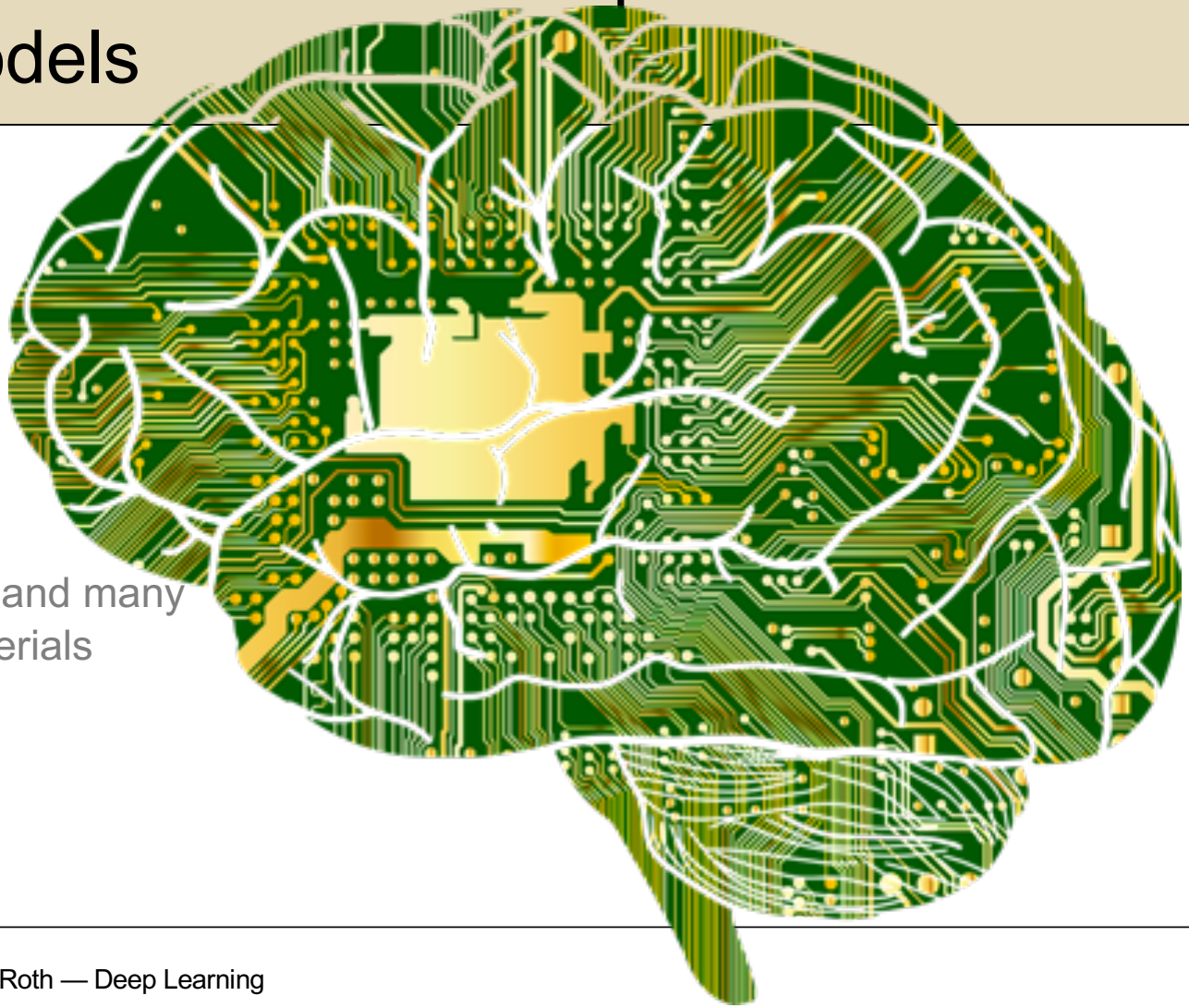
Deep Learning

TECHNISCHE UNIVERSITÄT DARMSTADT

Architectures and Methods: Deep Probabilistic Models



Thanks to Pedro Domingos and many others for making their materials publically available.

Judea Pearl received 2012 the ACM Turing Award 2012 for his work on graphical models

Goals

Get in touch with

1. one of the most exciting topics in AI and ML in recent years, namely **Graphical Models (GMs),** and their basics for **inference**, and
2. tools for making GM „tractable“

Caution! Necessarily incomplete!

Judea Pearl received 2012 the ACM Turing Award 2012 for his work on graphical models

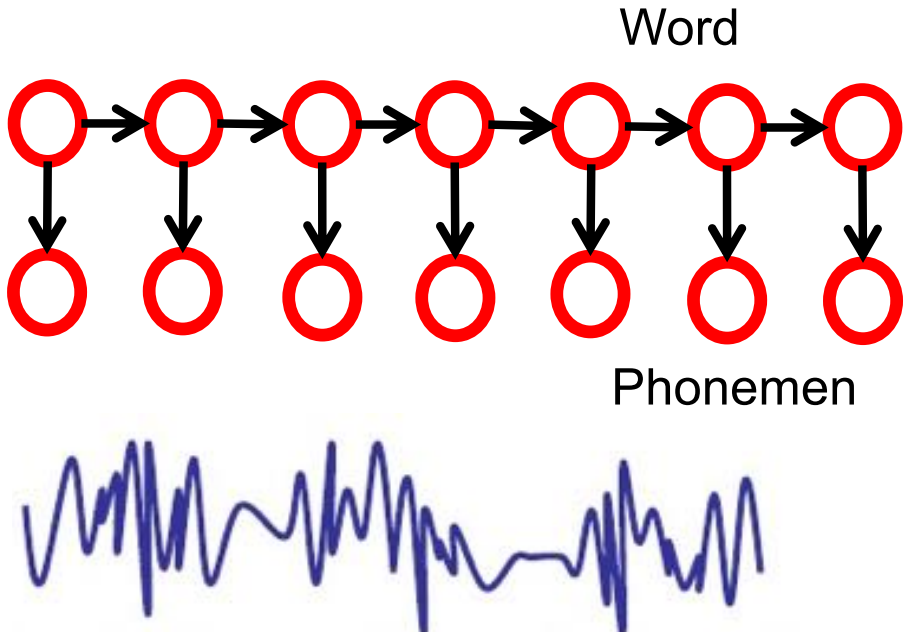
Roadmap of the course

1 Basics of graphical models

2a Exploiting symmetries for inference

2b Sum-product networks

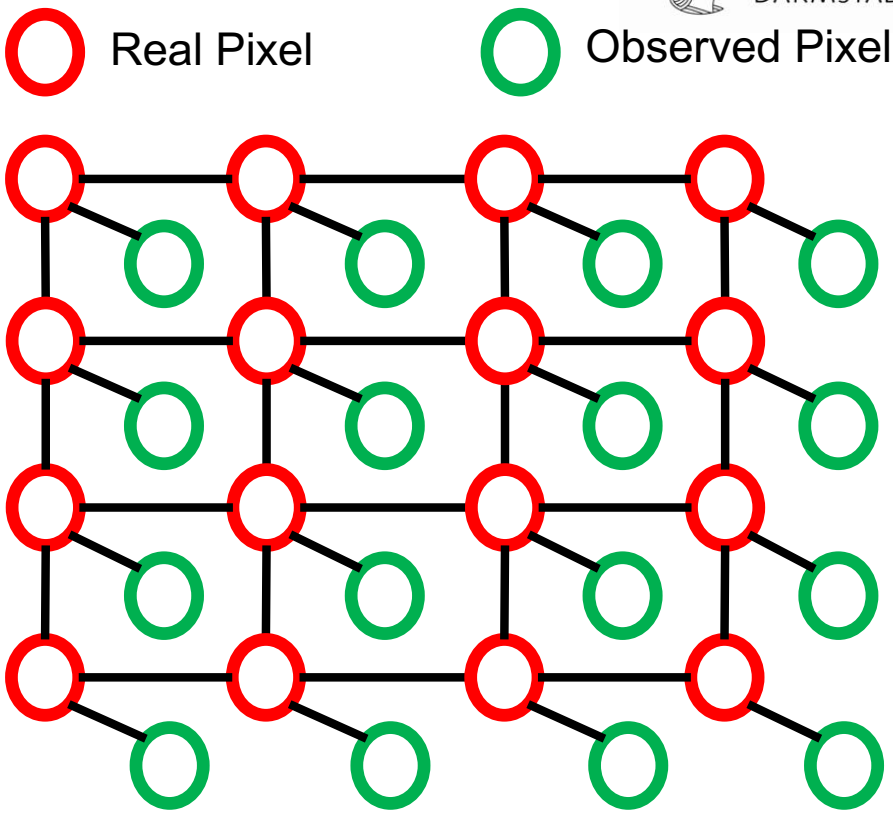
Application: Speech recognition



„He ate the cookies on the couch“

Infer the words from spoken language:
Hidden Markov Model

Application: Image Denoising



Infer the original image from the noisy observed one: Markov networks

Graphical models are omnipresent

TECHNISCHE UNIVERSITÄT DARMSTADT

Information retrivel, search, collaborative filtering, gene expression analysis, natural language processing, bioinformatics, and many,

