTITUTE · DFG · Federal Ministry of Education and Research

195 commits | 2 branches | 0 releases | 6 contrib...

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

https://github.com/SPFlow/SPFlow

```
from spn.structure.leaves.parametric.Parametric import Categorical

from spn.structure.Base import Sum, Product

from spn.structure.base import assign_ids, rebuild_scopes_bottom_up


p0 = Product(children=[Categorical(p=[0.3, 0.7], scope=1), Categorical(p=[0.4, 0.6], scope=2)])
p1 = Product(children=[Categorical(p=[0.5, 0.5], scope=1), Categorical(p=[0.6, 0.4], scope=2)])
s1 = Sum(weights=[0.3, 0.7], children=[p0, p1])
p2 = Product(children=[Categorical(p=[0.2, 0.8], scope=0), s1])
p3 = Product(children=[Categorical(p=[0.2, 0.8], scope=0), Categorical(p=[0.3, 0.7], scope=1)])
p4 = Product(children=[p3, Categorical(p=[0.4, 0.6], scope=2)])
spn = Sum(weights=[0.4, 0.6], children=[p2, p4])

assign_ids(spn)
rebuild_scopes_bottom_up(spn)

return spn
```

Domain Specific Language, Inference, EM, and Model Selection as well as Compilation of SPNs into TF and PyTorch and also into flat, library-free code even suitable for running on devices: C/C++,GPU, FPGA

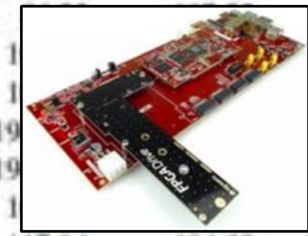
SPFlow, an open-source Python library providing a simple interface to inference, learning and manipulation routines for deep and tractable probabilistic models called Sum-Product Networks (SPNs). The library allows one to quickly create SPNs both from data and through a domain specific language (DSL). It efficiently implements several probabilistic inference

[Sommer, Oppermann, Molina, Binnig, Kersting, Koch ICDD 2018]

TABLE II

PERFORMANCE COMPARISON. BEST END-TO-END THROUGHPUTS (T), EXCLUDING THE CYCLE COUNTER MEASUREMENTS, ARE DENOTED BOLD.

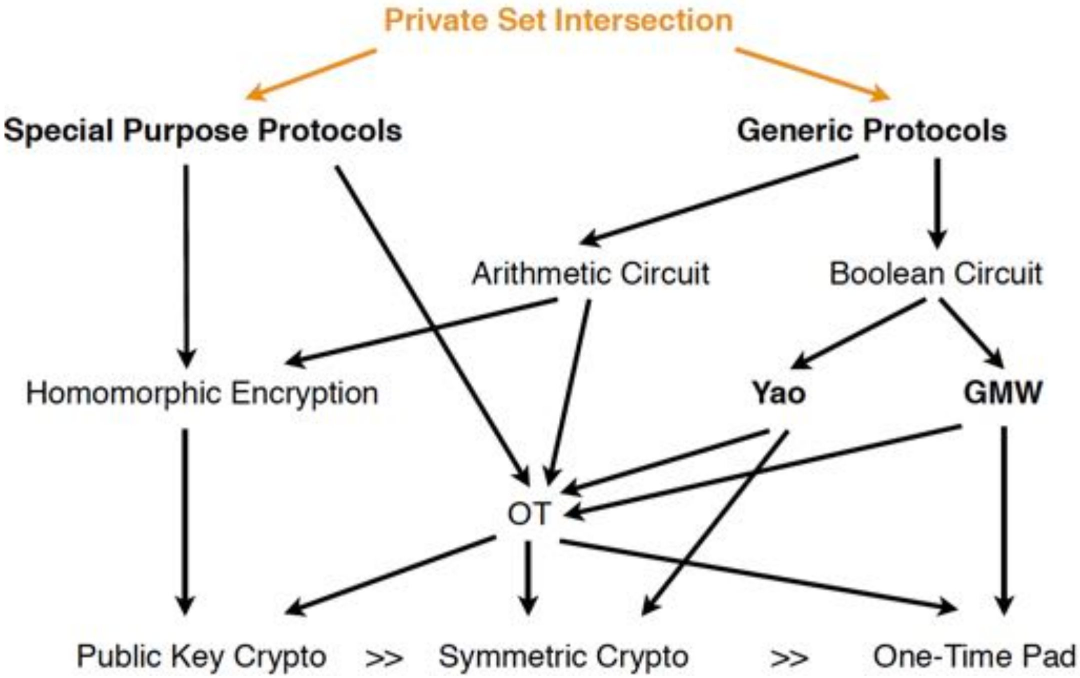
Dataset	Rows	CPU (μs)	T-CPU (rows/ μs)	CPUF (μs)	T-CPUF (rows/ μs)	GPU (μs)	T-GPU (rows/ μs)	FPGA Cycle Counter	FPGAC (μs)	T-FPGAC (rows/ μs)	FPGA (μs)	T-FPGA (rows/ μs)
Accidents	17009	2798.27	[illegible]	[illegible]	7.87	63090.94	0.27	17249	[illegible]	[illegible]	[illegible]	**24.44**
Audio	20000	4271.78	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	20317	[illegible]	[illegible]	[illegible]	**26.28**
Netflix	20000	4892.22	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	20322	[illegible]	[illegible]	[illegible]	**30.58**
MSNBC200	388434	15476.05	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	388900	[illegible]	[illegible]	[illegible]	**77.56**
MSNBC300	388434	10060.78	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	388810	[illegible]	[illegible]	[illegible]	**78.74**
NLTCS	21574	791.80	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	21904	[illegible]	[illegible]	[illegible]	**38.12**
Plants	23215	3621.71	6.41	3521.04	6.59	67004.41	0.35	23592	117.96	196.80	778.00	**29.84**
NIPS5	10000	25.11	**398.31**	26.37	379.23	8210.32	1.22	10236	51.18	195.39	337.30	[illegible]
NIPS10	10000	83.60	**119.61**	84.39	118.49	11550.82	0.87	10279	51.40	194.57	464.30	21.54
NIPS20	10000	191.30	52.27	182.73	**54.72**	18689.04	0.54	10285	51.43	194.46	543.60	18.40
NIPS30	10000	387.61	25.80	349.84	**28.58**	25355.93	0.39	10308	51.80	193.06	592.30	16.88
NIPS40	10000	551.64	18.13	471.26	**21.22**	30820.49	0.32	10306	51.53	194.06	632.20	15.82
NIPS50	10000	812.44	12.31	792.13	12.62	36355.60	0.28	10559	52.80	189.41	720.60	**13.88**
NIPS60	10000	1046.38	9.56	662.53	**15.09**	40778.36	0.25	12271	61.36	162.99	799.20	12.51
NIPS70	10000	1148.17	8.71	1134.80	8.81	46759.26	0.21	14022	70.11	142.63	858.60	**11.65**
NIPS80	10000	1556.99	6.42	1277.81	7.83	63217.99	0.16	14275	78.51	127.37	961.80	**10.40**



