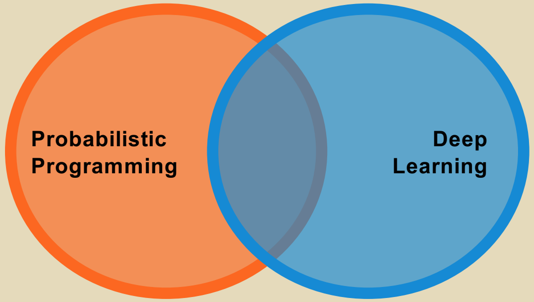
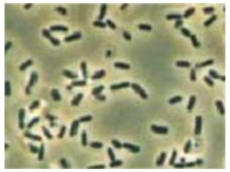
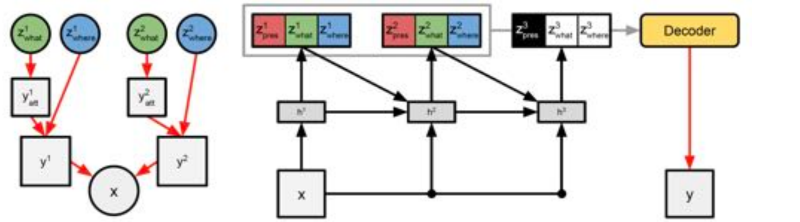
Deep machines that know when they do not know*

*Thanks for Pedro Domingos for making his slides publically available



Consider e.g. unsupervised scene understanding using a generative model



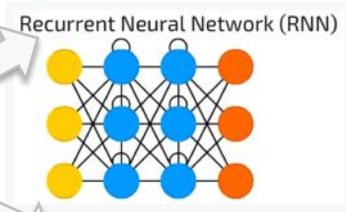


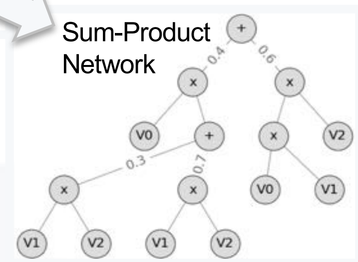
[Attend-Infer-Repeat (AIR) model, Hinton et al. NIPS 2016]

Probabilistic Programming: Easier modelling by programming generative models in a high-level, prob. language

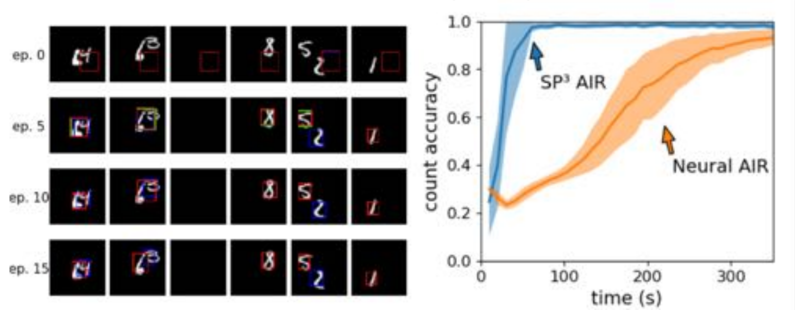
Deep Probabilistic Prog.: Modelling and inference might be hard, so use a deep neural network for it

```
def prior_step(t):
    # Sample object pose. This is a 3-dimensional vector representing x,y position and size.
    z_where = pyro.sample('z_where_{}'.format(t),
                          dist.normal,
                          z_where_prior_mu, z_where_prior_sigma)

    # Sample object code. This is a 50-dimensional vector.
    z_what = pyro.sample('z_what_{}'.format(t),
                         dist.normal,
                         z_what_prior_mu, z_what_prior_sigma)

    y_att = decode(z_what)                # Map latent code to pixel space using the neural ne
```





Sum-Product Probabilistic Programming: Making machine learning and data science easier [Stelzner, Molina, Peharz, Vergari, Trapp, Valera, Ghahramani, Kersting ProgProb 2018]



Use deep probabilistic models that feature tractable, deterministic inference

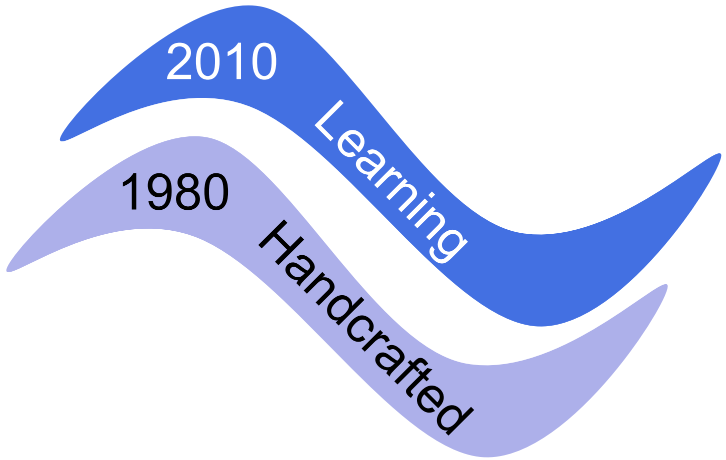
```
from spn.structure.leaves.parametric.Parametric import Categorical

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
```

Kristian Kersting

Third wave of AI

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

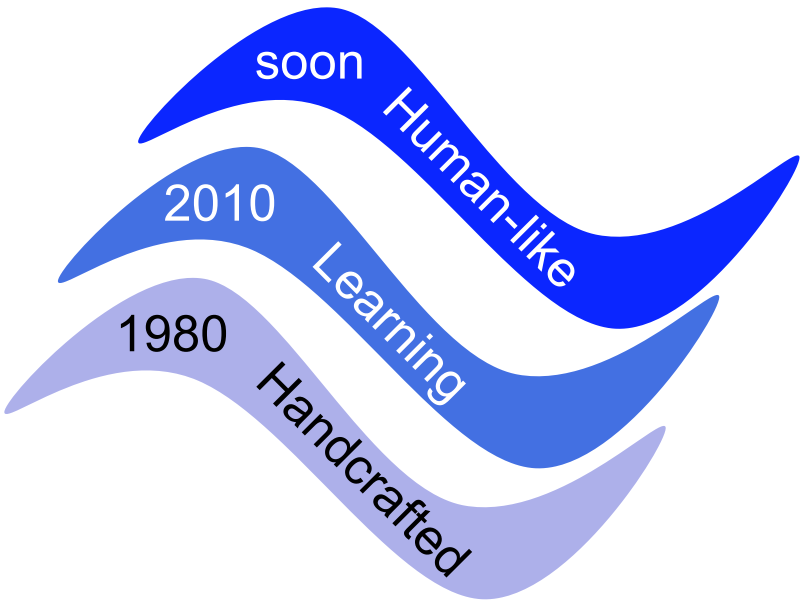
However, data is not everything



Third wave of AI

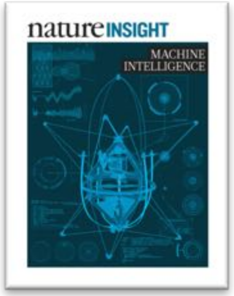
Data are now ubiquitous; there is great value from understanding this data, building models and making predictions

However, data is not everything



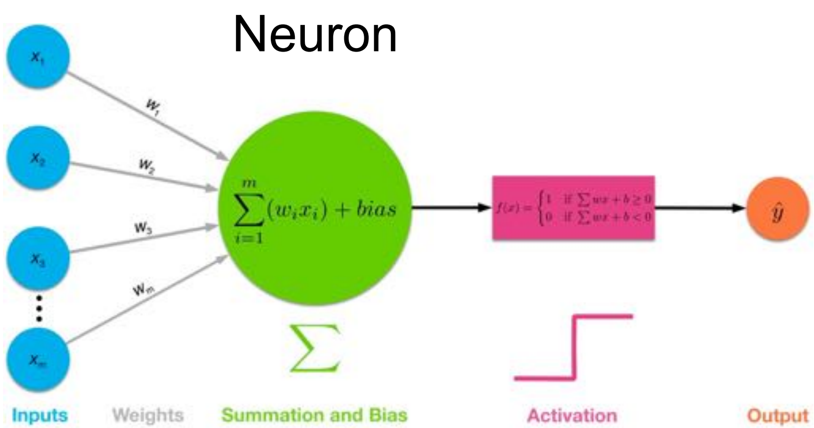
AI systems that can acquire human-like communication and reasoning capabilities, with the ability to recognise new situations and adapt to them.

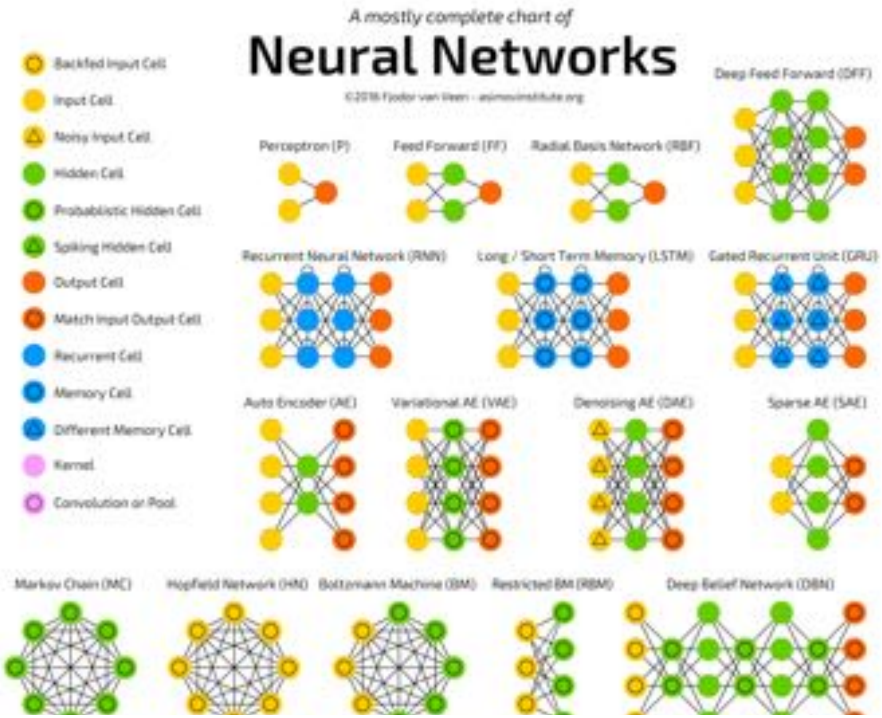
Deep Neural Networks



Potentially much more powerful than shallow architectures, represent computations

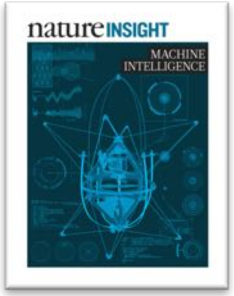
[LeCun, Bengio, Hinton Nature 521, 436–444, 2015]





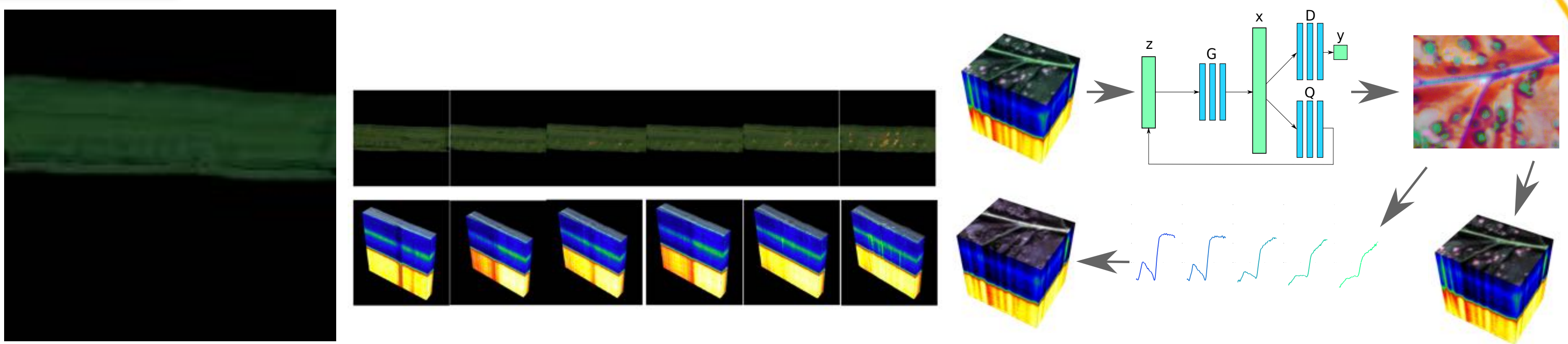
Differentiable Programming

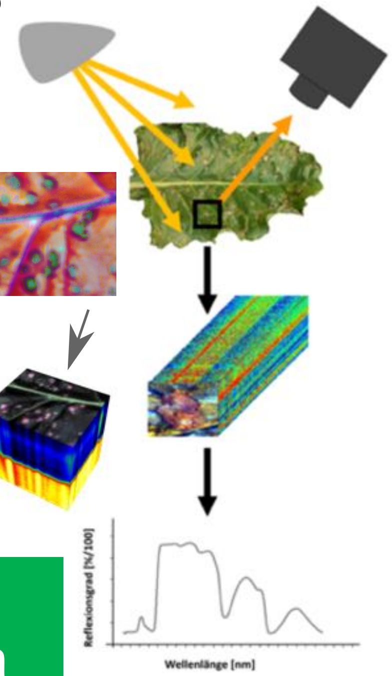
Deep Neural Networks



Potentially much more powerful than shallow architectures, represent computations

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





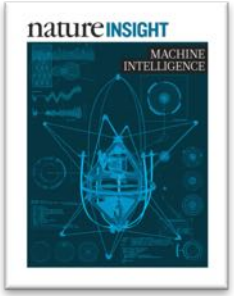
They "develop intuition" about complicated biological processes and generate scientific data

[Schramowski, Brugger, Mahlein, Kersting 2019]

DePhenSe

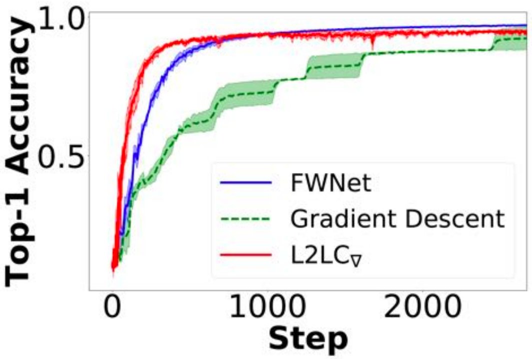
Bundesanstalt für Landwirtschaft und Ernährung