many,

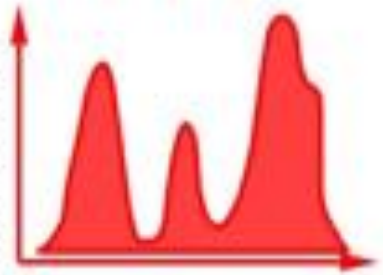
many,

many

more!

OK, so what are probabilities and graphical models?

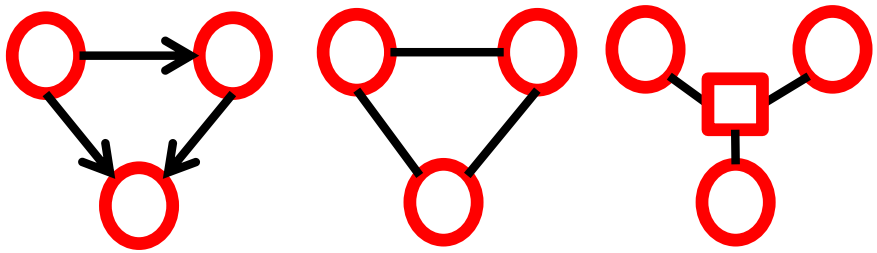
In a Nutshell, Graphical Models are …



Probability

… a graphical notation for (conditional) independency assumptions and therefore a (hopefully) compact specification of probability distributions

Nodes=
Random Variables (RVS)
Edges=
Dependencies among RVs



Bayesian Networks

Markov Networks

Factor Graphs

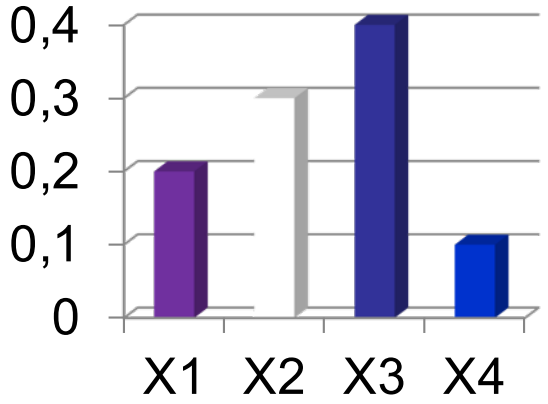
Discrete Random Variables

TECHNISCHE UNIVERSITÄT DARMSTADT

Finite set X of possible states

$$X \in \{x_1, x_2, x_3, \ldots, x_n\}$$

$$P(x_i) \geq 0 \quad \sum_{i=1}^{n} P(x_i) = 1$$



OK, but answering the question requires the joint distribution

What is the probability that smoking causes cancer?

Smoking

no	few	many
0.800	0.150	0.050

Cancer

no	benigne	maligne
0.935	0.046	0.019

Joint Distribution

TECHNISCHE UNIVERSITÄT DARMSTADT

Probability that X=x and Y=y are “true”

$$P(x, y) \equiv P(X = x \wedge Y = y)$$

Cancer

Smoking	no	benigne	maligne
no	0.768	0.024	0.008
few	0.132	0.012	0.006
many	0.035	0.010	0.005

The joint distribution allows us to answer any question! But how?

Make use of basic probability theory

TECHNISCHE UNIVERSITÄT DARMSTADT

Marginalization

$$P(Y) = \sum_{i=1}^{n} P(Y, x_i)$$

sum →

Cancer

Smoking	No	Benigne	Maligne	TOTAL
No	0.768	0.024	0.008	0.800
few	0.132	0.012	0.006	0.150
many	0.035	0.010	0.005	0.050
TOTAL	0.935	0.046	0.019	

P(smoking)

P(cancer)

Product rule & **conditional probability**

$$P(X,Y) = \boxed{P(X \mid Y)} P(Y) = P(Y \mid X) P(X)$$

Probability that X=x if we have observed Y=y (P(y)>0)

Probably the most important rule: Bayes

TECHNISCHE UNIVERSITÄT DARMSTADT

$$P(R,K) = P(R \mid K)P(K) = P(K \mid R)P(R)$$

$$P(R \mid K) = \frac{P(K \mid R)P(R)}{P(K)} = \frac{P(K,R)}{P(K)}$$

cancer

smoking	no	benigne	maligne
no	0.768	0.024	0.008
few	0.132	0.012	0.006
many	0.035	0.010	0.005
TOTAL	0.935	0.046	0.019

P(cancer)

Since we know already P(R,K) und auch P(K), just divide them.

Probably the most important rule: Bayes

TECHNISCHE UNIVERSITÄT DARMSTADT

$$P(R,K) = P(R \mid K)P(K) = P(K \mid R)P(R)$$

$$P(R \mid K) = \frac{P(K \mid R)P(R)}{P(K)} = \frac{P(K,R)}{P(K)}$$

cancer

smoking	no	benigne	maligne
no	0.768/0.935	0.024/ 0.46	0.008/ 0.019
few	0.132/0.935	0.012/ 0.46	0.006/ 0.019
many	0.035/0.935	0.010/ 0.46	0.005/ 0.019
TOTAL	0.935	0.046	0.019

P(cancer)

Probably the most important rule: Bayes

TECHNISCHE UNIVERSITÄT DARMSTADT

$$P(R,K) = P(R \mid K)P(K) = P(K \mid R)P(R)$$

$$P(R \mid K) = \frac{P(K \mid R)P(R)}{P(K)} = \frac{P(K,R)}{P(K)}$$

cancer= …)

P(smoking|	no	benigne	maligne
no	0.821	0.522	0.421
few	0.141	0.261	0.316
many	0.037	0.217	0.263

As long as all entries are >0, everything can be computed! Mission completed?

No!! Our mission has just started

TECHNISCHE UNIVERSITÄT DARMSTADT

Joint distribution is enumerating everything

- Worst-case run time: $O(2^n)$
  - n = # of RVs
- Space is $O(2^n)$ too
  - Size of the table of the joint distribution

Main idea: make use of independencies to compress the representation

Indpendency

TECHNISCHE UNIVERSITÄT DARMSTADT

(Current) age and the gender of a person are independent

$$P(G, A) = P(G) \cdot P(A)$$

$$P(A \mid G) = P(A)$$

$$P(G \mid A) = P(G)$$

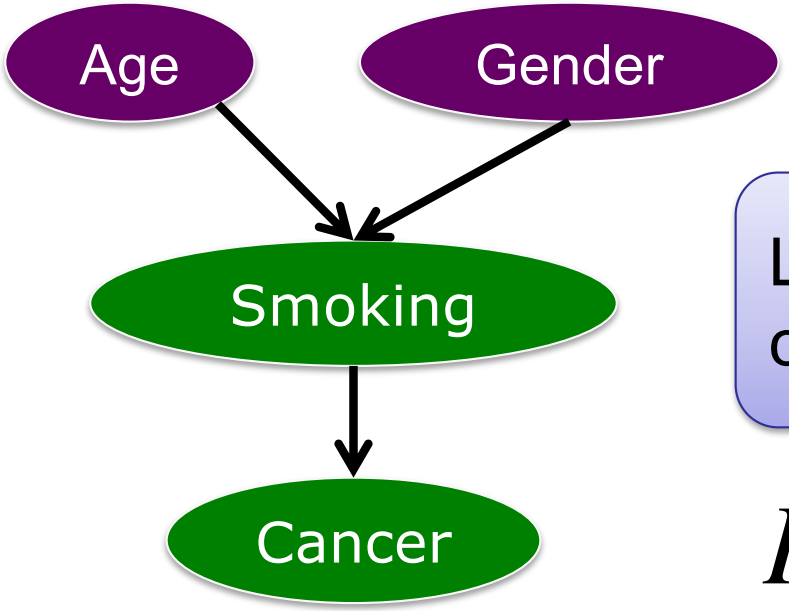
You would not give me money for information on the gender to know the age of a person!