How do we do deep learning offshore?

MADESI

Federal Ministry of Education and Research

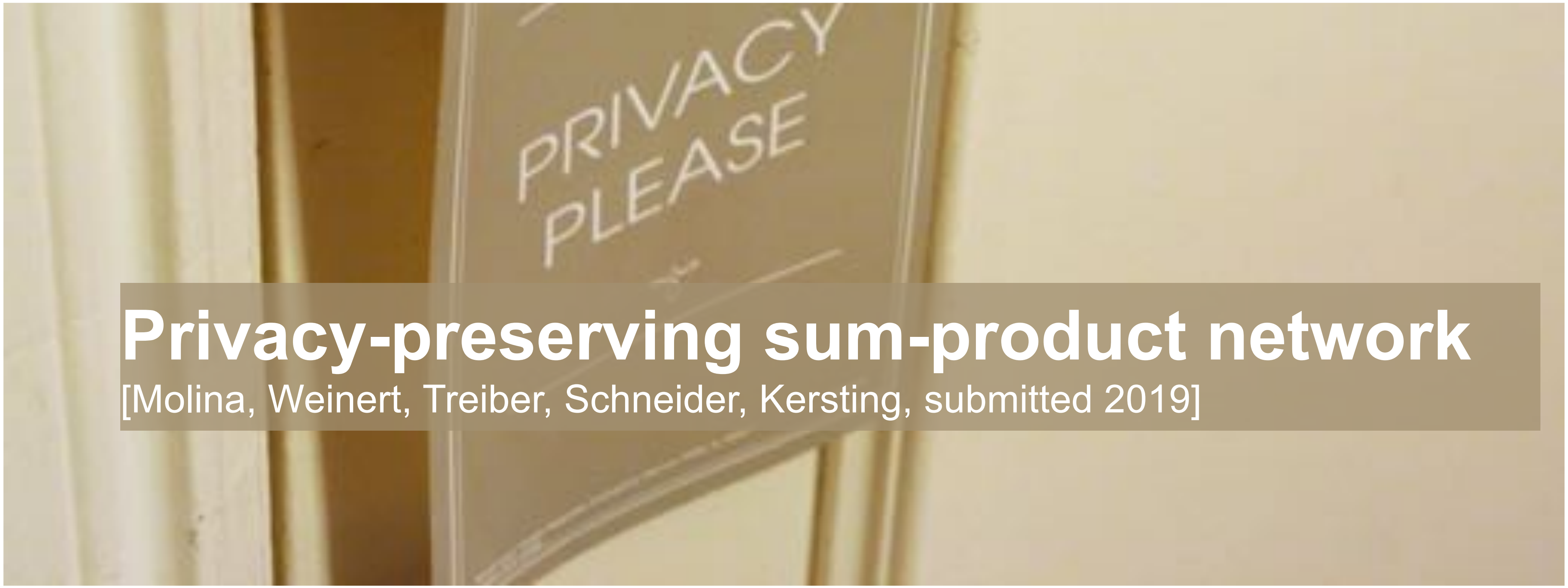


There are generic protocols to validate computations on authenticated data without knowledge of the secret key