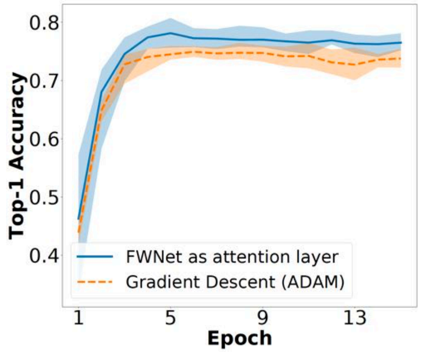
Deep Neural Networks

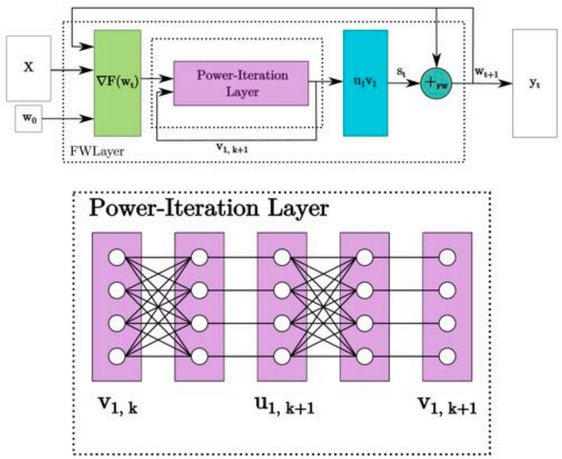


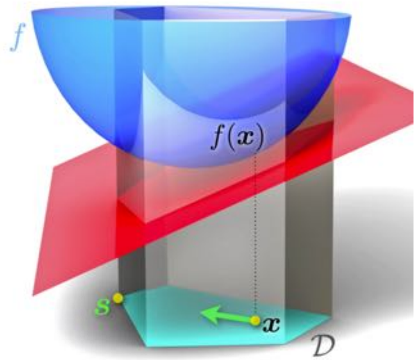
Potentially much more powerful than shallow architectures, represent computations

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







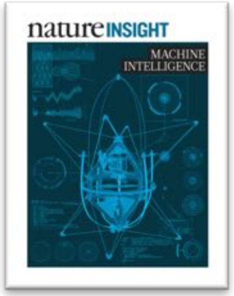


They "invent" constrained optimizers

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

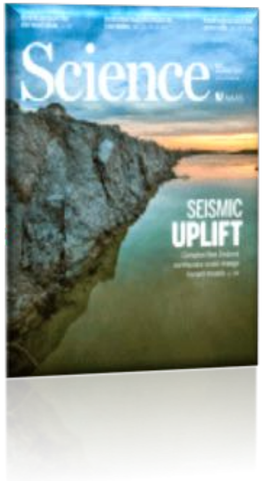
Bundesanstalt für Landwirtschaft und Ernährung

Deep Neural Networks



Potentially much more powerful than shallow architectures, represent computations

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



SHARE

1.02k

0

REPORTS | PSYCHOLOGY

Semantics derived automatically from language corpora contain human-like biases

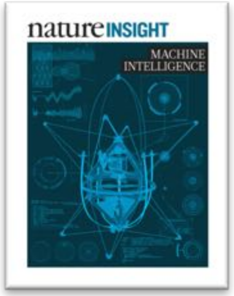
Aylin Caliskan[1,*], Joanna J. Bryson[1,2,*], Arvind Narayanan[1,*]

+ See all authors and affiliations

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

They "capture" stereotypes from human language

Deep Neural Networks

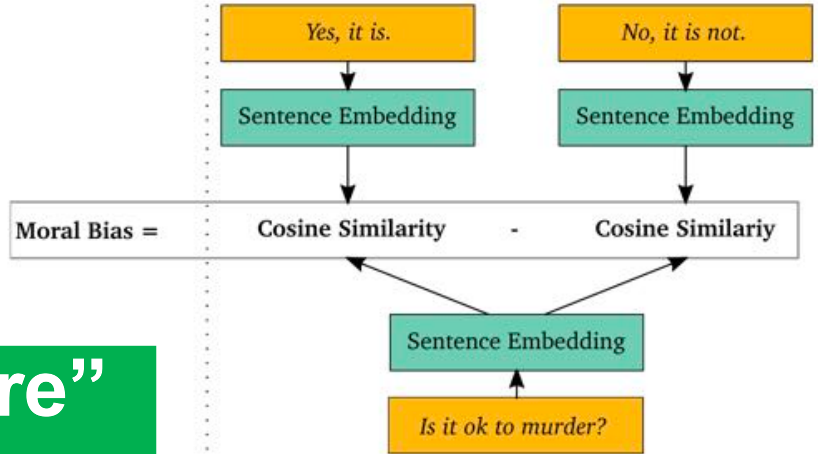


Potentially much more powerful than shallow architectures, represent computations

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

The Moral Choice Machine

Dos	WEAT	Bias	**Don'ts**	WEAT	Bias
smile	0.116	0.348	rot	-0.099	-1.118
sightsee	0.090	0.281	negative	-0.101	-0.763
cheer	0.094	0.277	harm	-0.110	-0.730
celebrate	0.114	0.264	damage	-0.105	-0.664
picnic	0.093	0.260	slander	-0.108	-0.600
snuggle	0.108	0.238	slur	-0.109	-0.569



But lucky they also "capture" our moral choices

[Jentzsch, Schramowski, Rothkopf, Kersting AIES 2019]

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

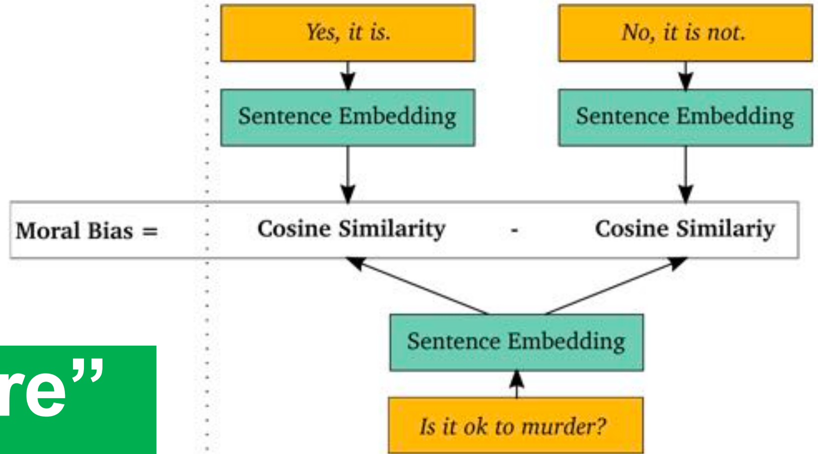
Video 05:10 Min.

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

hauptsache kultur | 14.03.19, 22:45 Uhr

The Moral Choice Machine

Dos	WEAT	Bias	Don'ts	WEAT	Bias
smile	0.116	0.348	rot	-0.099	-1.118
sightsee	0.090	0.281	negative	-0.101	-0.763
cheer	0.094	0.277	harm	-0.110	-0.730
celebrate	0.114	0.264	damage	-0.105	-0.664
picnic	0.093	0.260	slander	-0.108	-0.600
snuggle	0.108	0.238	slur	-0.109	-0.569

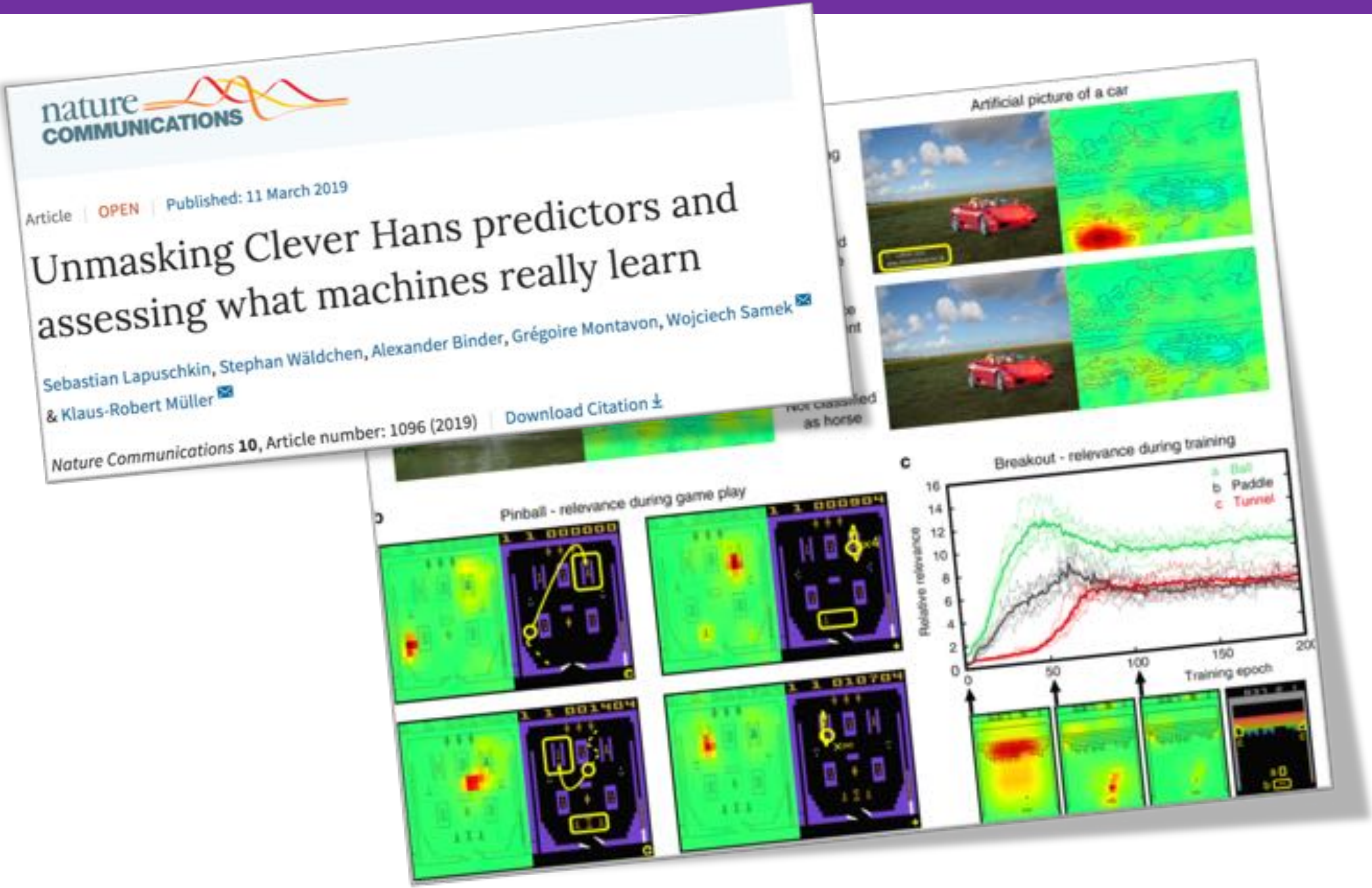


But lucky they also "capture" our moral choices

[Jentzsch, Schramowski, Rothkopf, Kersting AIES 2019]

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

Can we trust deep neural networks?



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

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

MNIST

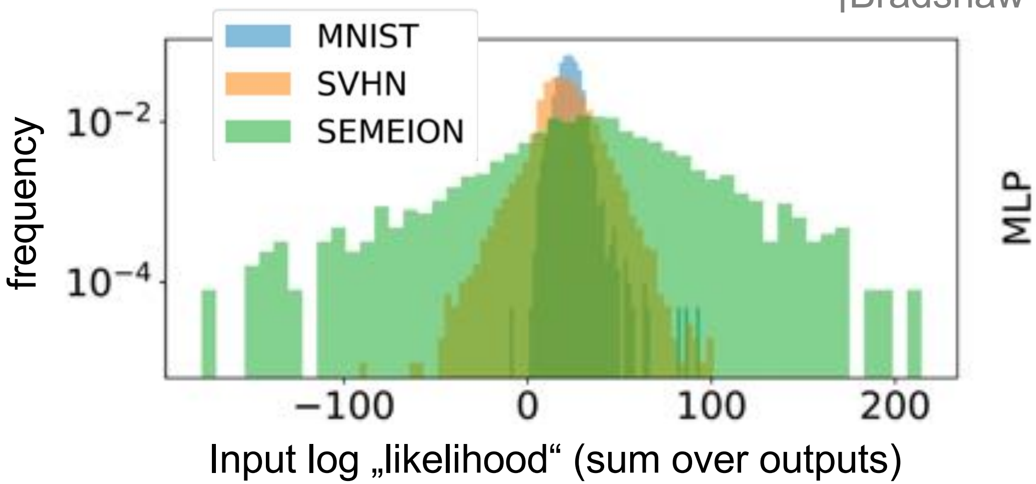
SVHN

SEMEION

Train & Evaluate

Transfer Testing

[Bradshaw et al. arXiv:1707.02476 2017]