Conditional Independence

TECHNISCHE UNIVERSITÄT DARMSTADT

Cance is independent of age and gender, if the person smokes.

If you have not observed anything,age and gender are independent.



Less entries, therefore lower complexity

$$P(C \mid S, G, A) = P(C \mid S)$$

Bayesian Networks [Pearl 1989]

TECHNISCHE UNIVERSITÄT DARMSTADT

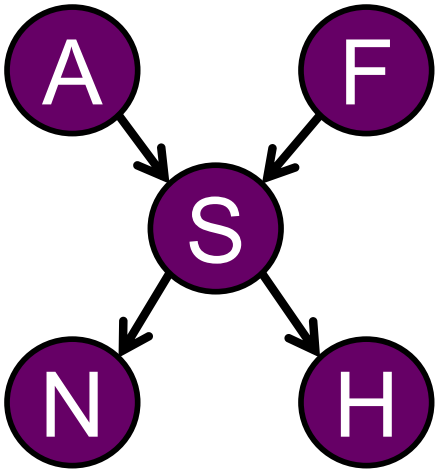
Set of random variables $\{X_1, \ldots, X_n\}$

Directed, acyclic graph (DAG)

To each RV Xi we associate the

conditional probabilitiy distribution:

$$P\left(X_i \mid \mathrm{Pa}\left(X_i\right)\right)$$



The **joint distribution** is

$$P\left(X_1, \ldots, X_n\right) = \prod_{i=1}^{n} P\left(X_i \mid \mathrm{Pa}\left(X_i\right)\right)$$

BN semantics

Local Markov Assumption

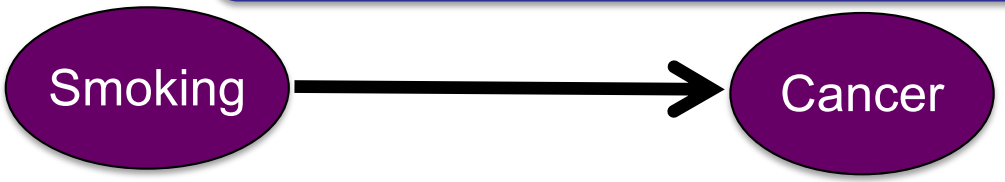
Each RV X is independent of ist „non-descendant“ given ist parents (Xi ⊥ nonDescendants| Pa_{Xi})

Example

TECHNISCHE UNIVERSITÄT DARMSTADT

But how do we do inference?

$R \in \{no, few, many\}$



P(S=n)	0.80
P(S=f)	0.15
P(S=m)	0.05

$K \in \{no, benigne, maligne\}$

Smoking=	n	f	m
P(C=n)	0.96	0.88	0.60
P(C=b)	0.03	0.08	0.25
P(C=m)	0.01	0.04	0.15

What is Inference?

TECHNISCHE UNIVERSITÄT DARMSTADT

Query: $P(X \mid e)$

Definiton of conditional probability $P(X \mid e) = \frac{P(X, e)}{P(e)}$

Up to normalization $P(X \mid e) \propto P(X, e)$

Hence, this rewrites to

$$P(\mathbf{Y}) = \sum_{X_i \notin \mathbf{Y}} \prod_{i=1}^{n} P(X_i \mid \mathrm{Pa}(X_i))$$

BN semantics

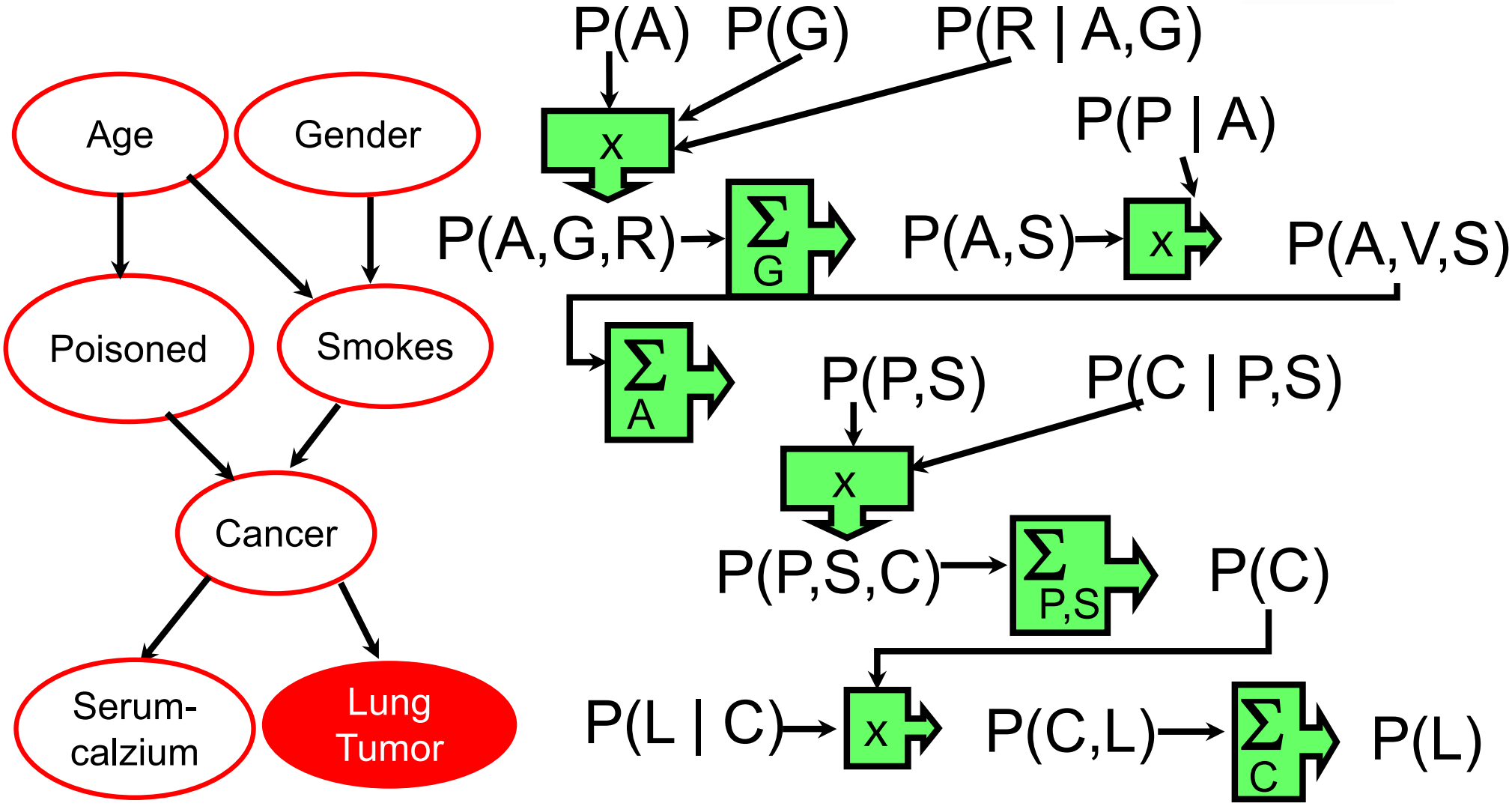
Marginalization

Main observation: Σ and ∏ commute

$\Sigma_a(P_1 \times P_2) = (\Sigma_a P_1) \times P_2$ if A is not in P_2

Complete example: Let's comput P(L)

elimination order is : G,A,V,R,K



Exponentiall in the size of the largest (induced) factor (table) also called treewidth: 2^3 vs 2^7

As an algorithm, this is called: Variable elimination

TECHNISCHE UNIVERSITÄT DARMSTADT

Given a BN and a query P(X|e) / P(X,e)

Instantiate evidence **e**

Choose an elimination order over the variables, e.g., $X_1, \ldots, X_n$

Initial *factors* $\{f_1,\ldots,f_n\}$: $f_i = P(X_i|\mathbf{Pa}_{Xi})$ (CPT for X_i)

For i = 1 to n, if $X_i \notin \{X,\mathbf{E}\}$

- Collect factors $f_1,\ldots,f_k$ that include X_i
- Generate a new factor by eliminating X_i from these factors

$$g = \sum_{X_i} \prod_{j=1}^{k} f_j$$

- Variable X_i has been eliminated! Add g to the set of factors

Normalize P(X,**e**) to obtain P(X|**e**)

What have we learned so far?