DNA MSPN
Gates: 298208 Yao Bytes: 9542656 Depth: 615

DNA PSPN
Gates: 228272 Yao Bytes: 7304704 Depth: 589

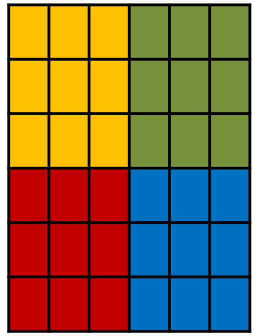
NIPS MSPN
Gates: 1001477 Yao Bytes: 32047264 Depth: 970

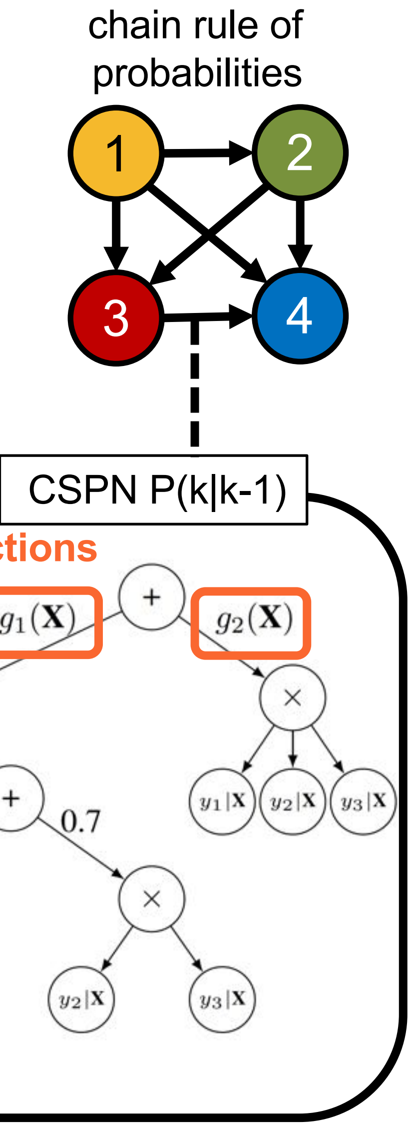


Privacy-preserving sum-product network

[Molina, Weinert, Treiber, Schneider, Kersting, submitted 2019]

Putting a little bit of structure into SPN models allows one to realize autoregressive deep models akin to PixelCNNs [van den Oord et al. NIPS 2016]





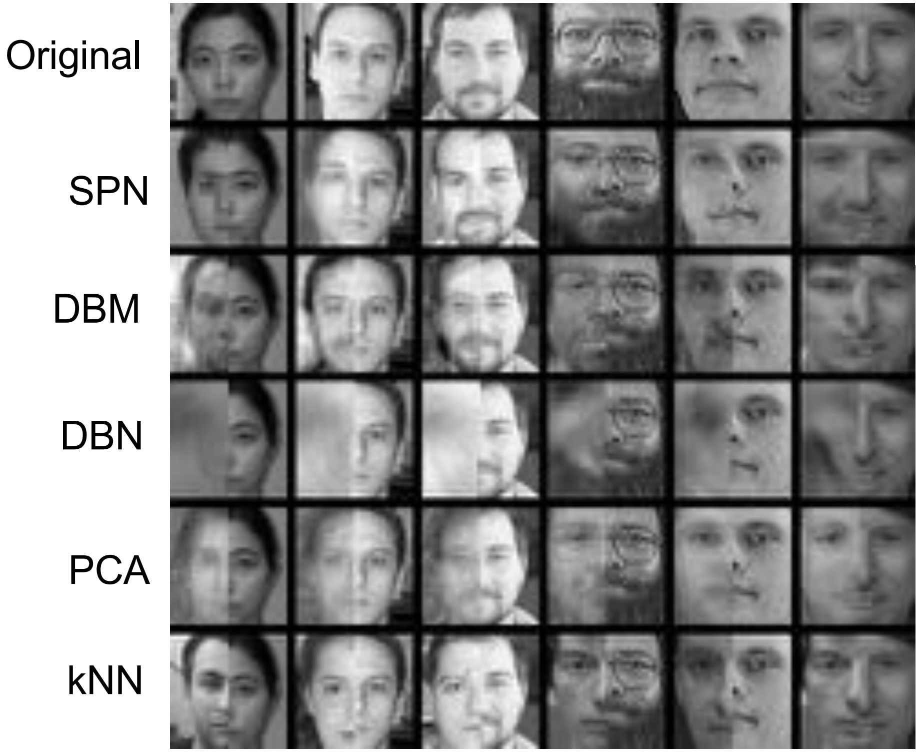
Learn Conditional SPN (CSPNs) by non-parametric conditional independence testing and conditional clustering [Zhang et al. UAI 2011; Lee, Honovar UAI 2017; He et al. ICDM 2017; Zhang et al. AAAI 2018; Runge AISTATS 2018] **encoded using gating functions**