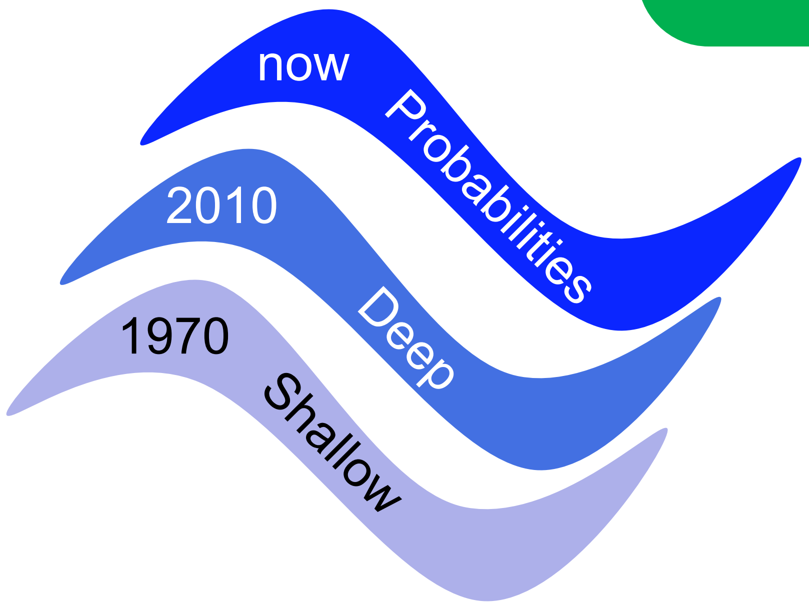


Many DNNs cannot distinguish the datasets

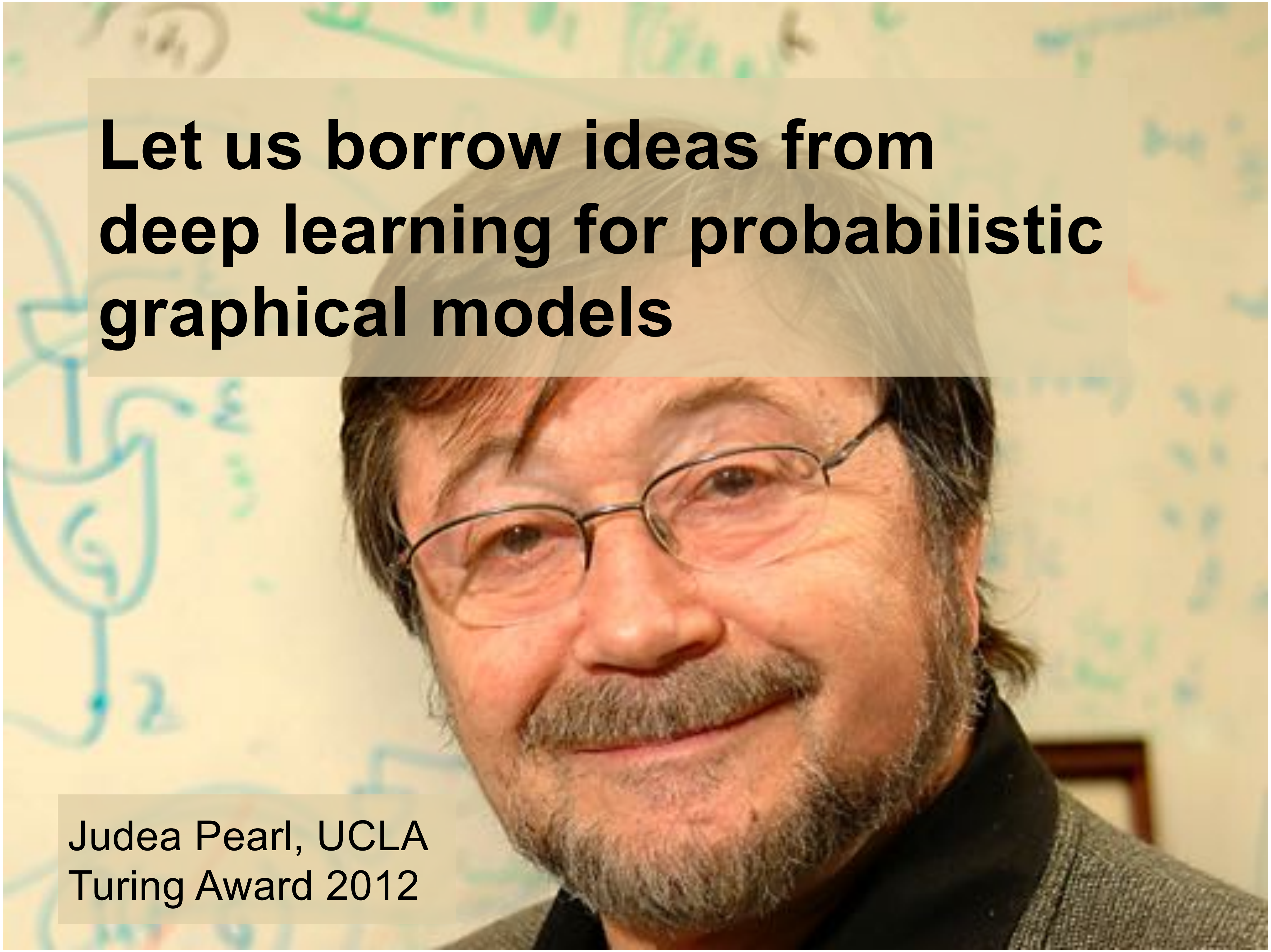
[Peharz, Vergari, Molina, Stelzner, Trapp, Kersting, Ghahramani UAI 2019]

The third wave of deep learning

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

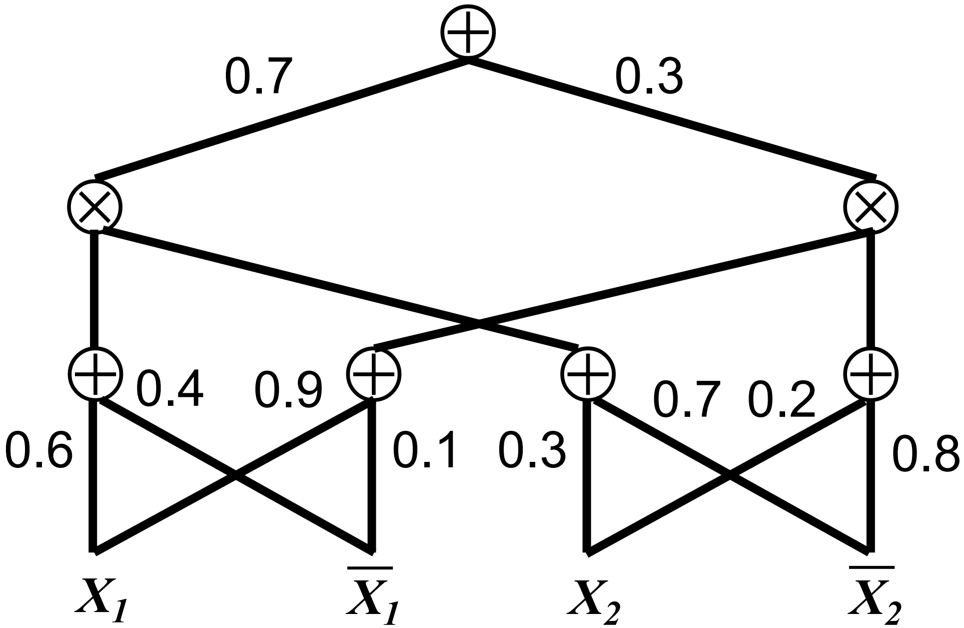


Let us borrow ideas from deep learning for probabilistic graphical models



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

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

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

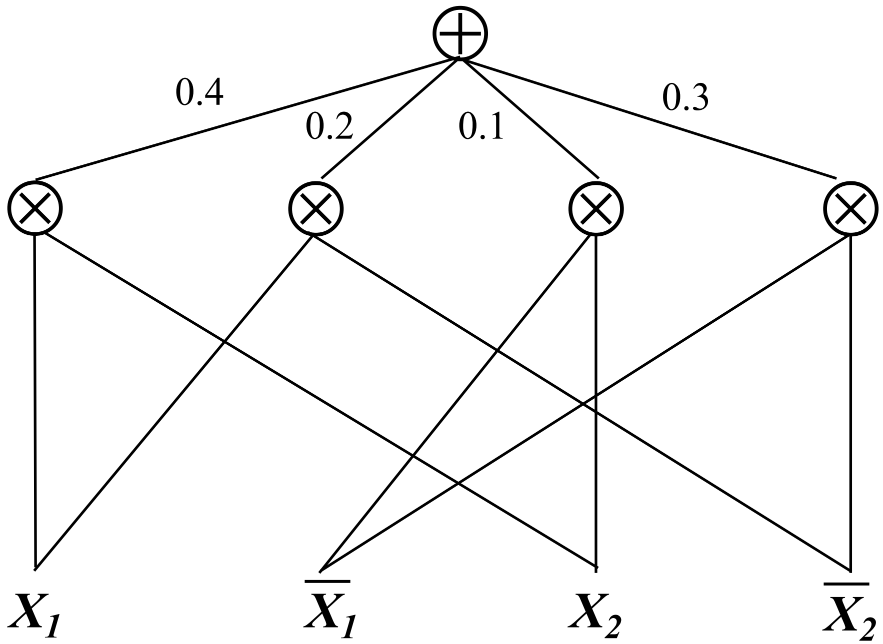
$$\begin{aligned} P(e) = &\ \mathbf{0.4 \cdot X_1 \cdot X_2} \\ &+ \mathbf{0.2 \cdot X_1 \cdot \overline{X}_2} \\ &+ 0.1 \cdot \overline{X}_1 \cdot X_2 \\ &+ 0.3 \cdot \overline{X}_1 \cdot \overline{X}_2 \end{aligned}$$