TECHNISCHE UNIVERSITÄT DARMSTADT

Uncertainty is omnipresent

Uncertainty can be captured using probability distributions

Graphical models are compact encodings of probability distributions

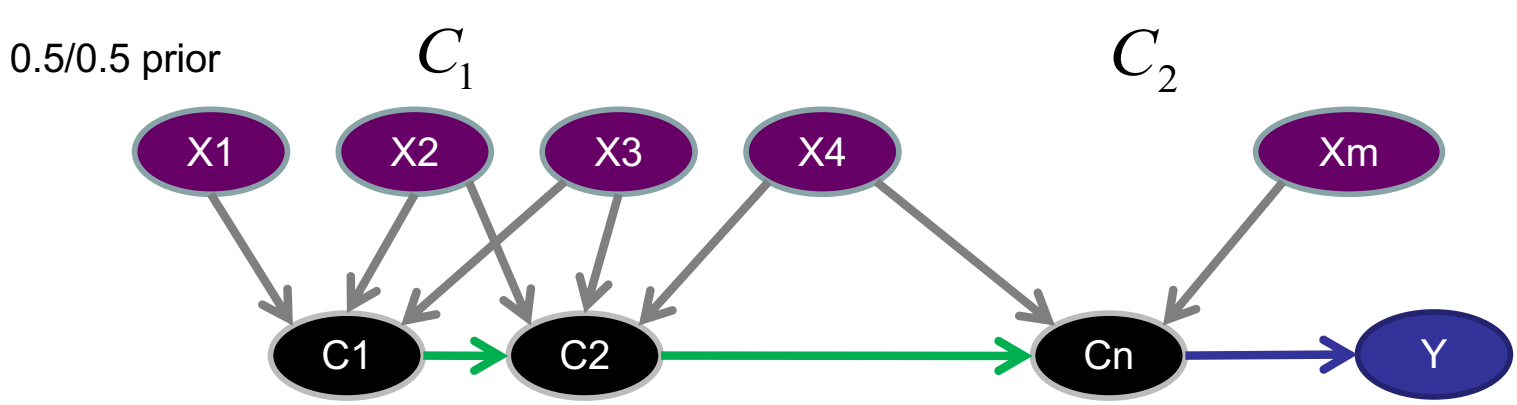
They lead to „efficient“ algorithms for inference such as Variablen-Elimination

Mission Completed? No …

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem: Inference (even approximate) in Bayesian networks is NP-hard (#P; via reduction to 3-SAT)

$$\underbrace{(\overline{X}_1 \vee X_2 \vee X_3)}_{C_1} \wedge \underbrace{(\overline{X}_2 \vee X_3 \vee X_4)}_{C_2} \wedge \ldots \underbrace{\quad}_{C_n}$$



P(Y=t)>0 iff 3-SAT formula is satisfiable

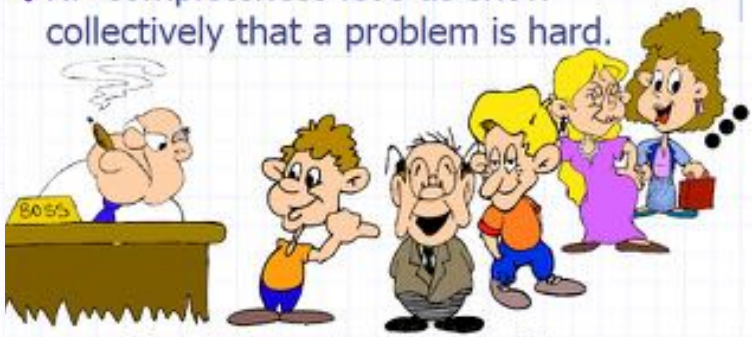
◆What to do when we find a problem that looks hard...

I couldn't find a polynomial-time algorithm;
I guess I'm too dumb.

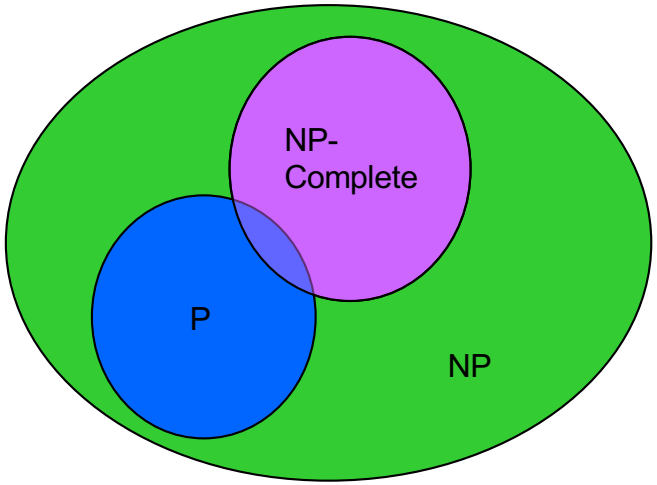
◆Sometimes we can prove a strong lower bound... (but not usually)

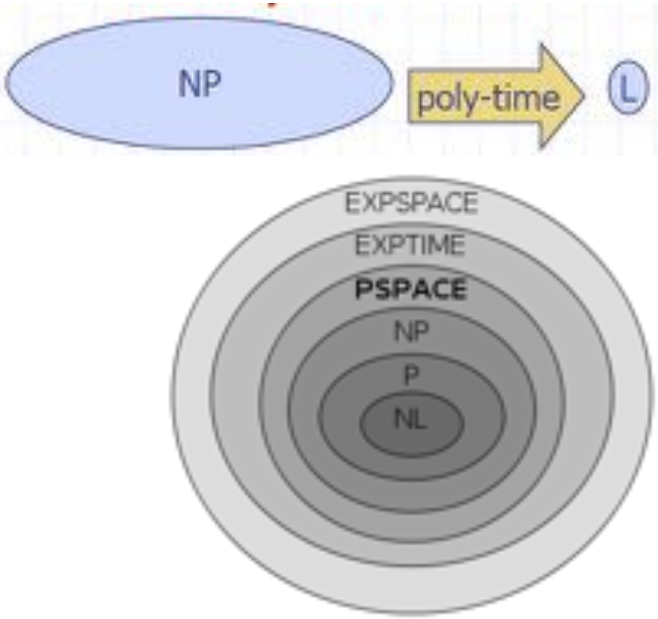
I couldn't find a polynomial-time algorithm,
because no such algorithm exists!

◆NP-completeness let's us show collectively that a problem is hard.



I couldn't find a polynomial-time algorithm,
but neither could all these other smart people.





What have we learnt about Bayesian networks?