Conditional SPNs

[Shao, Molina, Vergari, Peharz, Liebig, Kersting TPM@ICML 2019]

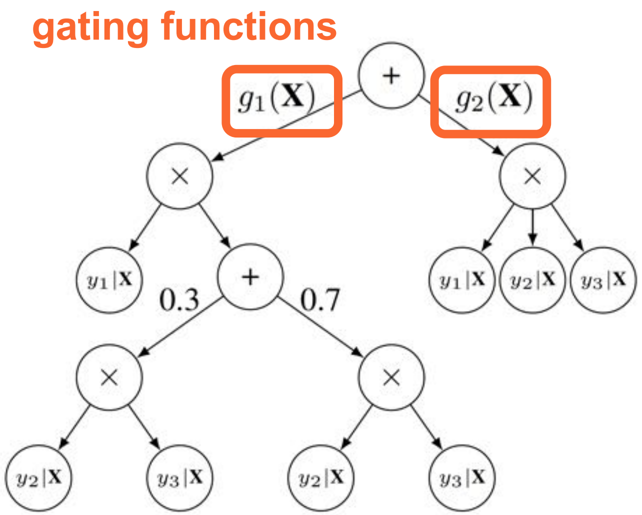
CAML DFG

[Poon, Domingos UAI’11]



Gating functions encoded as deep network

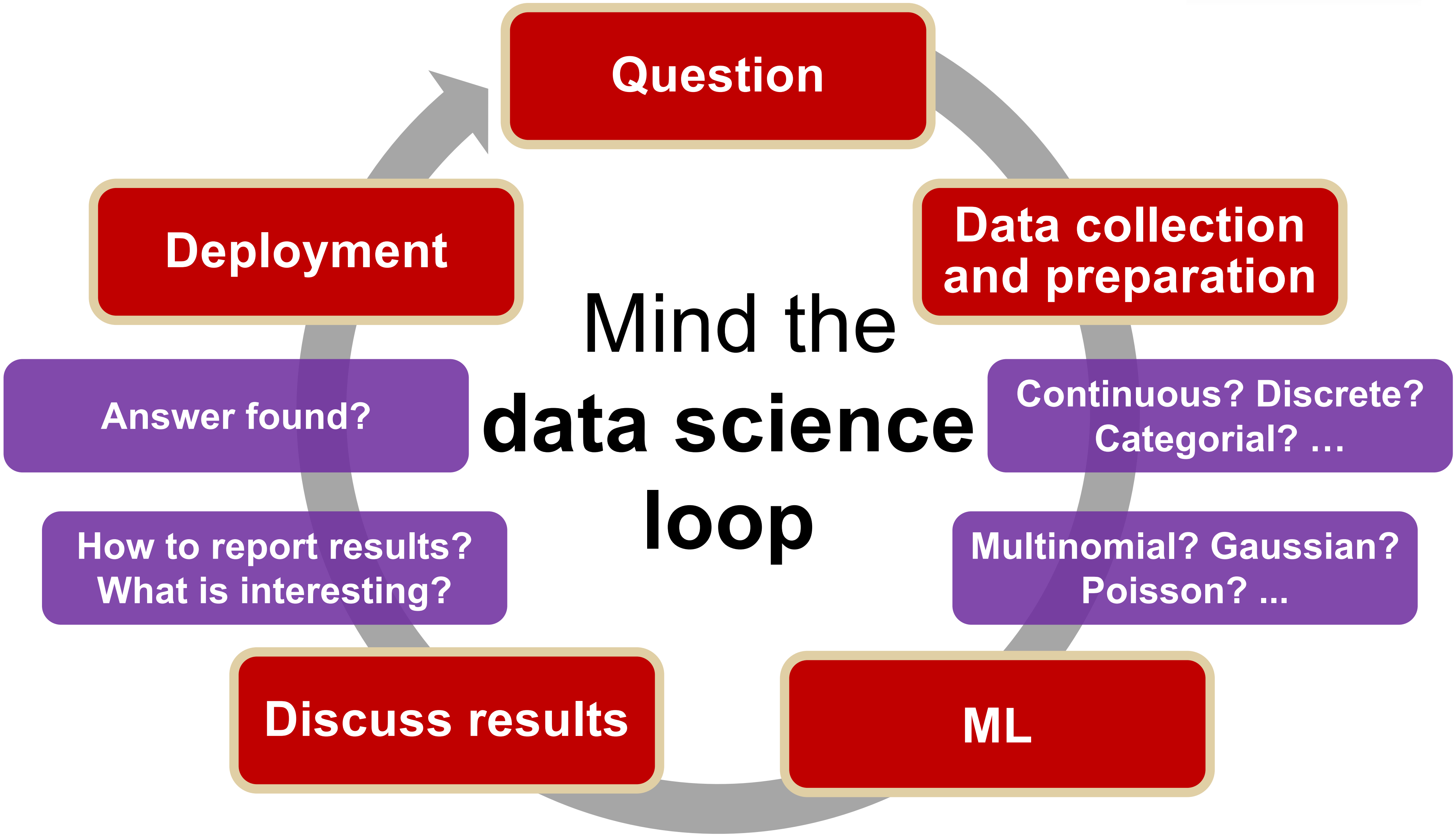
Learn Conditional SPN (CSPNs) by non-parametric conditional independence testing and conditional clustering [Zhang et al. UAI 2011; Lee, Honovar UAI 2017; He et al. ICDM 2017; Zhang et al. AAAI 2018; Runge AISTATS 2018] **encoded using gating functions**