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

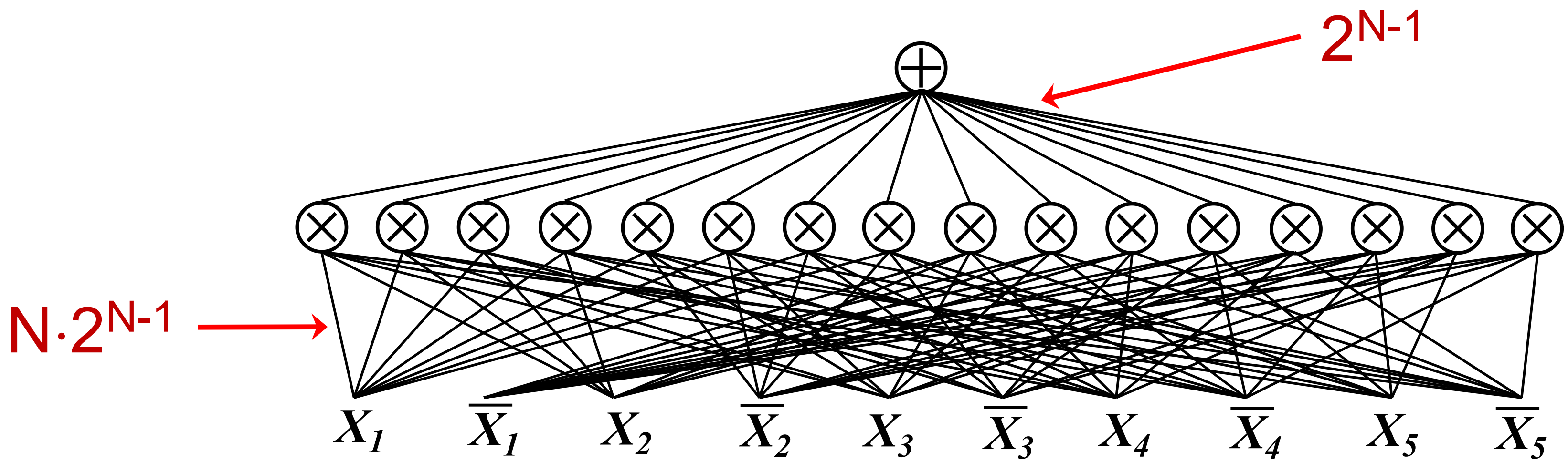


network polynomial

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

Example: Parity

Uniform distribution over states with even number of 1's



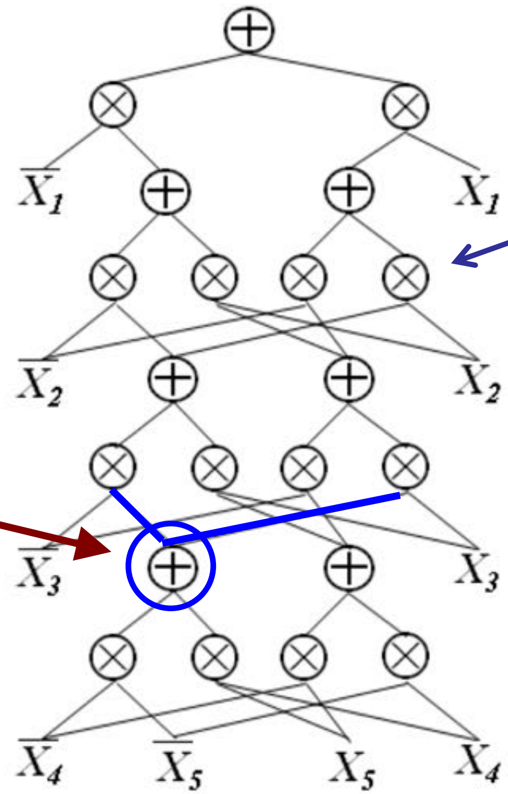
Make the computational graphs deep

Example: Parity

Uniform distribution over states with even number of 1's

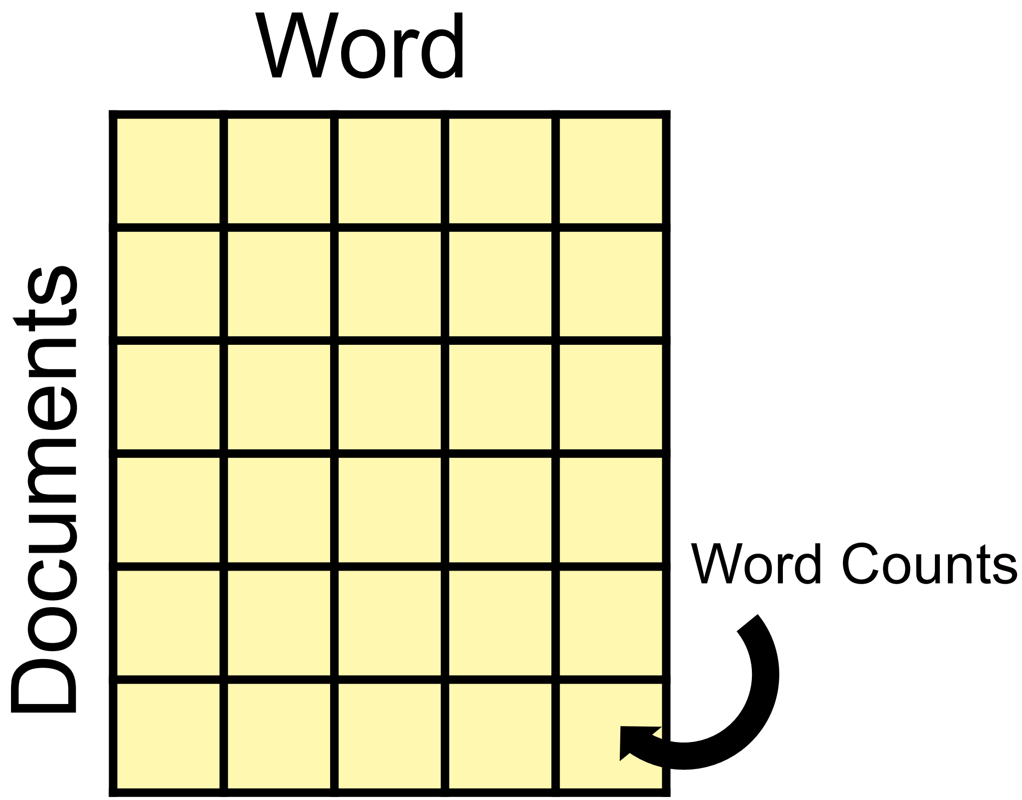
Induce many hidden layers

Reuse partial computation



Principled approach to selecting (Tree-)SPNs

Testing independence using a (non-parametric) independency test



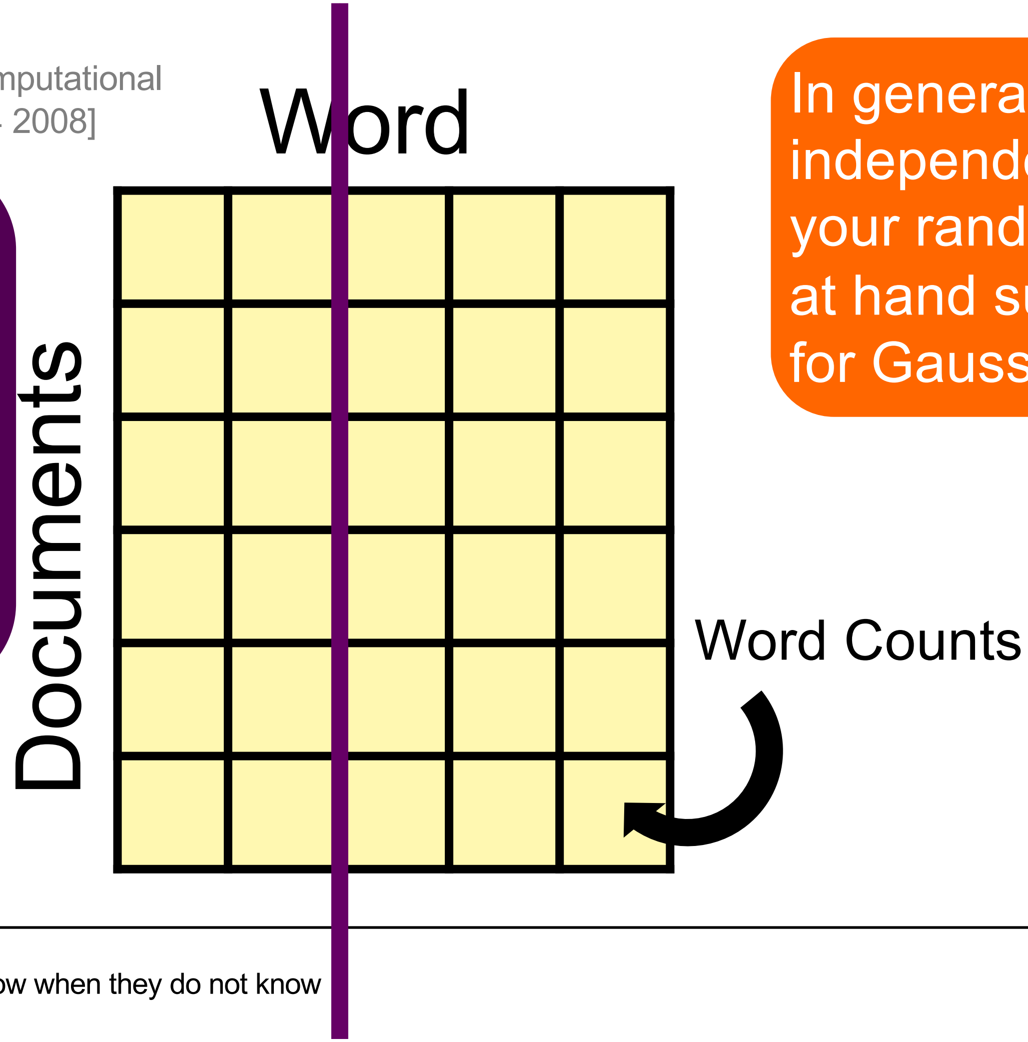
[Poon, Domingos UAI'11; Molina, Natarajan, Kersting AAAI´17]

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)



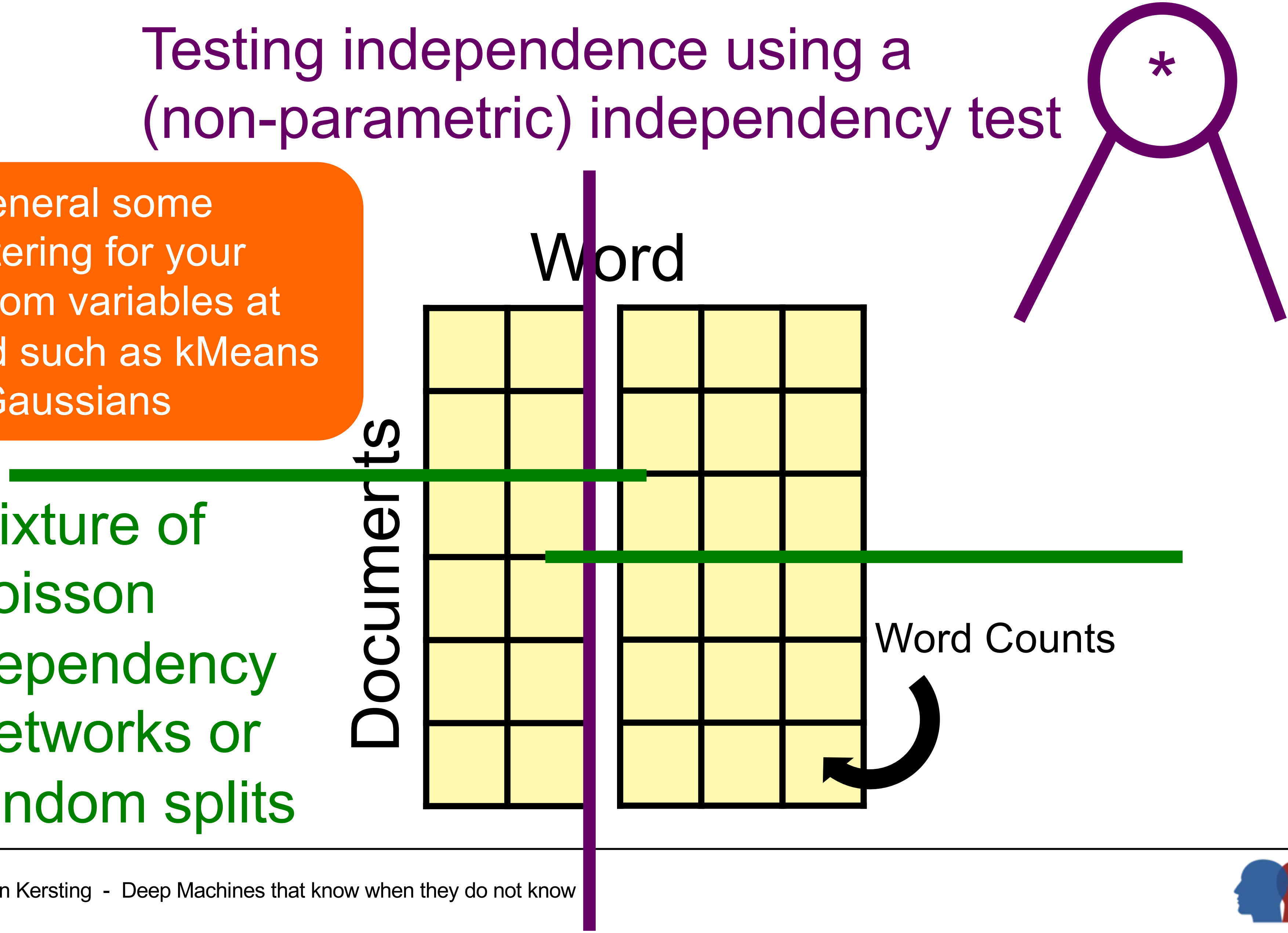
In general use the independency test for your random variables at hand such as g-test for Gaussians

Principled approach to selecting (Tree-)SPNs

Testing independence using a (non-parametric) independency test

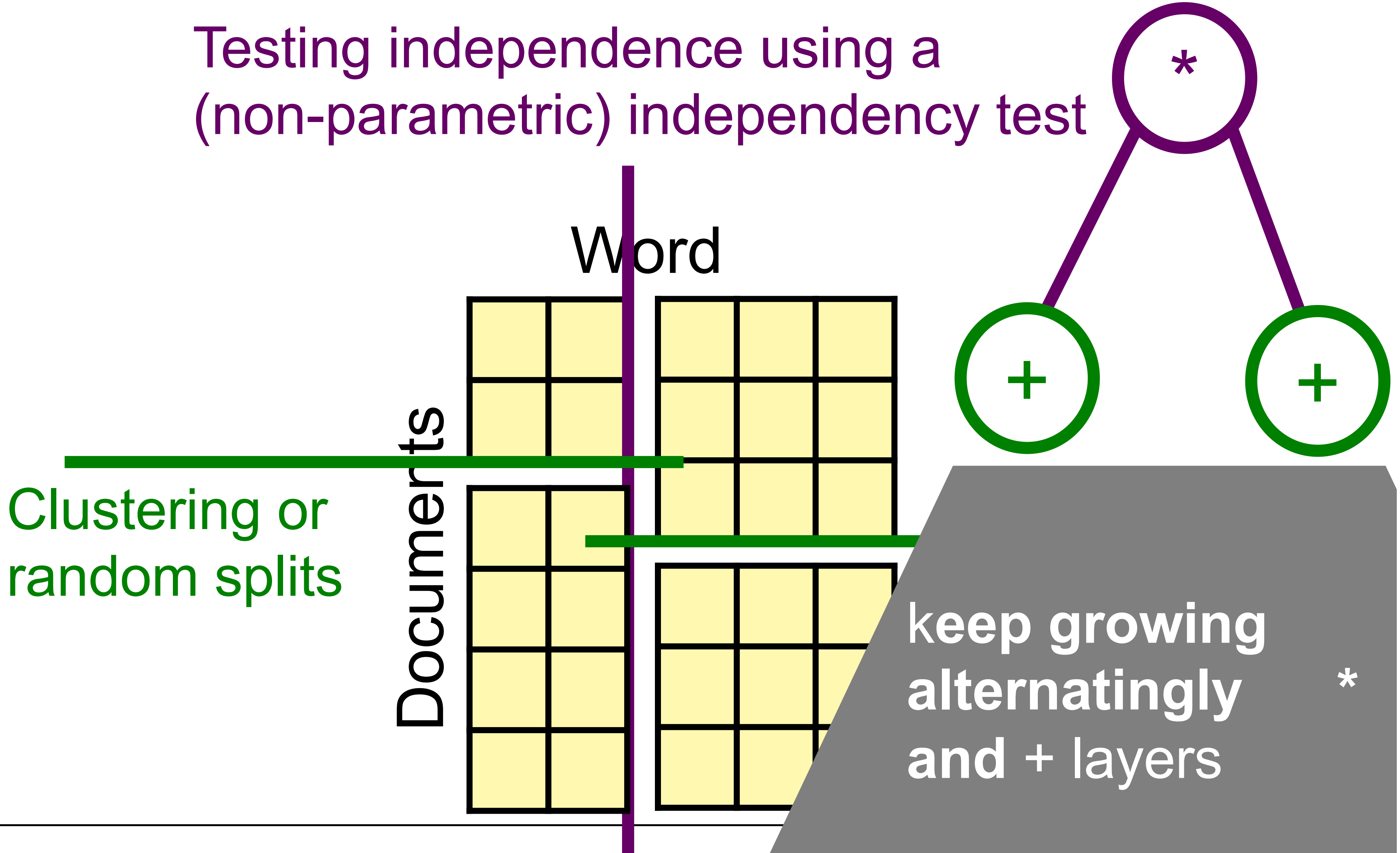
In general some clustering for your random variables at hand such as kMeans for Gaussians

Mixture of Poisson Dependency Networks or random splits



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

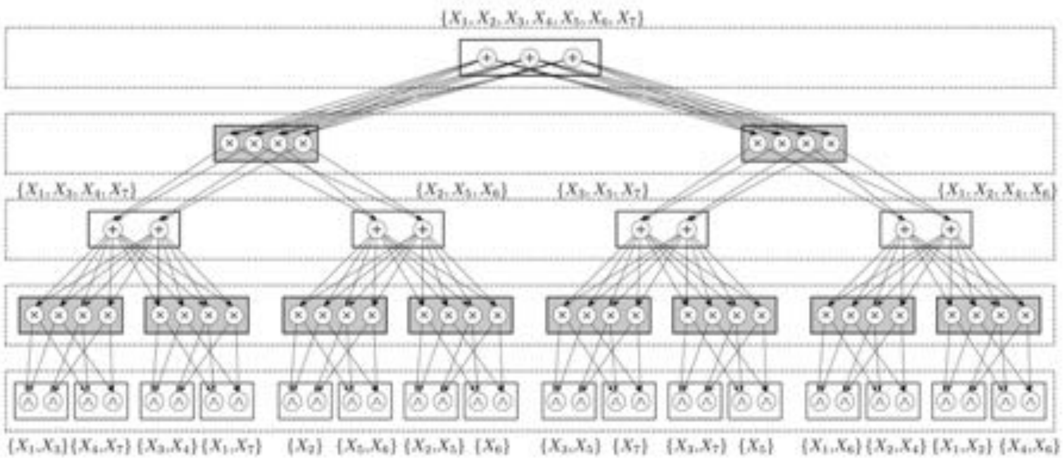
Principled approach to selecting (Tree-)SPNs



Random sum-product networks

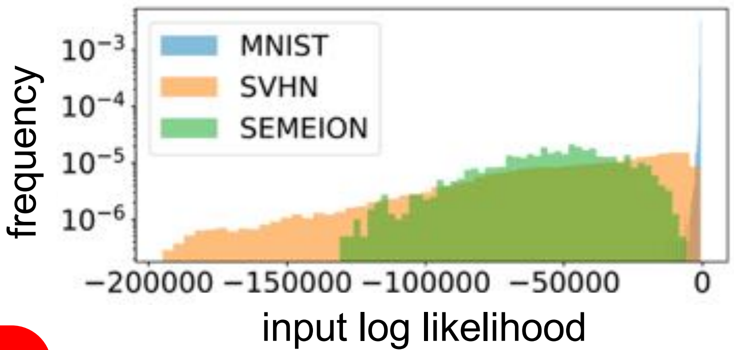
[Peharz, Vergari, Molina, Stelzner, Trapp, Kersting, Ghahramani UDL@UAI 2018]

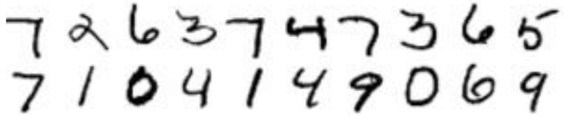
UBER AI Labs · UNIVERSITY OF CAMBRIDGE · Max Planck Institute for Intelligent Systems · UNI GRAZ · TECHNISCHE UNIVERSITÄT DARMSTADT



Build a random SPN structure over univariate distributions. 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]