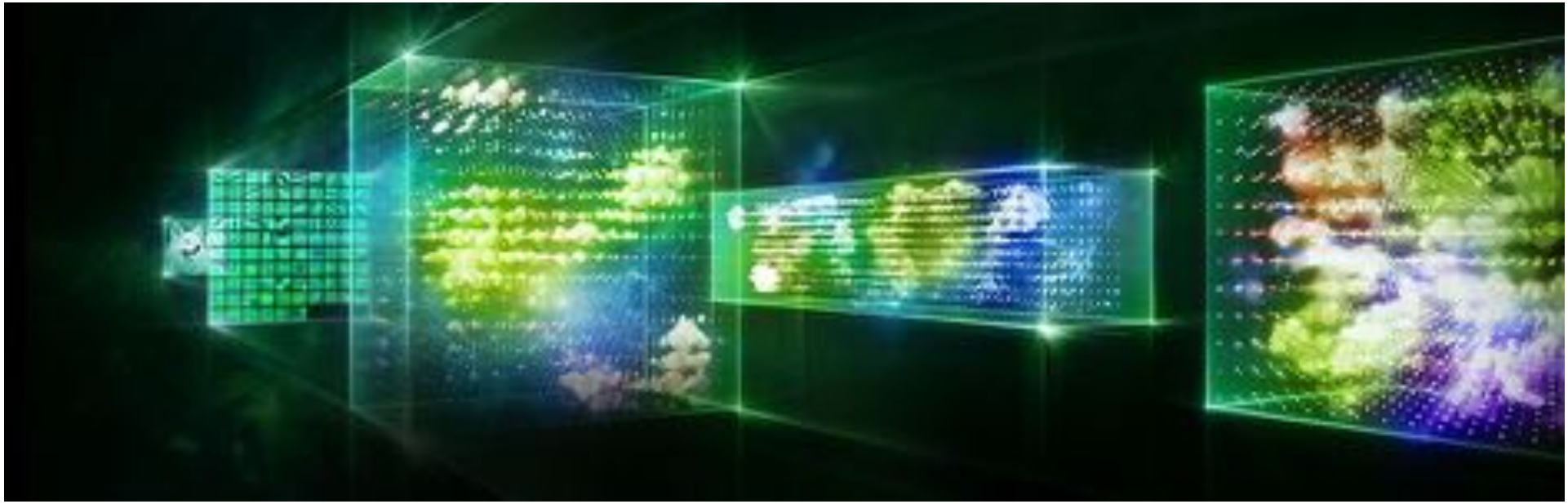
TECHNISCHE UNIVERSITÄT DARMSTADT

- Bayesian networks (BNs) encode joint distributions
- They are DAGs

  (nodes = RVs, edges = dependencies)
- Inference is NP-hard
- Variable Elimination is one of the most basic inference approaches; there are many other inference approaches
- We have skipped learning BNs

TECHNISCHE UNIVERSITÄT DARMSTADT

Can we borrow ideas from deep learning for graphical models? And if so, are there advantages?



There is another major trend in AI/ML

DEEP LEARNING

Deep Learning

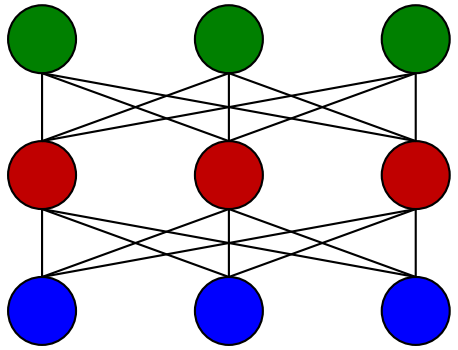
TECHNISCHE UNIVERSITÄT DARMSTADT

Stack many layers

E.g.: DBN [Hinton & Salakhutdinov,2006]

CDBN [Lee et al.,2009]

DBM [Salakhutdinov & Hinton,2010]



Potentially much more powerful than shallow architectures, represent computations [Bengio, 2009]

But …

- **Often no probabilistic semantics**
- **Learning requires extensive efforts**

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

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

Alternative Representation: Graphical Models as (Deep) Networks

TECHNISCHE UNIVERSITÄT DARMSTADT

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

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

Shorthand for Indicators

TECHNISCHE UNIVERSITÄT DARMSTADT

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

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

Sum Out Variables

TECHNISCHE UNIVERSITÄT DARMSTADT

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

$$\boxed{e: X_1 = 1}$$

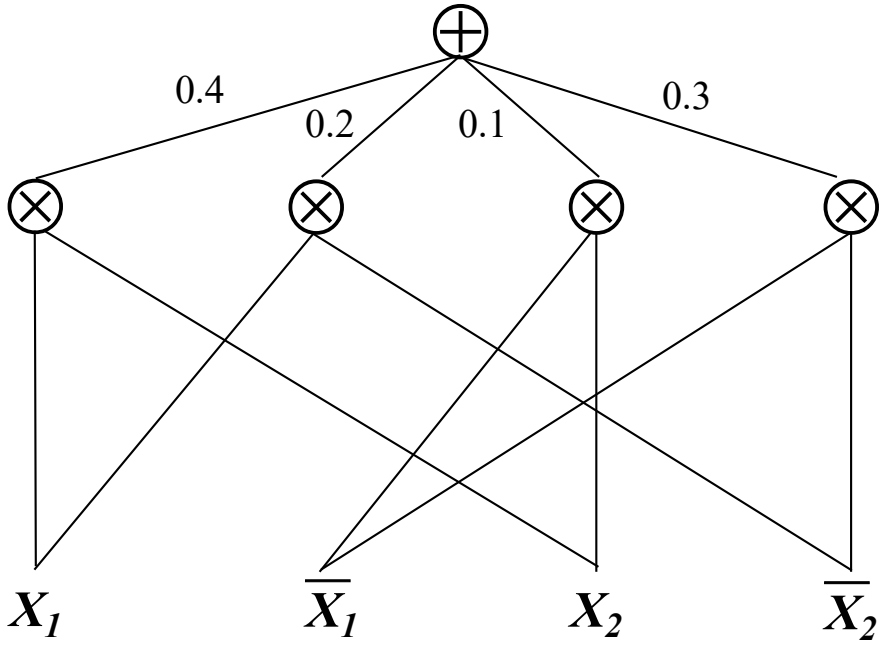
$$P(e) = \mathbf{0.4 \cdot X_1 \cdot X_2} + \mathbf{0.2 \cdot X_1 \cdot \overline{X}_2} + 0.1 \cdot \overline{X}_1 \cdot X_2 + 0.3 \cdot \overline{X}_1 \cdot \overline{X}_2$$

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

Easy: Set both indicators of X2 to 1

Idea: Deeper Network Representation of a Graphical Model that encodes how to compute probabilities

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

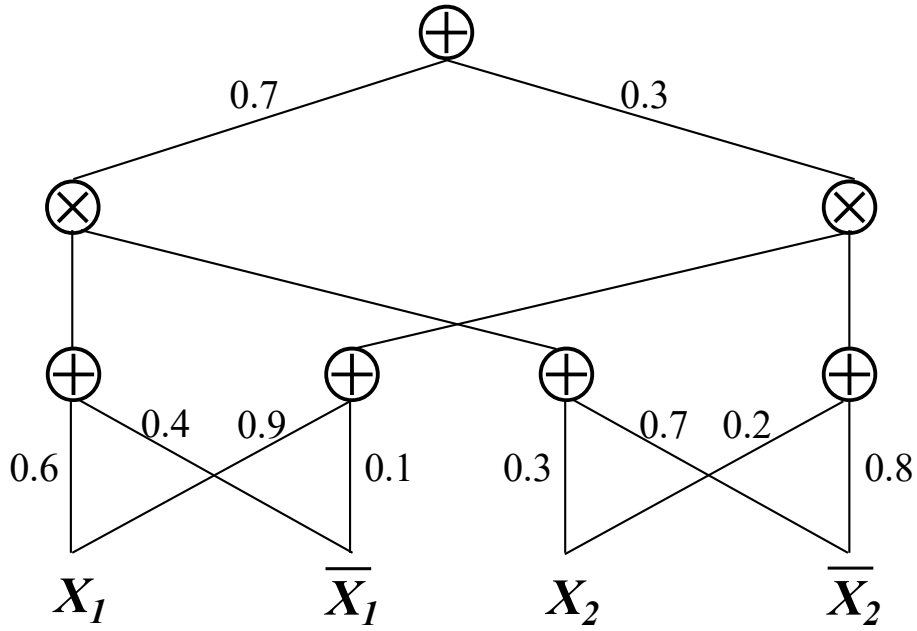


Sum-Product Networks* (SPNs)

[Poon, Domingos UAI 2011]

TECHNISCHE UNIVERSITÄT DARMSTADT

A SPN S is a rooted DAG where:
Nodes: Sum, product, input indicator
Weights on edges from sum to children