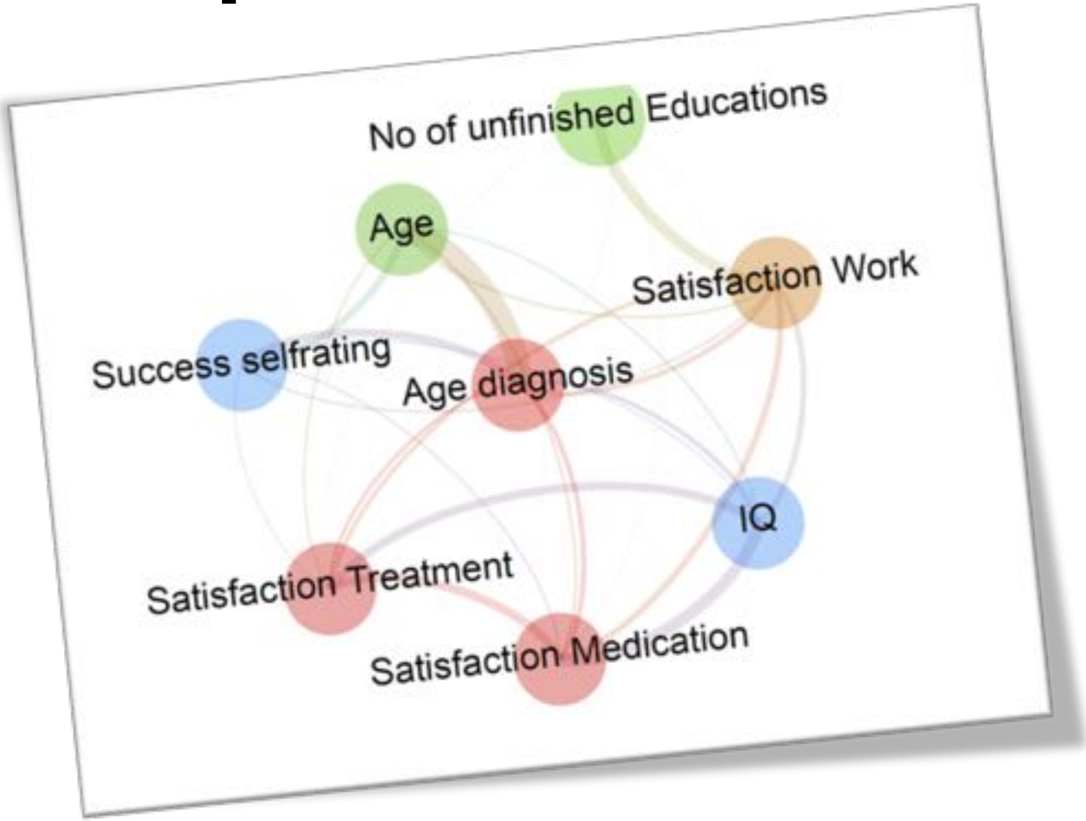
Conditional SPNs

[Shao, Molina, Vergari, Peharz, Liebig, Kersting TPM@ICML 2019]