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

[Molina, Vergari, Stelzner, Peharz, Subramani, Poupart, Di Mauro, Kersting 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
```

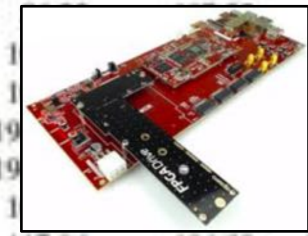
Domain Specific Language, Inference, EM, and Model Selection as well as Compilation of SPNs into TF and PyTorch and also into flat, library-free code even suitable for running on devices: C/C++,GPU, FPGA

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.



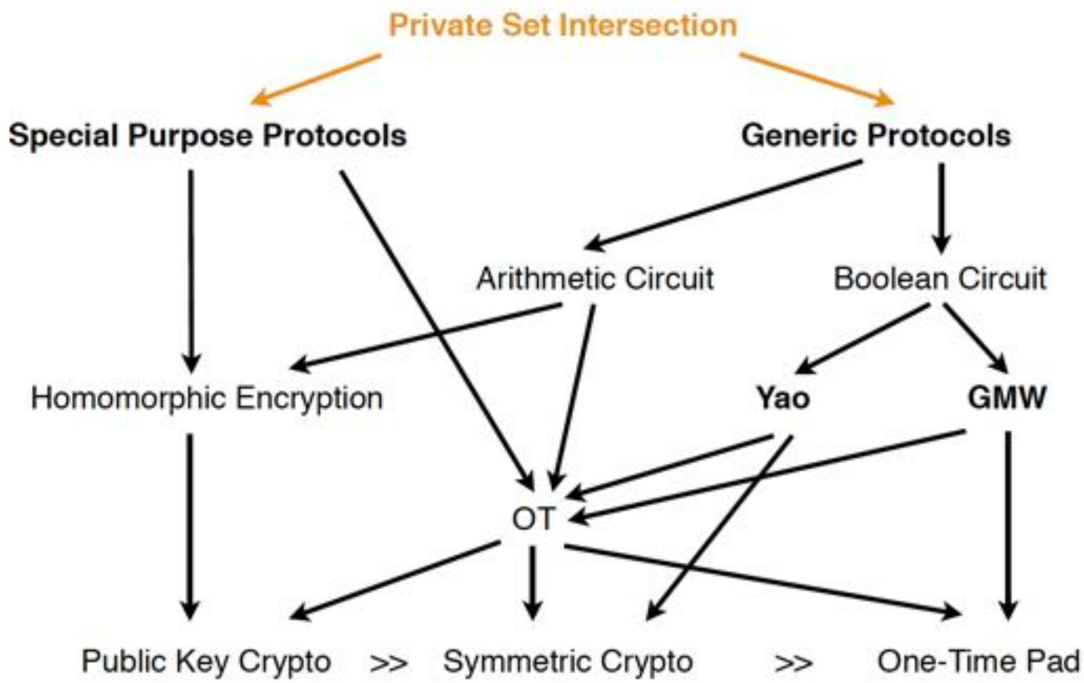
Dataset	Rows	CPU (μs)	T-CPU (rows/ μs)	CPUF (μs)	T-CPUF (rows/ μs)	GPU (μs)
Accidents	17009	2798.27	[illegible]	[illegible]	7.87	63090.94
Audio	20000	4271.78	[illegible]	[illegible]	[illegible]	[illegible]
Netflix	20000	4892.22	[illegible]	[illegible]	[illegible]	[illegible]
MSNBC200	388434	15476.05	[illegible]	[illegible]	[illegible]	[illegible]
MSNBC300	388434	10060.78	[illegible]	[illegible]	[illegible]	[illegible]
NLTCS	21574	791.80	[illegible]	[illegible]	[illegible]	[illegible]
Plants	23215	3621.71	6.41	3521.04	6.59	67004.41
NIPS5	10000	25.11	**398.31**	26.37	379.23	8210.32
NIPS10	10000	83.60	**119.61**	84.39	118.49	11550.82
NIPS20	10000	191.30	52.27	182.73	**54.72**	18689.04
NIPS30	10000	387.61	25.80	349.84	**28.58**	25355.93
NIPS40	10000	551.64	18.13	471.26	**21.22**	30820.49
NIPS50	10000	812.44	12.31	792.13	12.62	36355.60
NIPS60	10000	1046.38	9.56	662.53	**15.09**	40778.36
NIPS70	10000	1148.17	8.71	1134.80	8.81	46759.26
NIPS80	10000	1556.99	6.42	1277.81	7.83	63217.99

Dataset	T-GPU (rows/ μs)	FPGA Cycle Counter	FPGAC (μs)	T-FPGAC (rows/ μs)	FPGA (μs)	T-FPGA (rows/ μs)
Accidents	0.27	17249	[illegible]	[illegible]	[illegible]	**24.44**
Audio	[illegible]	20317	[illegible]	[illegible]	[illegible]	**26.28**
Netflix	[illegible]	20322	[illegible]	[illegible]	[illegible]	**30.58**
MSNBC200	[illegible]	388900	[illegible]	[illegible]	[illegible]	**77.56**
MSNBC300	[illegible]	388810	[illegible]	[illegible]	[illegible]	**78.74**
NLTCS	[illegible]	21904	[illegible]	[illegible]	[illegible]	**38.12**
Plants	0.35	23592	117.96	196.80	778.00	**29.84**
NIPS5	1.22	10236	51.18	195.39	337.30	[illegible]
NIPS10	0.87	10279	51.40	194.57	464.30	21.54
NIPS20	0.54	10285	51.43	194.46	543.60	18.40
NIPS30	0.39	10308	51.80	193.06	592.30	16.88
NIPS40	0.32	10306	51.53	194.06	632.20	15.82
NIPS50	0.28	10559	52.80	189.41	720.60	**13.88**
NIPS60	0.25	12271	61.36	162.99	799.20	12.51
NIPS70	0.21	14022	70.11	142.63	858.60	**11.65**
NIPS80	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

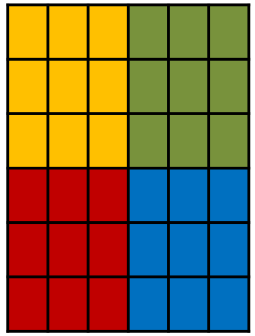
NIPS MSPN
Gates: 1001477 Yao Bytes: 32047264 Depth: 970

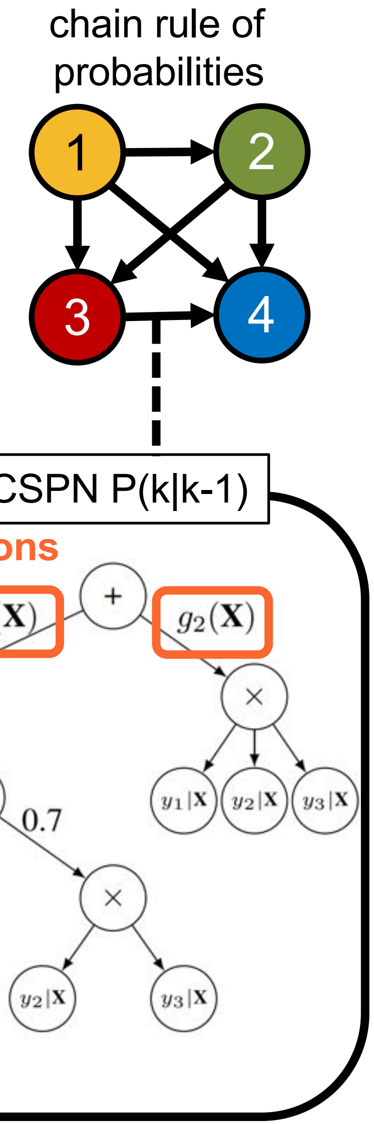
Homomorphic sum-product network

[Molina, Weinert, Treiber, Schneider, Kersting 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]

CSPNs
PixelCNNs





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]

Original

SPN

DBM

DBN

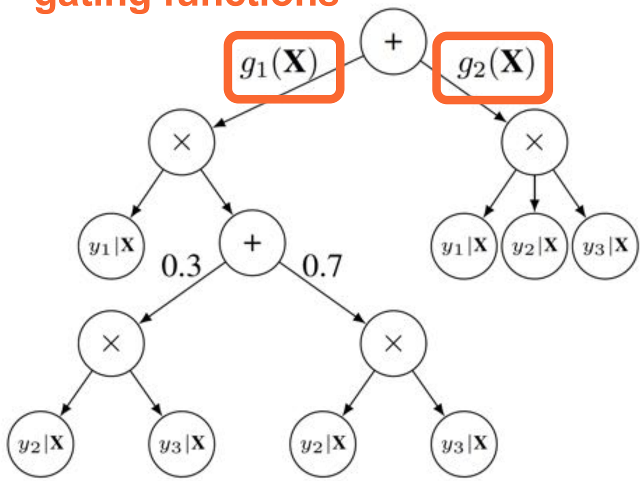
PCA

kNN

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

gating functions

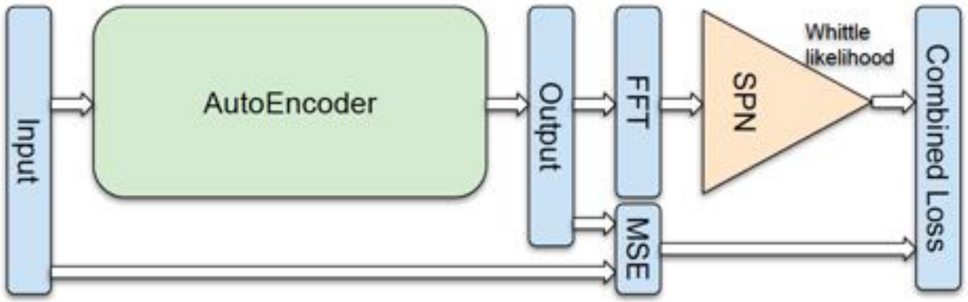


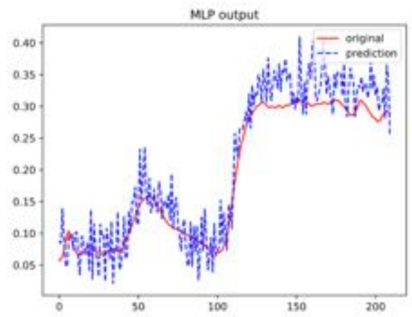
Conditional SPNs

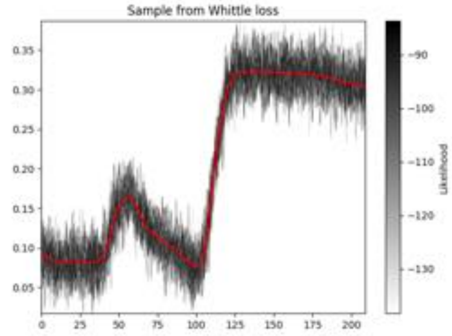
[Shao, Molina, Vergari, Peharz, Liebig, Kersting TPM@ICML 2019]