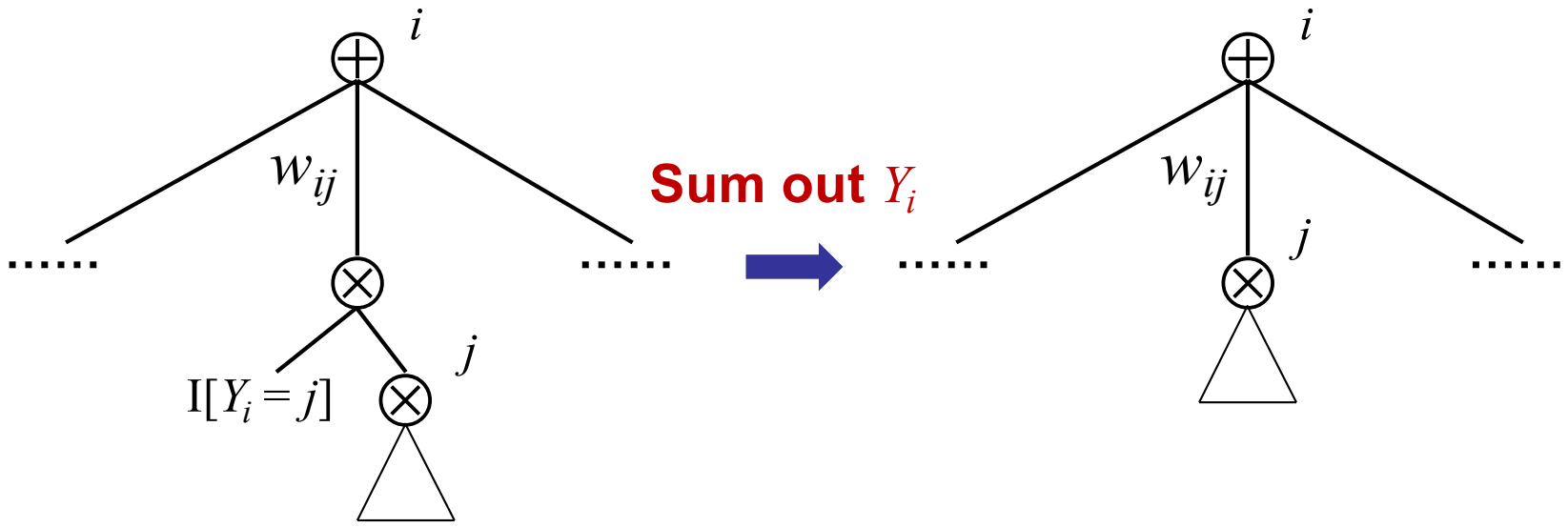
*SPNs are an instance of Arithmetic Circuits (ACs). ACs have been introduced into the AI literature more than15 years ago as a tractable representation of probability distributions

[Darwiche CACM 48(4):608-647 2001]

Semantics of Sums and Products

Product ~ Feature $\rightarrow$ Form feature hierarchy

Sum ~ Mixture (with hidden var. summed out)