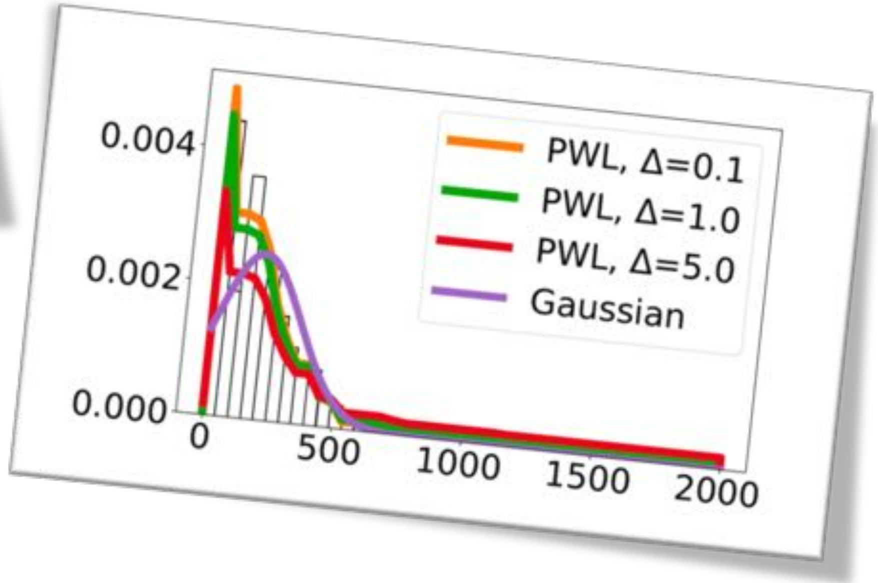
CAML DFG


Question
Data collection and preparation
Continuous? Discrete? Categorial? …
Multinomial? Gaussian? Poisson? ...
ML
Discuss results
How to report results? What is interesting?
Answer found?
Deployment
Mind the data science loop

Distribution-agnostic Deep Probabilistic Learning



Use nonparametric independency tests and piece-wise linear approximations



Distribution-agnostic Deep Probabilistic Learning

Use nonparametric independency tests and piece-wise linear approximations

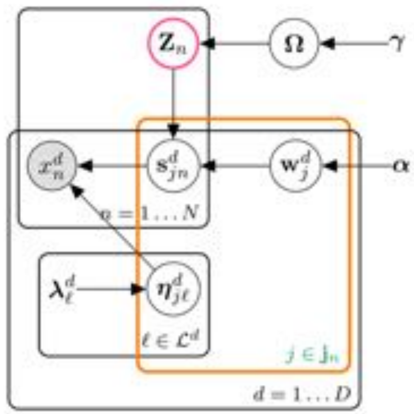
However, we have to provide the statistical types and do not gain insights into the parametric forms of the variables. **Are they Gaussians? Gammas? …**

[Vergari, Molina, Peharz, Ghahramani, Kersting, Valera AAAI 2019]

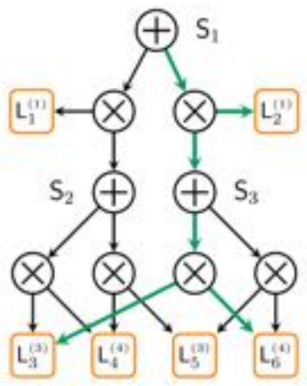
The Explorative Automatic Statistician

missing value

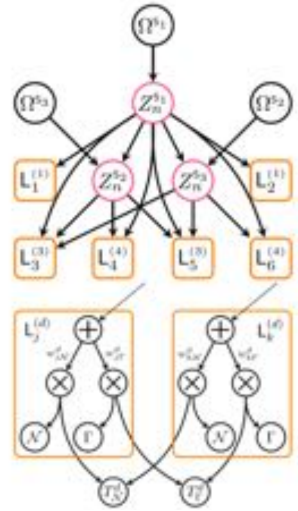
We can even automatically discovers the statistical types and parametric forms of the variables



Bayesian Type Discovery



Mixed Sum-Product Network



Automatic Statistician

That is, the machine understands the data with few expert input …

Voelcker, Molina, Neumann, Westermann, Kersting (2019): **DeepNotebooks: Deep Probabilistic Models Construct Python Notebooks for Reporting Datasets.** In Working Notes of the ECML PKDD 2019 Workshop on Automating Data Science (ADS)

Toggle Introduction | Toggle explanations | Toggle Code

Exploring the Titanic dataset

This report describes the dataset Titanic and contains general statistical information and an analysis on the influence different features and subgroups of the data have on each other. The first part of the report contains general statistical information about the dataset and an analysis of the variables and probability distributions.
The second part focusses on a subgroup analysis of the data. Different clusters identified by the network are analyzed and compared to give an insight into the structure of the data. Finally the influence different variables have on the predictive capabilities of the model are analyzes.
The whole report is generated by fitting a sum product network to the data and extracting all information from this model.