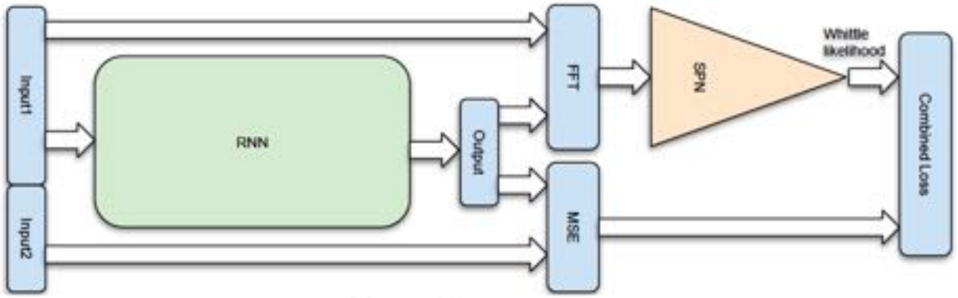
CAML DFG

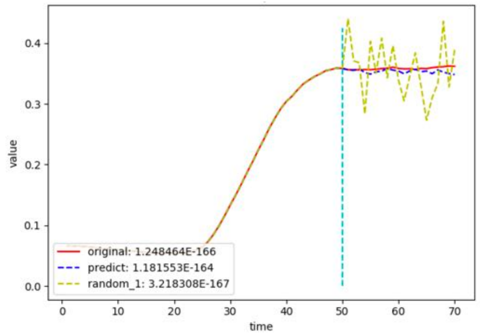
And SPNs may also provide likelihoods for time series

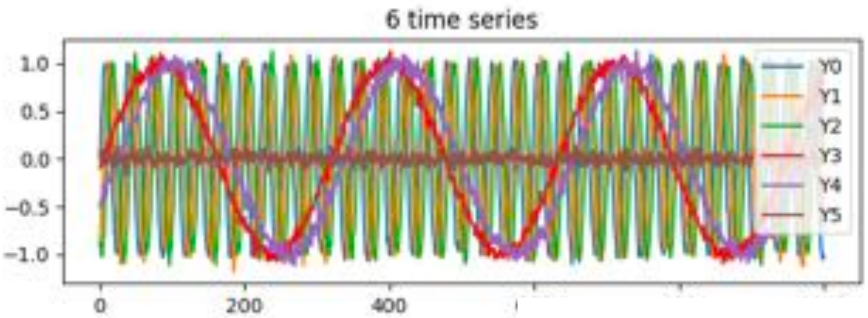


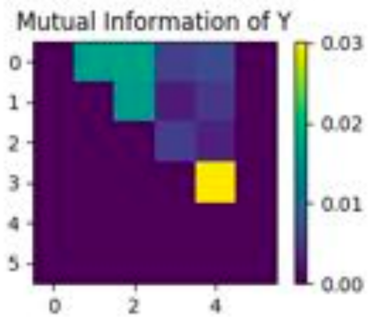






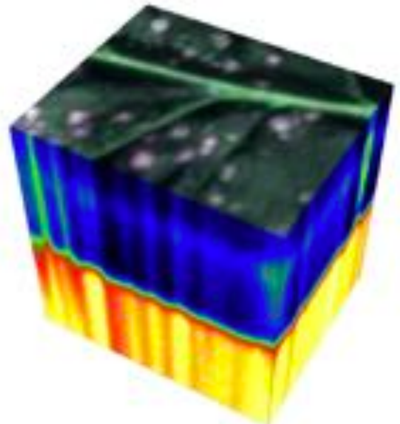




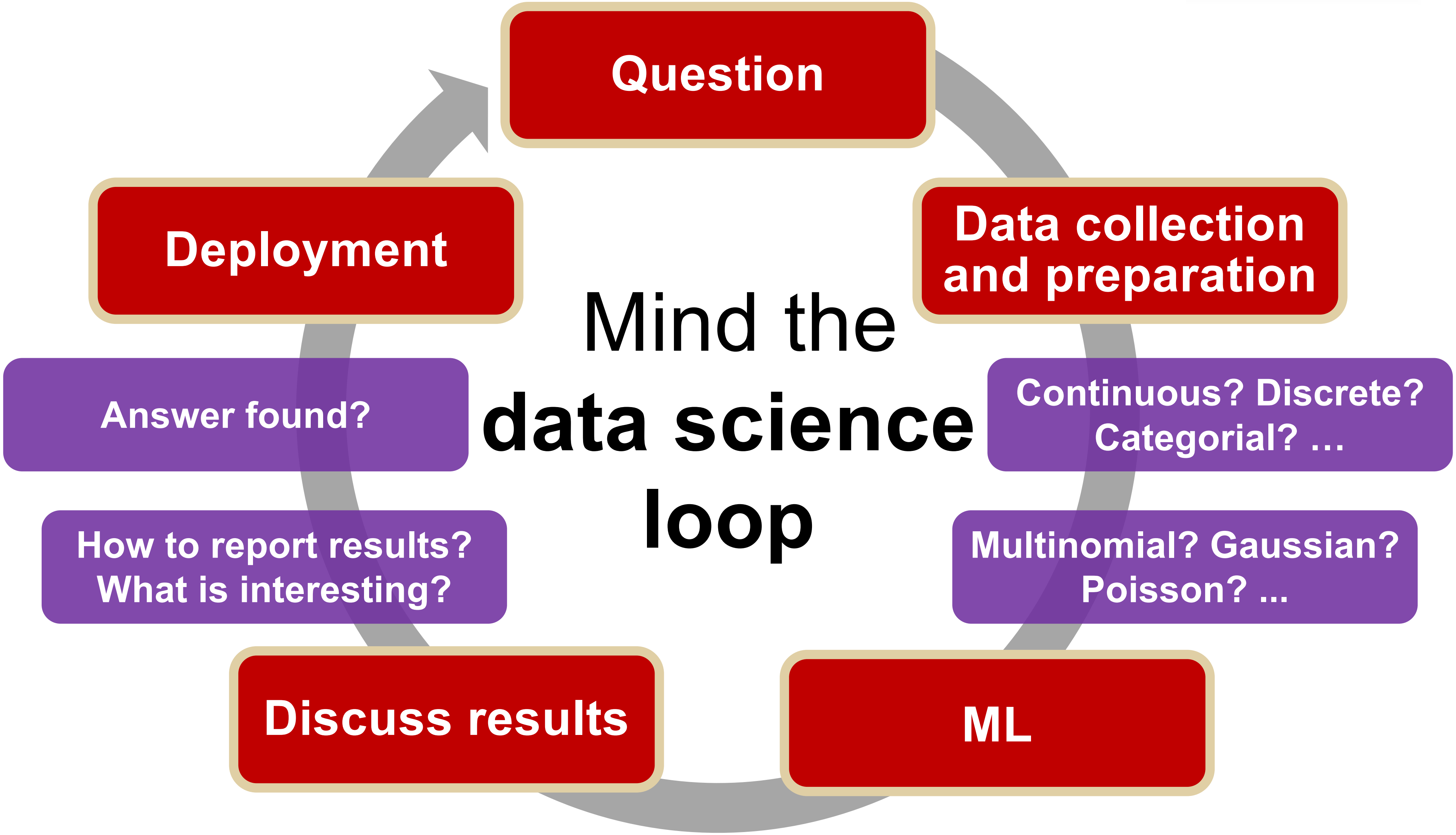


Whittle SPNs

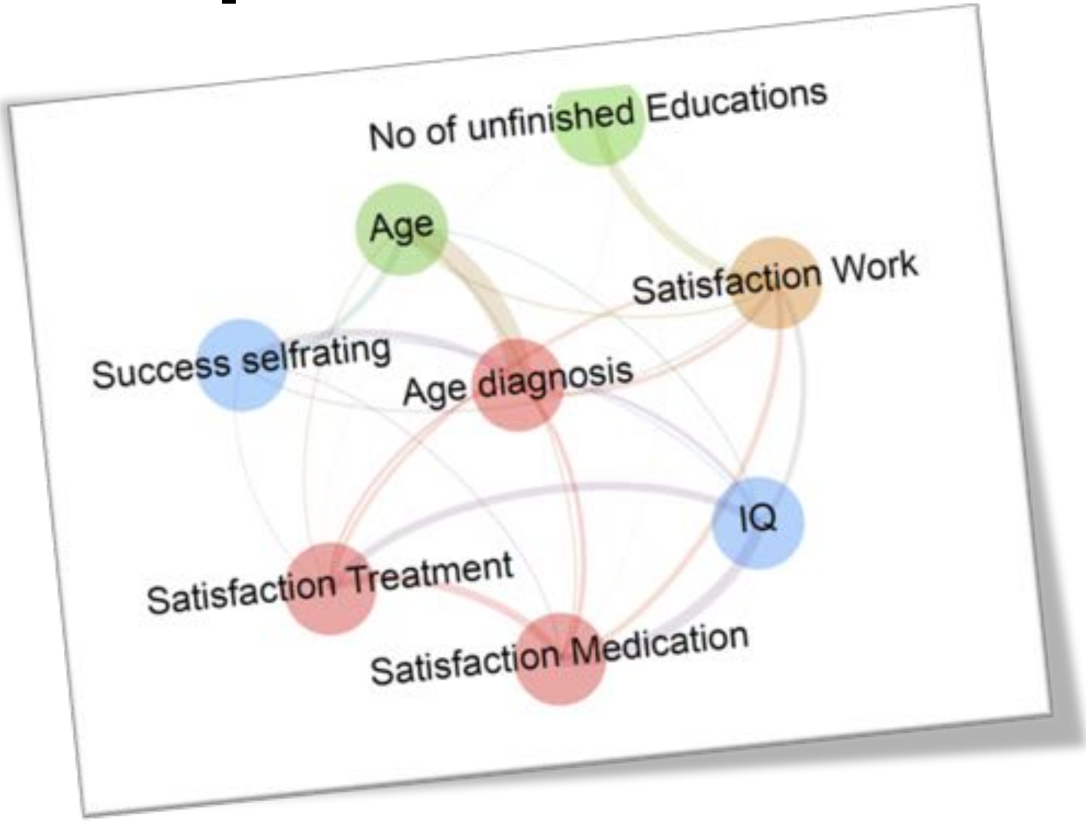
[Yu, Kersting 2019, to be submitted]



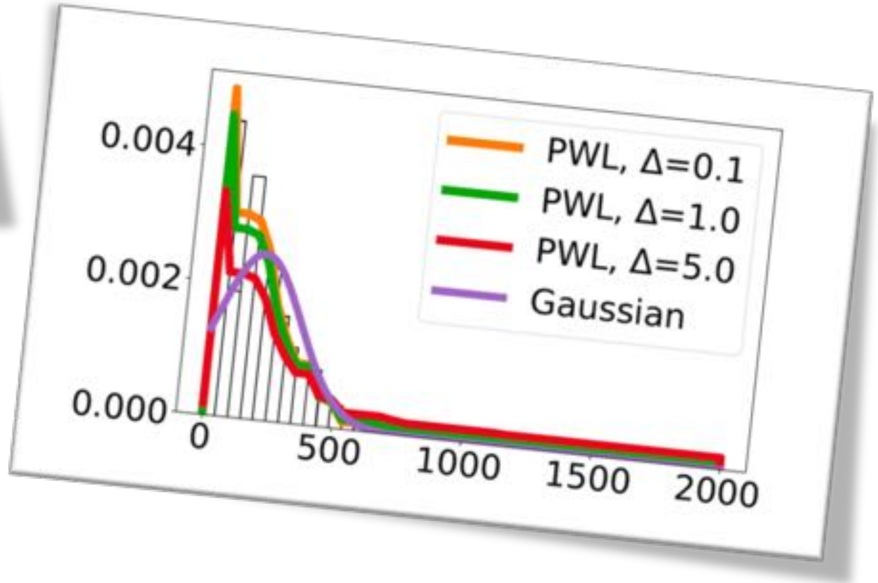
MADESI Federal Ministry of Education and Research Bundesanstalt für Landwirtschaft und Ernährung DePhenSe


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

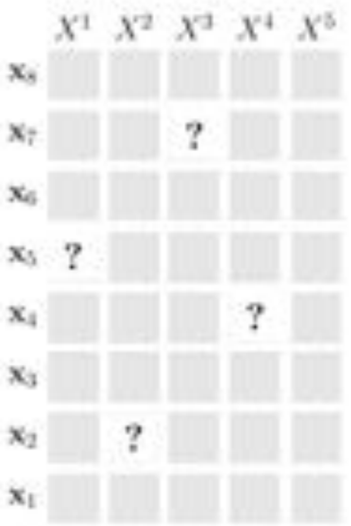
Use nonparametric independency tests and piece-wise linear approximations

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

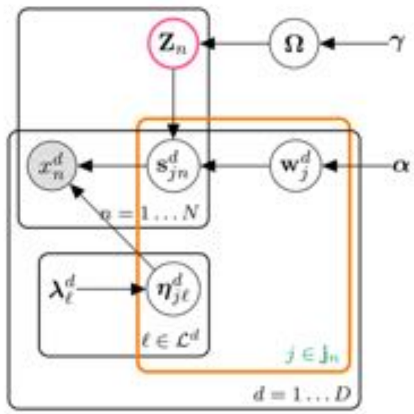
[Vergari, Molina, Peharz, Ghahramani, Kersting, Valera AAAI 2019]

The Explorative Automatic Statistician

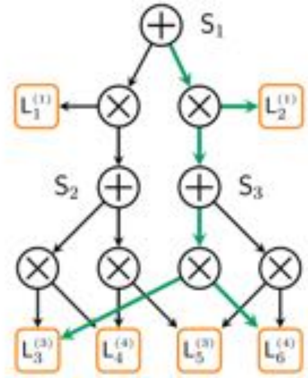
missing value



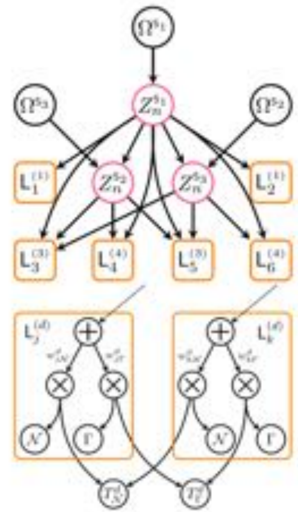
We can even automatically discovers the statistical types and parametric forms of the variables



Bayesian Type Discovery



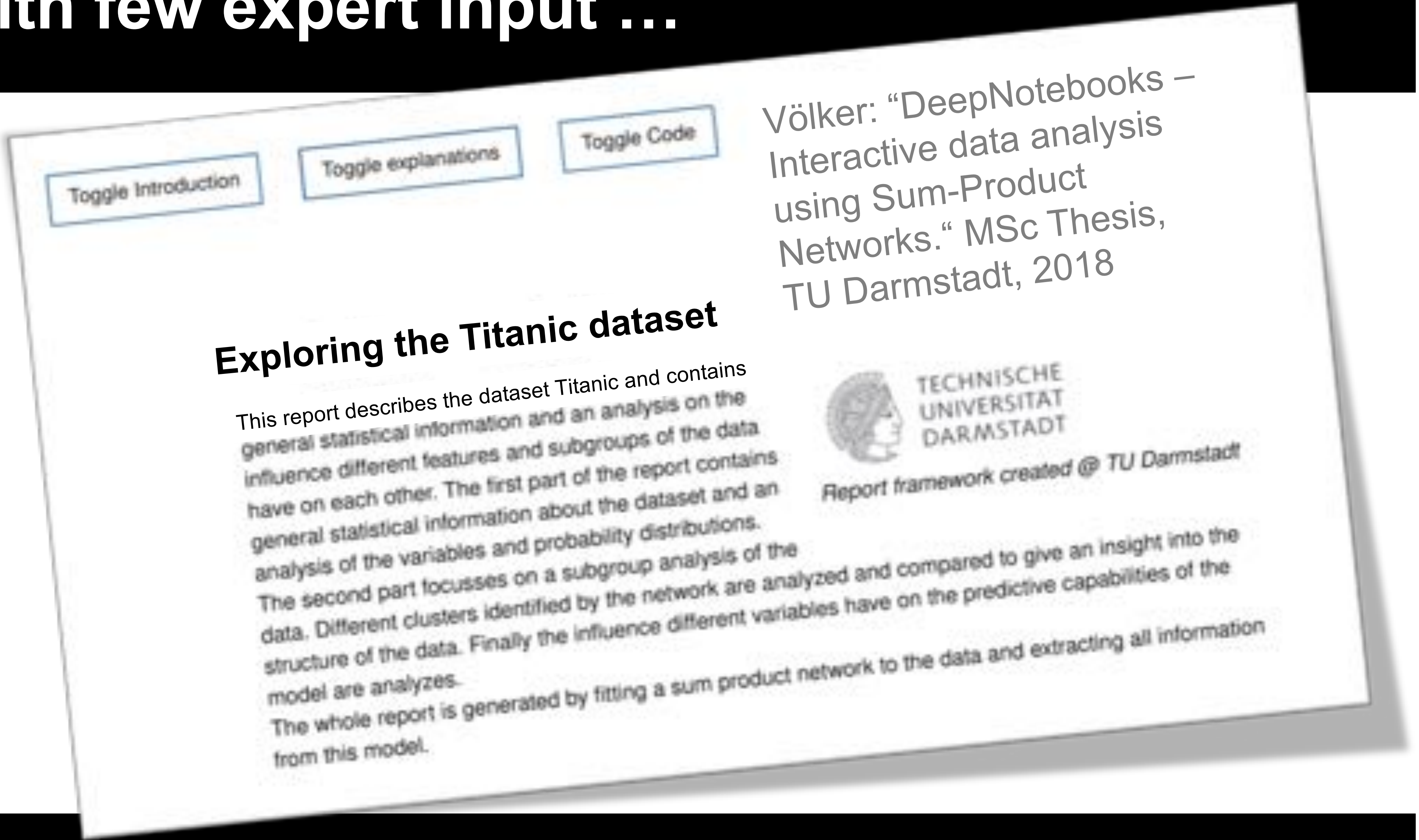
Mixed Sum-Product Network



Automatic Statistician

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

Völker: "DeepNotebooks – Interactive data analysis using Sum-Product Networks." MSc Thesis, TU Darmstadt, 2018


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.

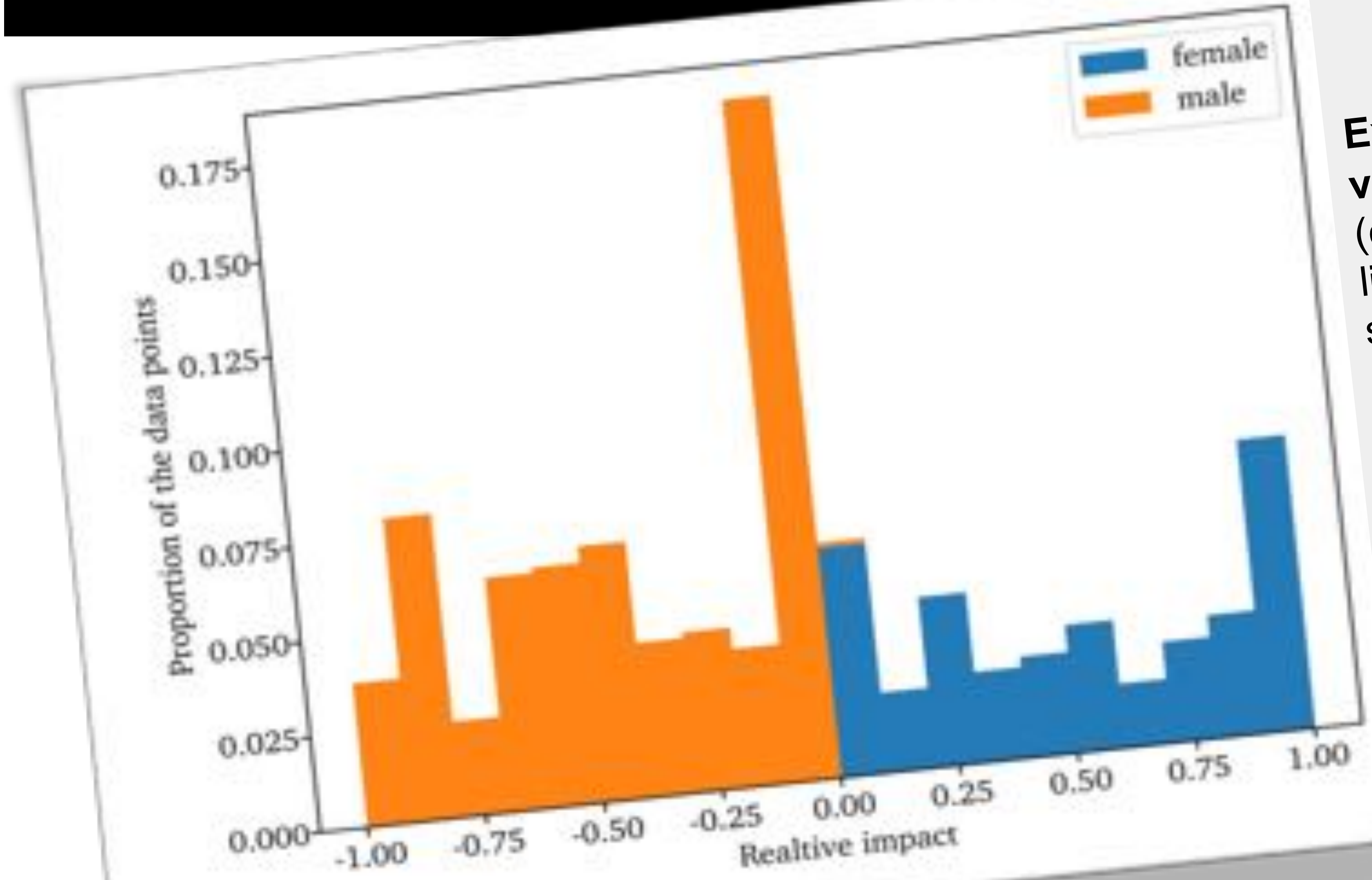
TECHNISCHE UNIVERSITÄT DARMSTADT

Report framework created @ TU Darmstadt

…and can compile data reports automatically

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

The machine understands the data with no 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

P(heart attack | )?

The New York Times

Opinion

A.I. Is Harder Than You Think

and Data Science

By Gary Marcus and Ernest Davis

Mr. Marcus is a professor of psychology and neural science. Mr. Davis is a professor of computer science.

May 18, 2018

P(heart attack | )?


The New York Times

Opinion

A.I. Is Harder Than You Think

and Data Science

By Gary Marcus and Ernest Davis

Mr. Marcus is a professor of psychology and neural science. Mr. Davis is a professor of computer science.

May 18, 2018

P(heart attack | )?

The New York Times

Opinion

A.I. Is Harder Than You Think

and Data Science

By Gary Marcus and Ernest Davis

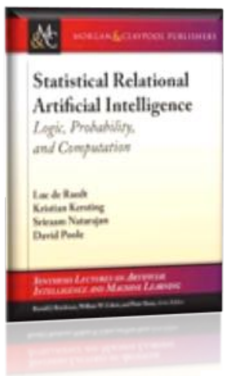
Mr. Marcus is a professor of psychology and neural science. Mr. Davis is a professor of computer science.

May 18, 2018

P(heart attack | )?

Crossover of ML and DS with data & programming abstractions

De Raedt, Kersting, Natarajan, Poole: Statistical Relational Artificial Intelligence: Logic, Probability, and Computation. Morgan and Claypool Publishers, ISBN: 9781627058414, 2016.



KATHOLIEKE UNIVERSITEIT LEUVEN

THE UNIVERSITY OF TEXAS AT DALLAS

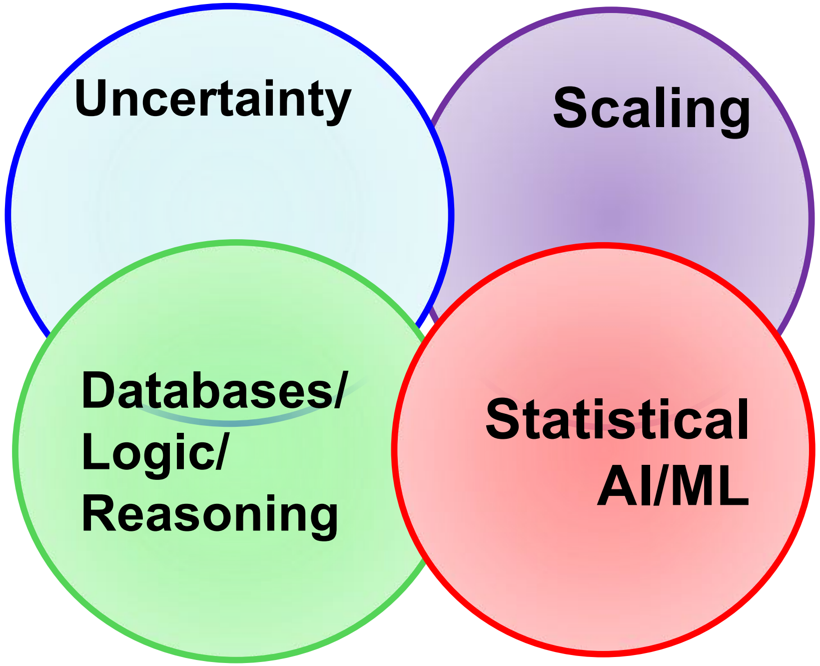
TECHNISCHE UNIVERSITÄT DARMSTADT

THE UNIVERSITY OF BRITISH COLUMBIA

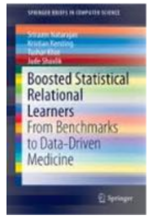
building general-purpose data science and ML machines

make the ML/DS expert more effective