Inference: Linear in Size of Network

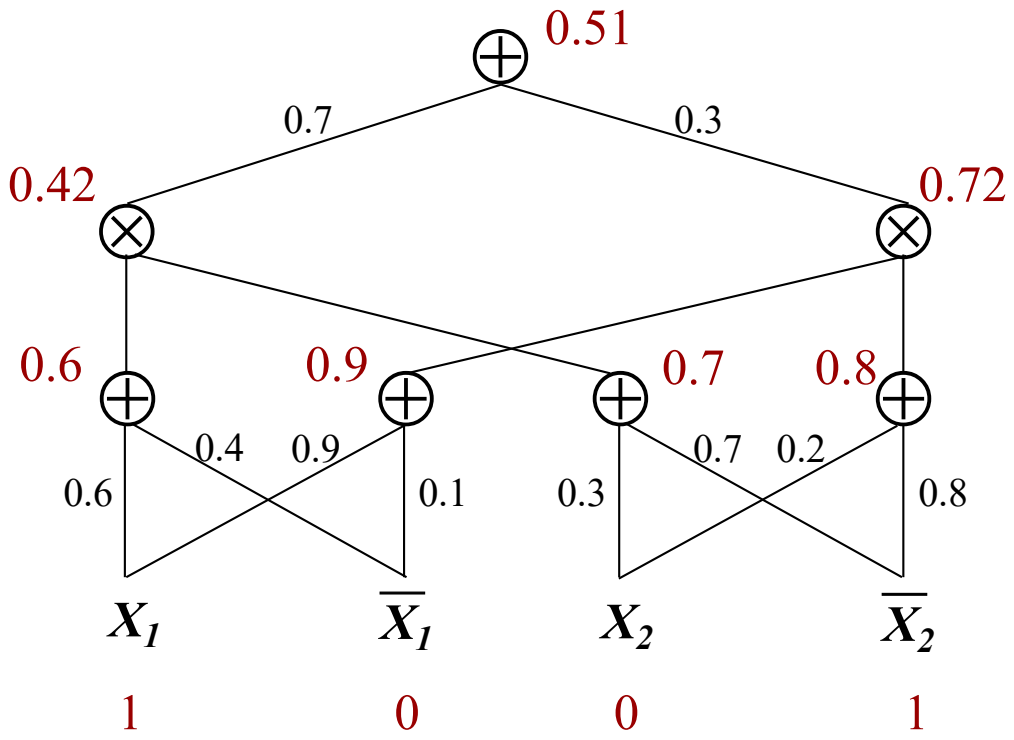
TECHNISCHE UNIVERSITÄT DARMSTADT

As long as weights sum to 1
at each sum node

$$P(X) = S(X)$$

X: $X_1 = 1, X_2 = 0$

X_1	1
$\overline{X}_1$	0
X_2	0
$\overline{X}_2$	1



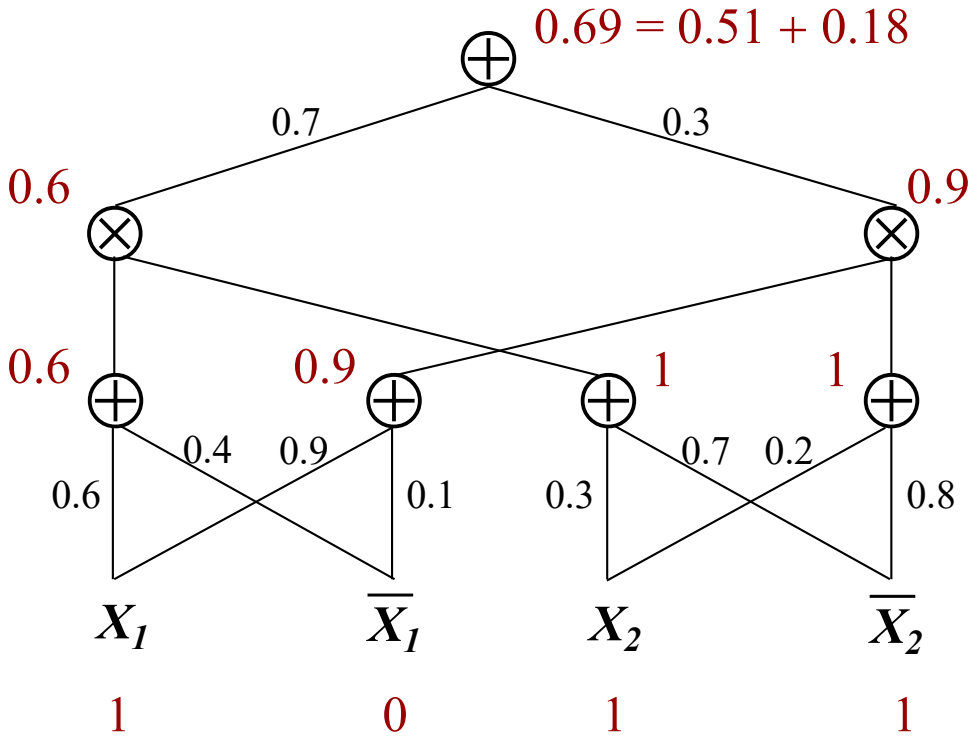
Inference: Linear in Size of Network

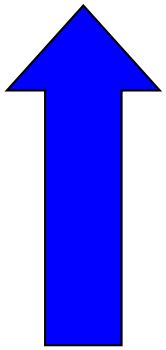
TECHNISCHE UNIVERSITÄT DARMSTADT

Marginal: $P(e) = S(e)$

e: $X_1 = 1$

X_1	1
$\overline{X}_1$	0
X_2	1
$\overline{X}_2$	1





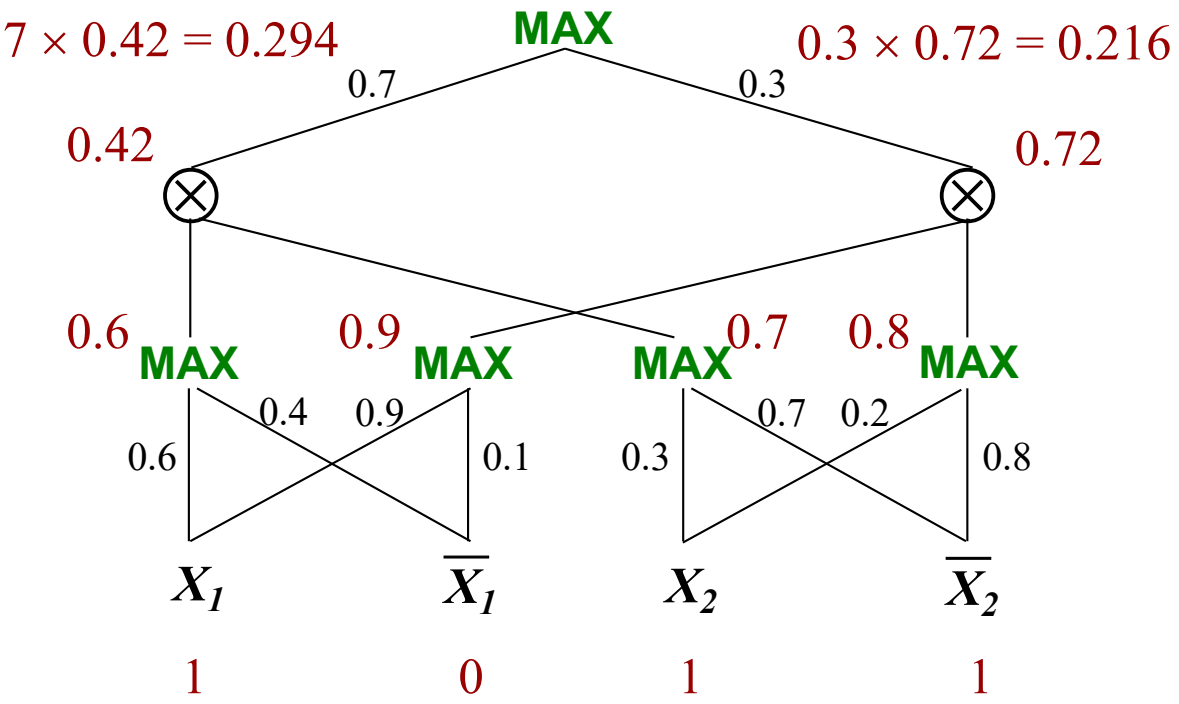
Inference: Linear in Size of Network

TECHNISCHE UNIVERSITÄT DARMSTADT

MAP: Replace sums with maxs

e: $X_1 = 1$

X_1	1
$\overline{X}_1$	0
X_2	1
$\overline{X}_2$	1



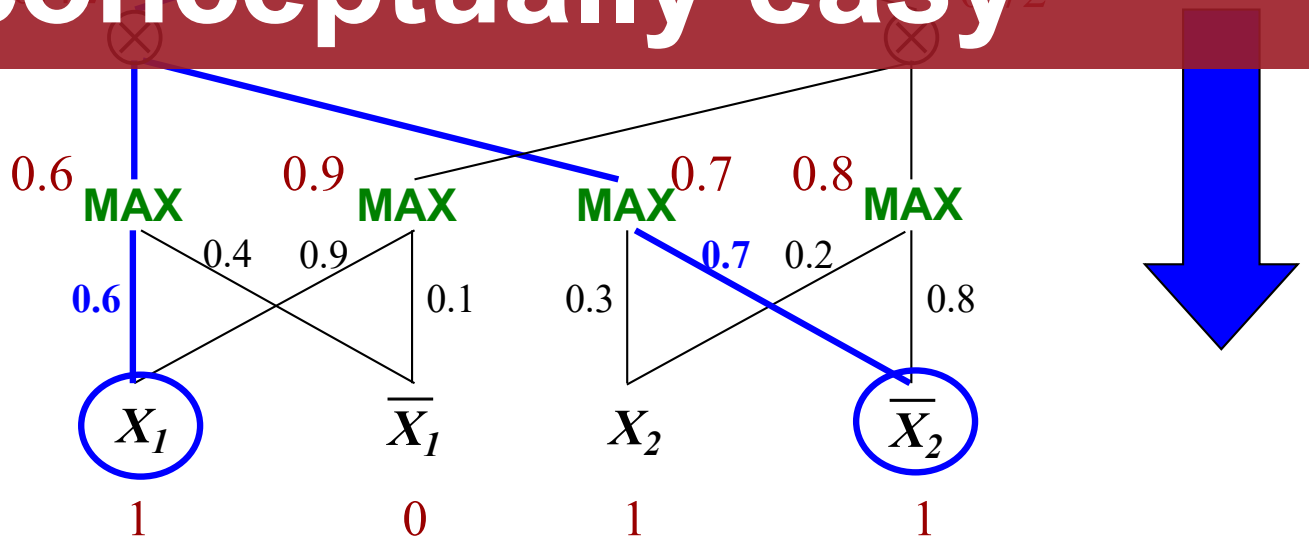
Inference: Linear in Size of Network

MAX: Pick child with highest value

MAP State: $X_1 = 1, X_2 = 0$

And also learning is conceptually easy

X_1	1
$\overline{X}_1$	0
X_2	1
$\overline{X}_2$	1



General Approach via Parameter Estimation assuming a fixed network (like in Deep Learning)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Start with a dense SPN
- Find the structure by (online) learning weights
  Zero weights signify absence of connections
- (Hard) EM beneficial to avoid gradient vanishing
  Each sum node is a mixture over children

In principle you can turn a given SPN into a TensorFlow computation graph and apply any known algorithm from there

Image Completion

TECHNISCHE UNIVERSITÄT DARMSTADT

Main evaluation: Caltech-101 [Fei-Fei et al., 2004]

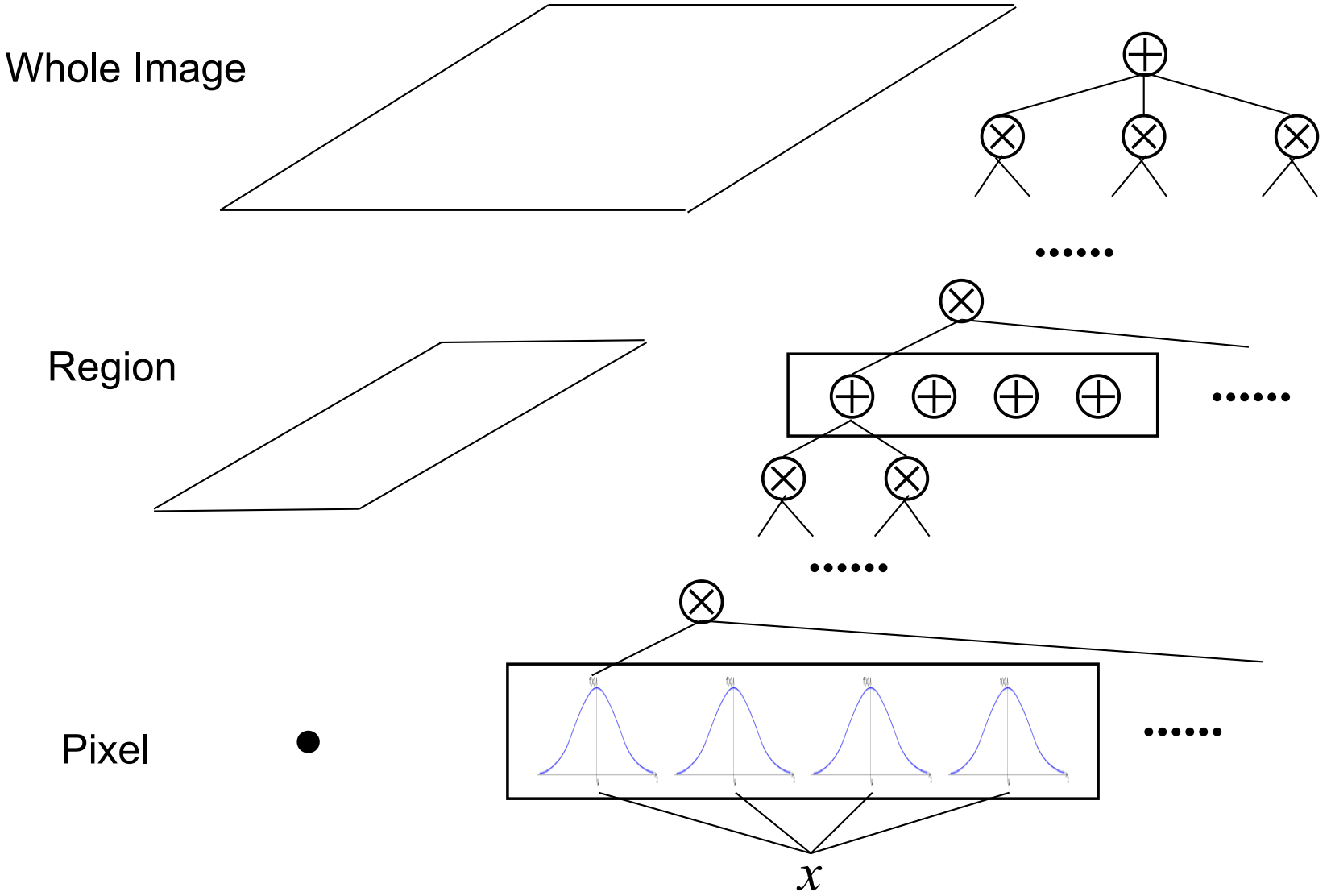
- 101 categories, e.g., faces, cars, elephants
- Each category: 30 – 800 images