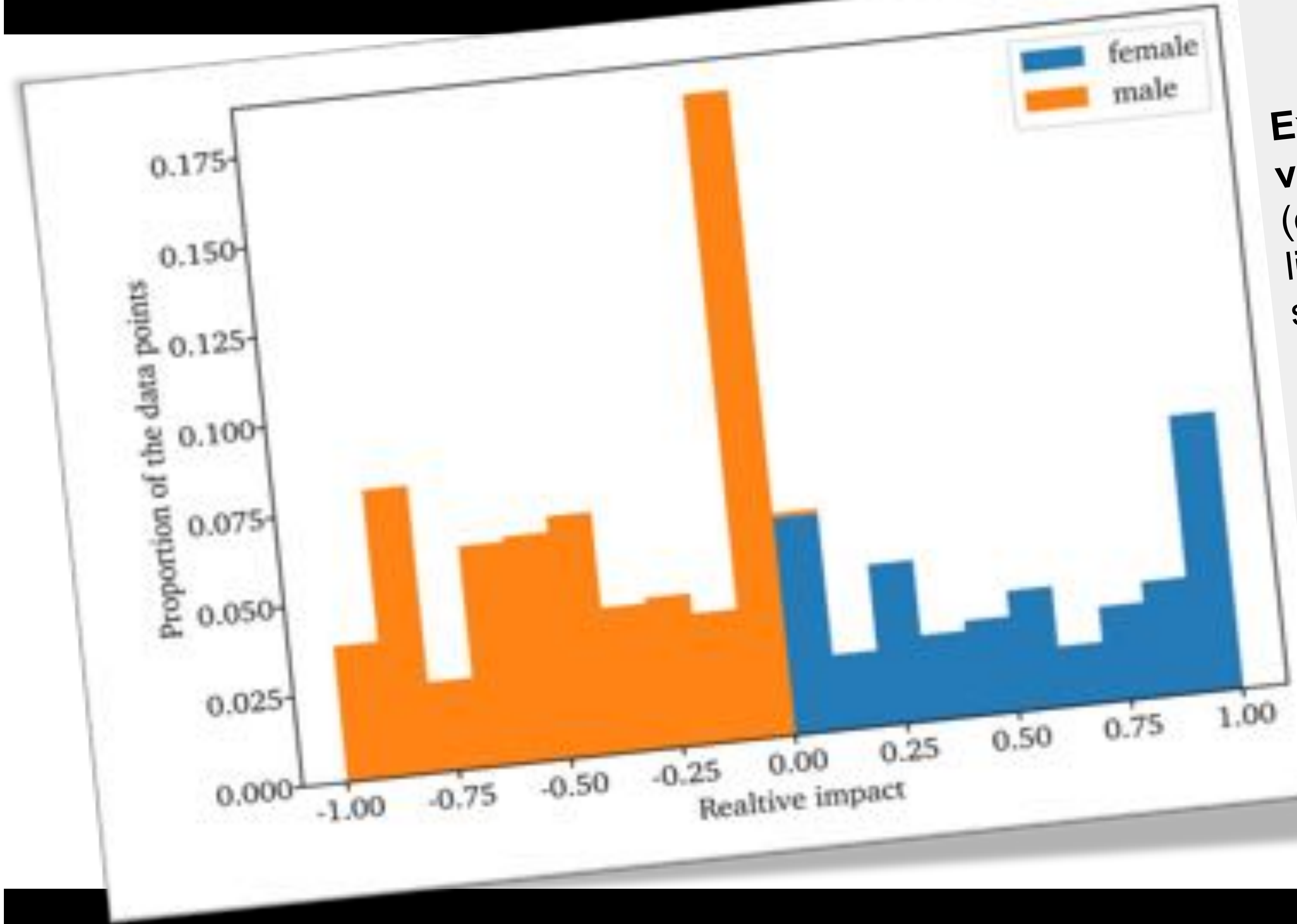
TECHNISCHE UNIVERSITÄT DARMSTADT

Report framework created @ TU Darmstadt

…and can compile data reports automatically

*[Baehrens, Schroeter, Harmeling, Kawanabe, Hansen, Müller JMLR 11:1803-1831, 2010]

That is, the machine understands the data with few expert input …



Explanation vector* (computable in linear time in the sizre of the SPN) **showing the impact of "gender" on the chances of survival for the Titanic dataset**

…and can compile data reports automatically

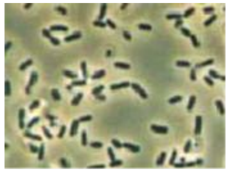
Unsupervised scene understanding

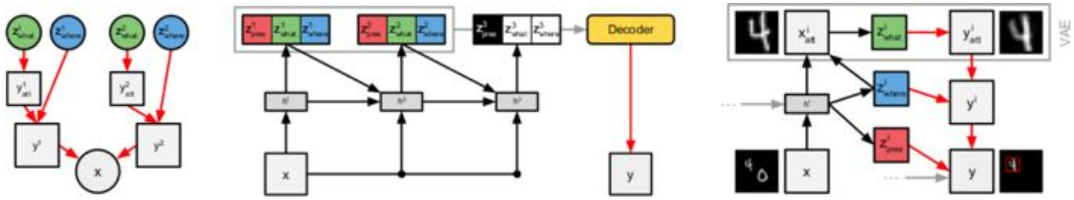
[Stelzner, Peharz, Kersting ICML 2019, Best Paper Award at TPM@ICML2019] https://github.com/stelzner/supair