increases the number of people who can successfully build ML/DS applications

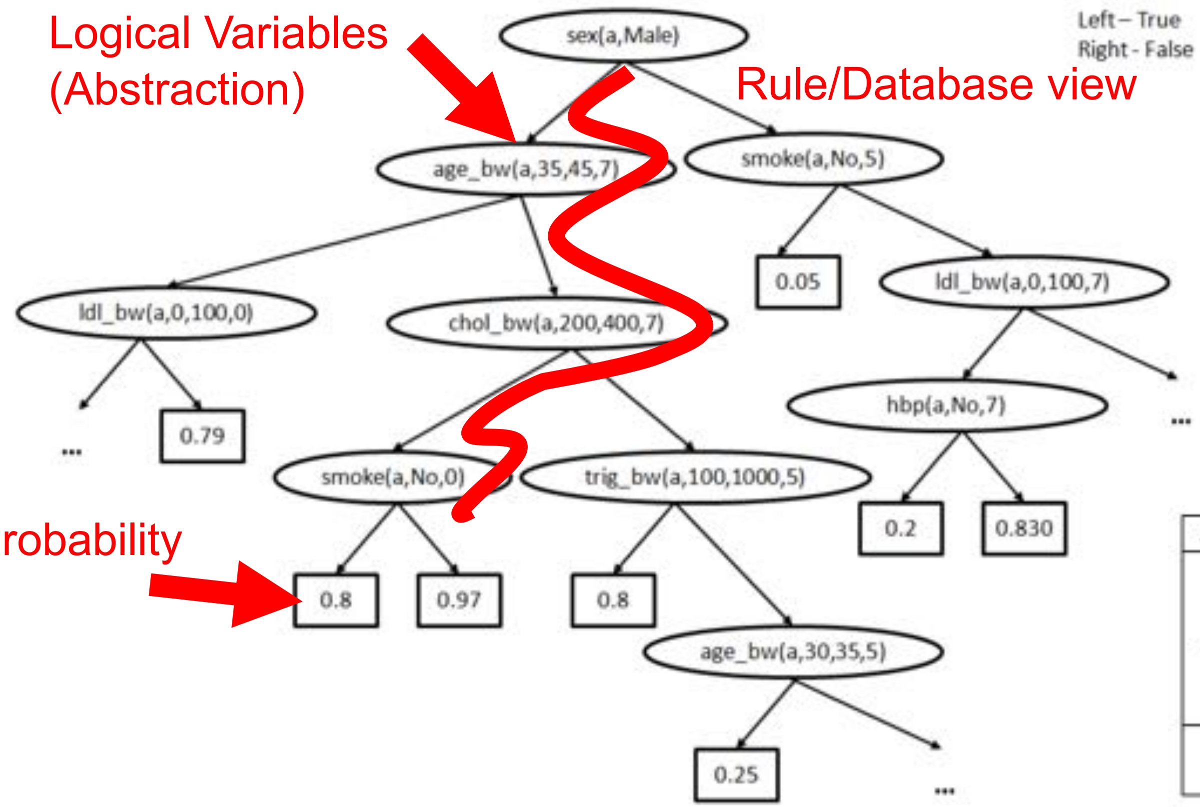


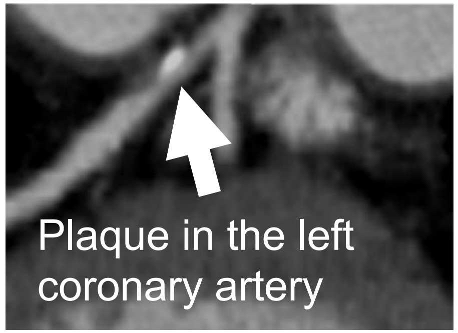
Natarajan, Khot, Kersting, Shavlik. Boosted Statistical Relational Learners. Springer Brief 2015



Understanding Electronic Health Records

Atherosclerosis is the cause of the majority of Acute Myocardial Infarctions (heart attacks)





[Circulation; 92(8), 2157-62, 1995; JACC; 43, 842-7, 2004]

Algorithm	Accuracy	AUC-ROC
J48	0.667	0.607
SVM	0.667	0.5
AdaBoost	0.667	0.608
Bagging	0.677	0.613
NB	0.75	0.653
RPT	0.669*	0.778
RFGB	0.667*	0.819

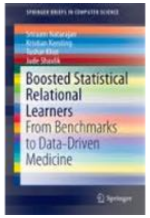
The higher, the better

25%

Algorithm for Mining Markov Logic Networks	Likelihood The higher, the better	AUC-ROC The higher, the better	AUC-PR The higher, the better	Time The lower, the better
Boosting	0.81	0.96	0.93	9s
LSM	0.73	0.54	0.62	93 hrs

11% 78% 50% 37200x faster

[Kersting, Driessens ICML´08; Karwath, Kersting, Landwehr ICDM´08; Natarajan, Joshi, Tadepelli, Kersting, Shavlik. IJCAI´11; Natarajan, Kersting, Ip, Jacobs, Carr IAAI `13; Yang, Kersting, Terry, Carr, Natarajan AIME ´15; Khot, Natarajan, Kersting, Shavlik ICDM´13, MLJ´12, MLJ´15, Yang, Kersting, Natarajan BIBM`17]



Natarajan, Khot, Kersting, Shavlik. Boosted Statistical Relational Learners. Springer Brief 2015

TECHNISCHE UNIVERSITÄT DARMSTADT

UTD THE UNIVERSITY OF TEXAS AT DALLAS

https://starling.utdallas.edu/software/boostsrl/wiki/

StARLinG LAB — People | Publications | Projects | Software | Datasets | Blog

BOOSTSRL BASICS

- Getting Started
- File Structure
- Basic Parameters
- Advanced Parameters
- Basic Modes
- Advanced Modes

ADVANCED BOOSTSRL

- Default (RDN-Boost)
- MLN-Boost
- Regression
- One-Class Classification
- Cost-Sensitive SRL
- Learning with Advice
- Approximate Counting
- Discretization of Continuous-Valued Attributes
- Lifted Relational Random Walks
- Grounded Relational Random Walks

APPLICATIONS

- Natural Language Processing

BoostSRL Wiki

BoostSRL (Boosting for Statistical Relational Learning) is a gradient-boosting based approach to learning different types of SRL models. As with the standard gradient-boosting approach, our approach turns the model learning problem to learning a sequence of regression models. The key difference to the standard approaches is that we learn relational regression models i.e., regression models that operate on relational data. We assume the data in a predicate logic format and the output are essentially first-order regression trees where the inner nodes contain conjunctions of logical predicates. For more details on the models and the algorithm, we refer to our book on this topic.

Sriraam Natarajan, Tushar Khot, Kristian Kersting and Jude Shavlik, Boosted Statistical Relational Learners: From Benchmarks to Data-Driven Medicine . SpringerBriefs in Computer Science, ISBN: 978-3-319-13643-1, 2015

Human-in-the-loop learning

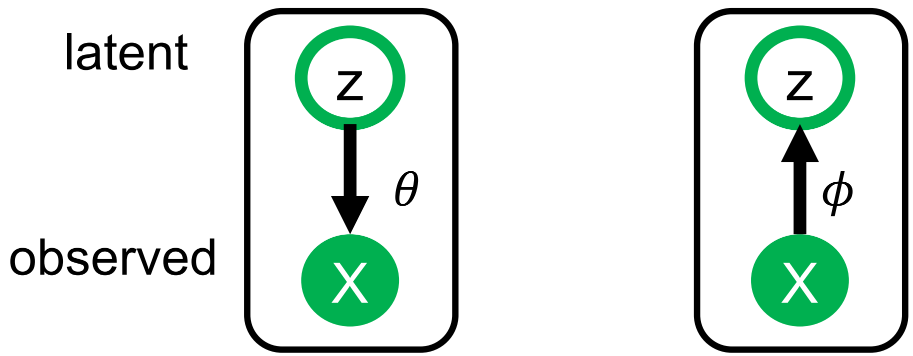
In general, computing the exact posterior is intractable, i.e., inverting the generative process to determine the state of latent variables corresponding to an input is time-consuming and error-prone.

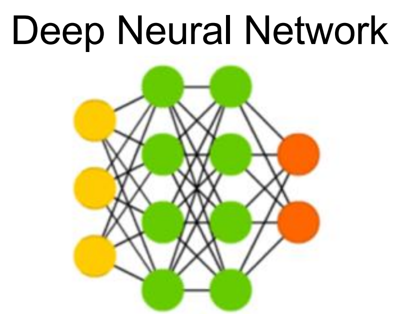
Deep Probabilistic Programming