UNIVERSITY OF CAMBRIDGE

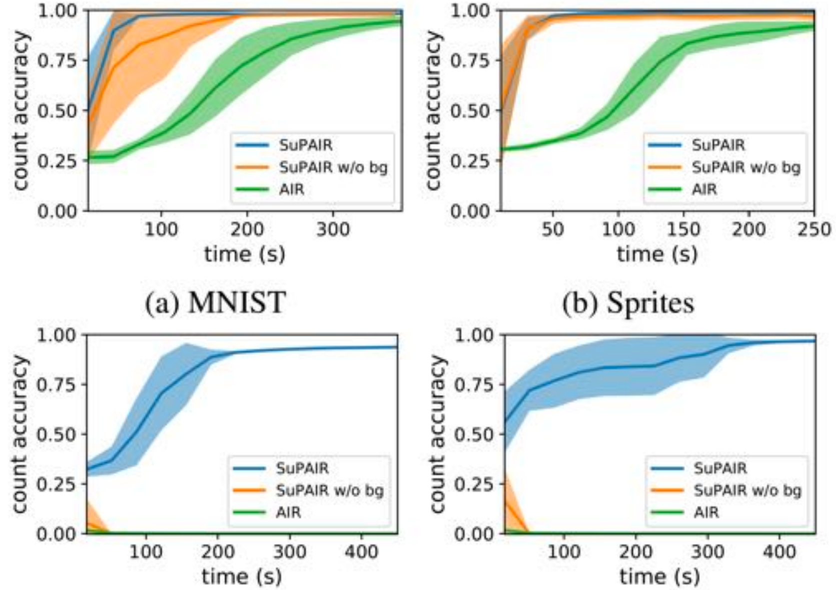
TECHNISCHE UNIVERSITÄT DARMSTADT

Consider e.g. unsupervised scene understanding using a generative model implemented in a neural fashion





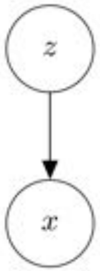
[Attend-Infer-Repeat (AIR) model, Hinton et al. NIPS 2016]



(a) MNIST (b) Sprites

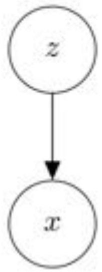
(c) Noisy MNIST (d) Grid MNIST

VAE



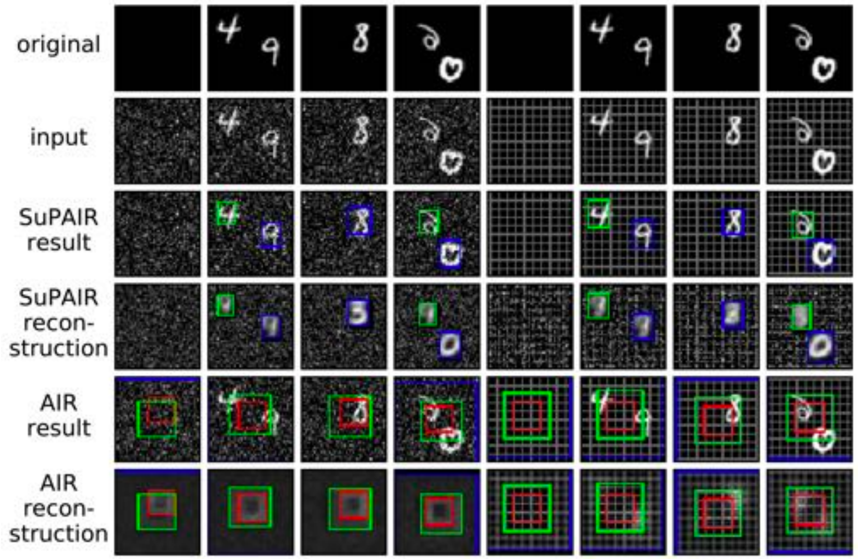
Replace VAE by SPN as object model

SPN



- infinite mixture model
- intractable density
- intractable posterior

- “large” but finite mixture model
- tractable density
- tractable marginals [Peharz et al., 2015]
- tractable posterior [Vergari et al., 2017]



Thanks to all students of the Research Training Group "Artificial Intelligence - Facts, Chances, Risks“ of the German National Academic Scholarship Foundation. Special thanks to **Maike Elisa Müller** and **Jannik Kossen** for taking the lead and to **Matthias Kleiner**, president of the Leibniz Association, for his preface

To summarize, DL is great. But AI is harder than you think. The third wave of AI requires integrative CS, from HPC, SoftEng and DBMS, over ML and AI, to computational CogSci

Illustration Nanina Föhr