```
import pyro.distributions as dist

def model(data):
    # define the hyperparameters that control the beta prior
    alpha0 = torch.tensor(10.0)
    beta0 = torch.tensor(10.0)
    # sample f from the beta prior
    f = pyro.sample("latent_fairness", dist.Beta(alpha0, beta0))
    # loop over the observed data
    for i in range(len(data)):
        # observe datapoint i using the bernoulli
        # likelihood Bernoulli(f)
        pyro.sample("obs_{}".format(i), dist.Bernoulli(f), obs=data[i])
```

```
def guide(data):
    # register the two variational parameters with Pyro.
    alpha_q = pyro.param("alpha_q", torch.tensor(15.0),
                         constraint=constraints.positive)
    beta_q = pyro.param("beta_q", torch.tensor(15.0),
                        constraint=constraints.positive)
    # sample latent_fairness from the distribution Beta(alpha_q, beta_q)
    pyro.sample("latent_fairness", dist.Beta(alpha_q, beta_q))
```

(2) Ease the implementation by some high-level, probabilistic programming language





(1) Instead of optimizating variational parameters for every new data point, use a deep network to predict the posterior given X [Kingma, Welling 2013, Rezende et al. 2014]

[Stelzner, Molina, Peharz, Vergari, Trapp, Valera, Ghahramani, Kersting ProgProb 2018]

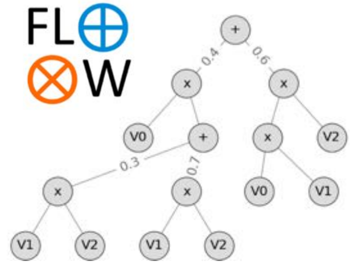
Sum-Product Probabilistic Programming

```python
import pyro.distributions as dist

def model(data):
    # define the hyperparameters that control the beta prior
    alpha0 = torch.tensor(10.0)
    beta0 = torch.tensor(10.0)
    # sample f from the beta prior
    f = pyro.sample("latent_fairness", dist.Beta(alpha0, beta0))
    # loop over the observed data
    for i in range(len(data)):
        # observe datapoint i using the bernoulli
        # likelihood Bernoulli(f)
        pyro.sample("obs_{}".format(i), dist.Bernoulli(f), obs=data[i])
```

```python
def guide(data):
    # register the two variational parameters with Pyro.
    alpha_q = pyro.param("alpha_q", torch.tensor(15.0),
                         constraint=constraints.positive)
    beta_q = pyro.param("beta_q", torch.tensor(15.0),
                        constraint=constraints.positive)
    # sample latent_fairness from the distribution Beta(alpha_q, beta_q)
    pyro.sample("latent_fairness", dist.Beta(alpha_q, beta_q))
```

Sum-Product Network

FL⊕⊗W

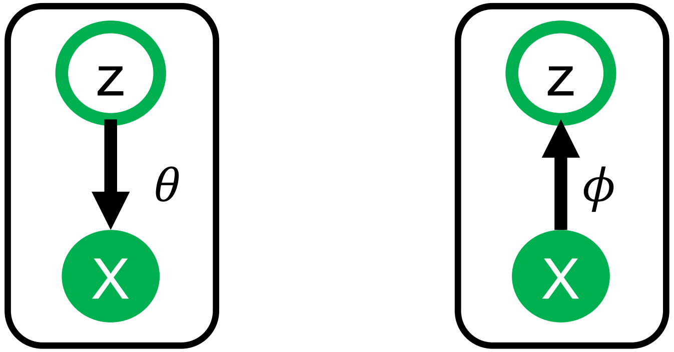


Deep Neural Network

(2) Ease the implementation by some high-level, probabilistic programming language

latent

observed



(1) Instead of optimizating variational parameters for every new data point, use a deep network to predict the posterior given X

[Kingma, Welling 2013, Rezende et al. 2014]

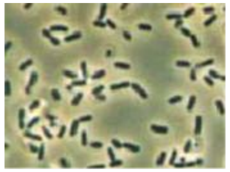
Unsupervised scene understanding

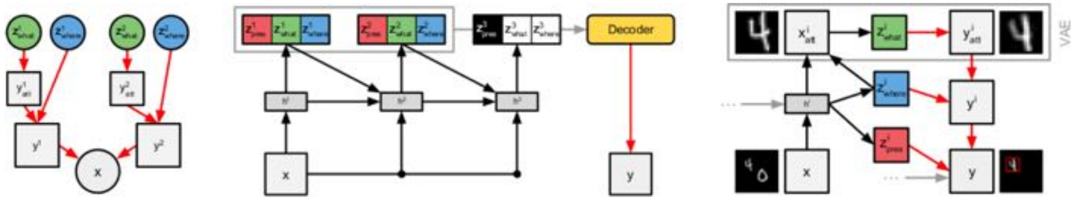
[Stelzner, Peharz, Kersting ICML 2019]

UNIVERSITY OF CAMBRIDGE

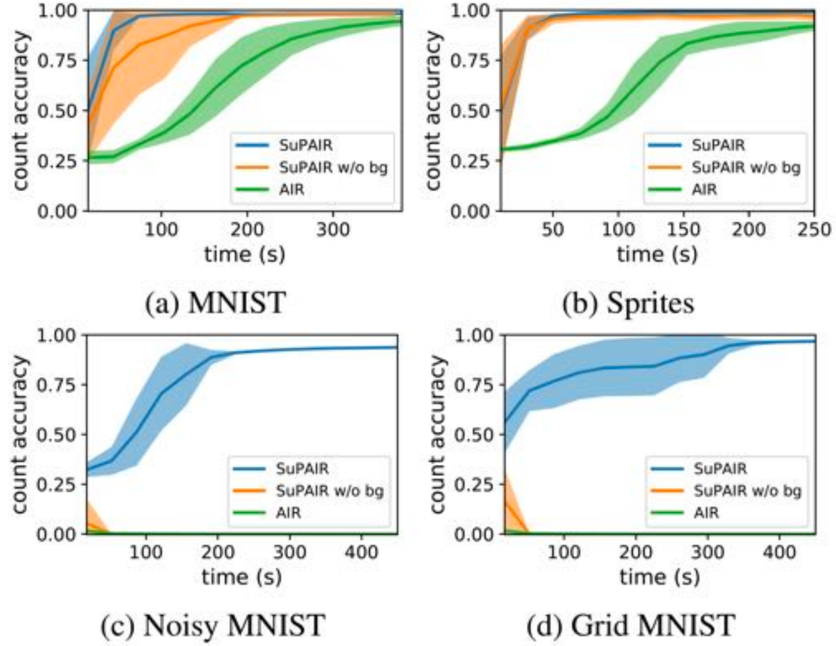
TECHNISCHE UNIVERSITÄT DARMSTADT

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

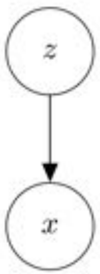




[Attend-Infer-Repeat (AIR) model, Hinton et al. NIPS 2016]

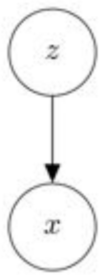


VAE



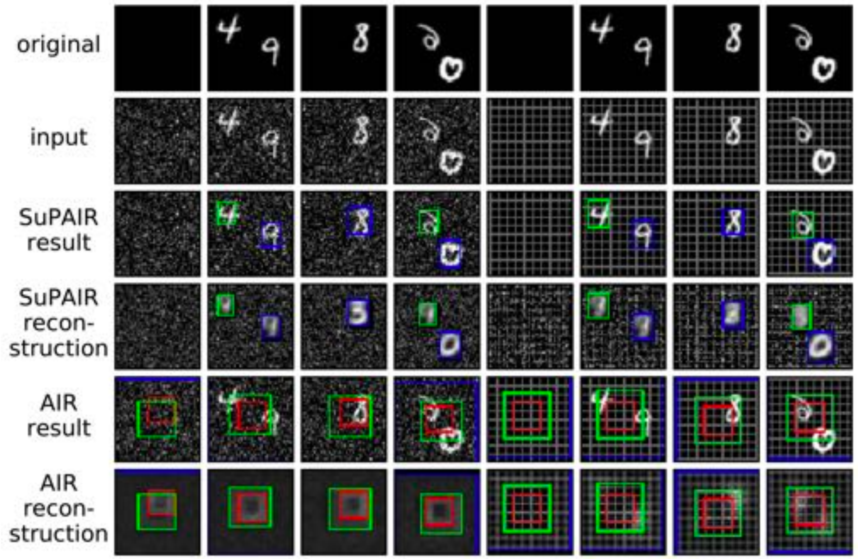
Replace VAE by SPN as object model

SPN



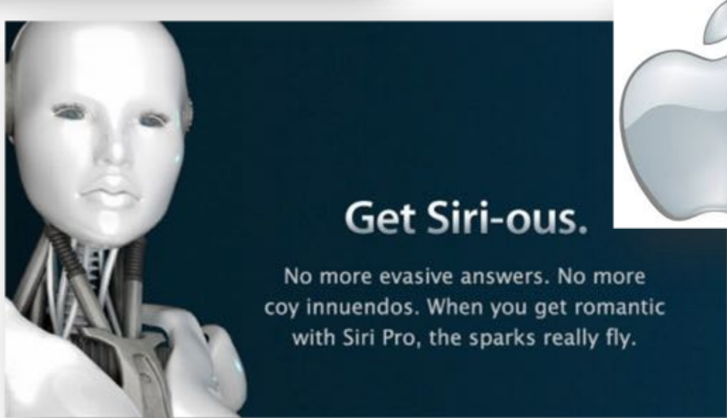
- infinite mixture model
- intractable density
- intractable posterior

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



There are strong invests into (deep) probabilistic programming

RelationalAI, Apple, Microsoft and Uber are investing hundreds of millions of US dollars



Since we need languages for Systems AI,

the computational and mathematical modeling of complex AI systems.

[Kordjamshidi, Roth, Kersting: “Systems AI: A Declarative Learning Based Programming Perspective.“ IJCAI-ECAI 2018]

Th … be a

…b … ne,
de … s

Eric Schmidt, Executive Chairman, Alphabet Inc.: Just Say "Yes”, Stanford Graduate School of Business, May 2, 2017.https://www.youtube.com/watch?v=vbb-AjiXyh0.

UNIVERSITY OF CAMBRIDGE

UBER AI Labs

Max Planck Institute for Intelligent Systems

UNI GRAZ

FLOW

TECHNISCHE UNIVERSITÄT DARMSTADT

Microsoft Research

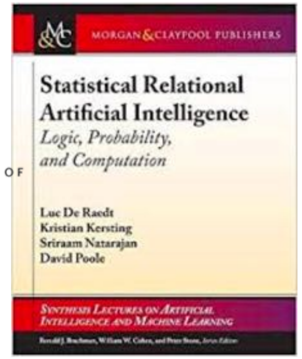
UBER AI Labs

KATHOLIEKE UNIVERSITEIT LEUVEN

UTD THE UNIVERSITY OF TEXAS AT DALLAS

TECHNISCHE UNIVERSITÄT DARMSTADT

UBC THE UNIVERSITY OF BRITISH COLUMBIA



relationalAI

Getting deep systems that reason and know when they don't know

Responsible AI systems that explain their decisions and co-evolve with the humans

Open AI systems that are easy to realize and understandable for the domain experts

„Tell the AI when it is right for the wrong reasons and it adapts its behavior"

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google's Inception network, highlighting positive pixels. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), **"Acoustic guitar"** ($p = 0.24$) **and "Labrador"** ($p = 0.21$)

Teso, Kersting AIES 2019

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

Human algorithms teaches AI a lot

The twin science: cognitive science

"How do we humans get so much from so little?" and by that I mean how do we acquire our understanding of the world given what is clearly by today's engineering standards so little data, so little time, and so little energy.

Centre for Cognitive Science at TU Darmstadt

Establishing cognitive science at the Technische Universität Darmstadt is a long-term commitment across multiple departments (see Members to get an impression on the interdisciplinary of the supporting groups and departments). The TU offers a strong foundation including several established top engineering groups in Germany, a prominent computer science department (which is among the top four in Germany), a

Centre for Cognitive Science

Josh Tenenbaum, MIT

Lake, Salakhutdinov, Tenenbaum, Science 350 (6266), 1332-1338, 2015

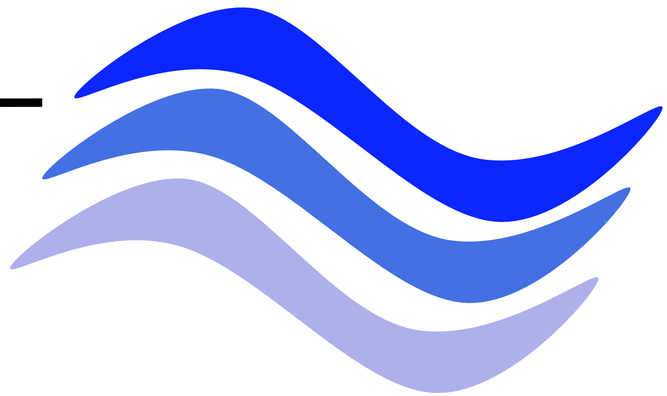
Tenenbaum, Kemp, Griffiths, Goodman, Science 331 (6022), 1279-1285, 2011

Indeed, AI has great impact, but …

+ **AI is more than deep neural networks.** Probabilistic and causal models are whiteboxes that provide insights into applications
+ **AI is more than a single table.** Loops, graphs, different data types, relational DBs, … are central to ML/AI and high-level programming languages for ML/AI help to capture this complexity and makes using ML/AI simpler
+ **AI is more than just Machine Learners and Statisticians**, AI is a team sport

---

= The third wave of AI requires integrative CS, from SoftEng and DBMS, over ML and AI, to computational CogSci



A lot left to be done