Also, Olivetti [Samaria & Harter, 1994] (400 faces)

Each category: Last third for test

Test images: Unseen objects

SPN Architecture



Caltech: Mean-Square Errors

TECHNISCHE UNIVERSITÄT DARMSTADT

	LEFT	BOTTOM
SPN	**3551**	**3270**
DBM	9043	9792
DBN	4778	4492
PCA	4234	4465
Nearest Neighbor	4887	5505

SPN vs. DBM / DBN

TECHNISCHE UNIVERSITÄT DARMSTADT

SPN is order of magnitude faster

	SPN	DBM / DBN
Learning	2-3 hours	Days
Inference	< 1 second	Minutes or hours

No elaborate preprocessing, tuning

Reduced errors by 30-60%

Learned up to 46 layers

Example Completions

TECHNISCHE UNIVERSITÄT DARMSTADT

Original

SPN

DBM

DBN

PCA

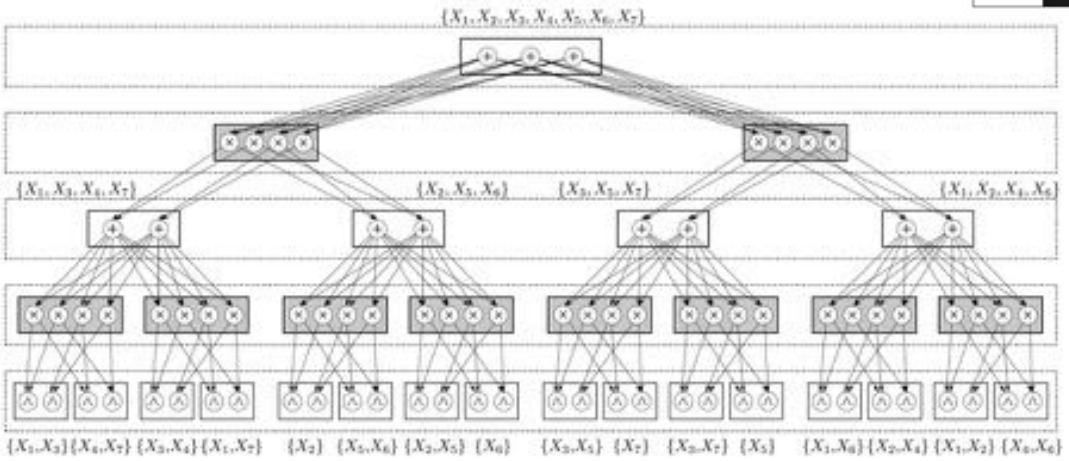
Nearest Neighbor

Random sum-product networks

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

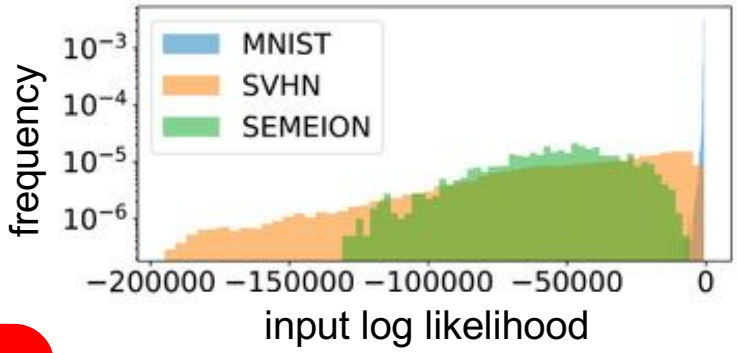
UNIVERSITY OF CAMBRIDGE · Max Planck Institute for Intelligent Systems · TECHNISCHE UNIVERSITÄT DARMSTADT · UBER AI Labs · UNI GRAZ

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



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 (9.38M)	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)



SPNs can have similar predictive performances as (simple) DNNs

SPNs can distinguish the datasets

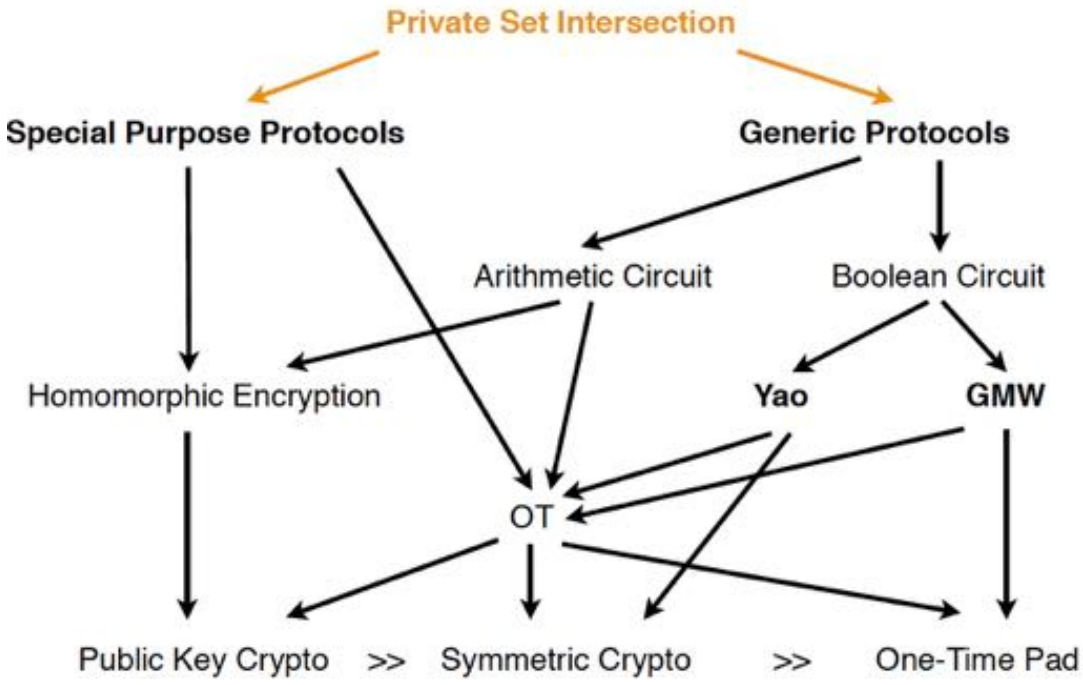
SPNs know when they do not know by design

[Sommer, Oppermann, Molina, Binnig, Kersting, Koch ICDD 2018, Weber, Sommer, Oppermann, Molina, Kersting, Koch FPT 2019]

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			7.87	63090.94	0.27	17249			696.00	**24.44**
Audio	20000	4271.78			5.4			20317			761.00	**26.28**
Netflix	20000	4892.22			4.8			20322			654.00	**30.58**
MSNBC200	388434	15476.05			30.5			388900	19		008.00	**77.56**
MSNBC300	388434	10060.78			41.2			388810	19		933.00	**78.74**
NLTCS	21574	791.80			31.3			21904			566.00	**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	29.65
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?

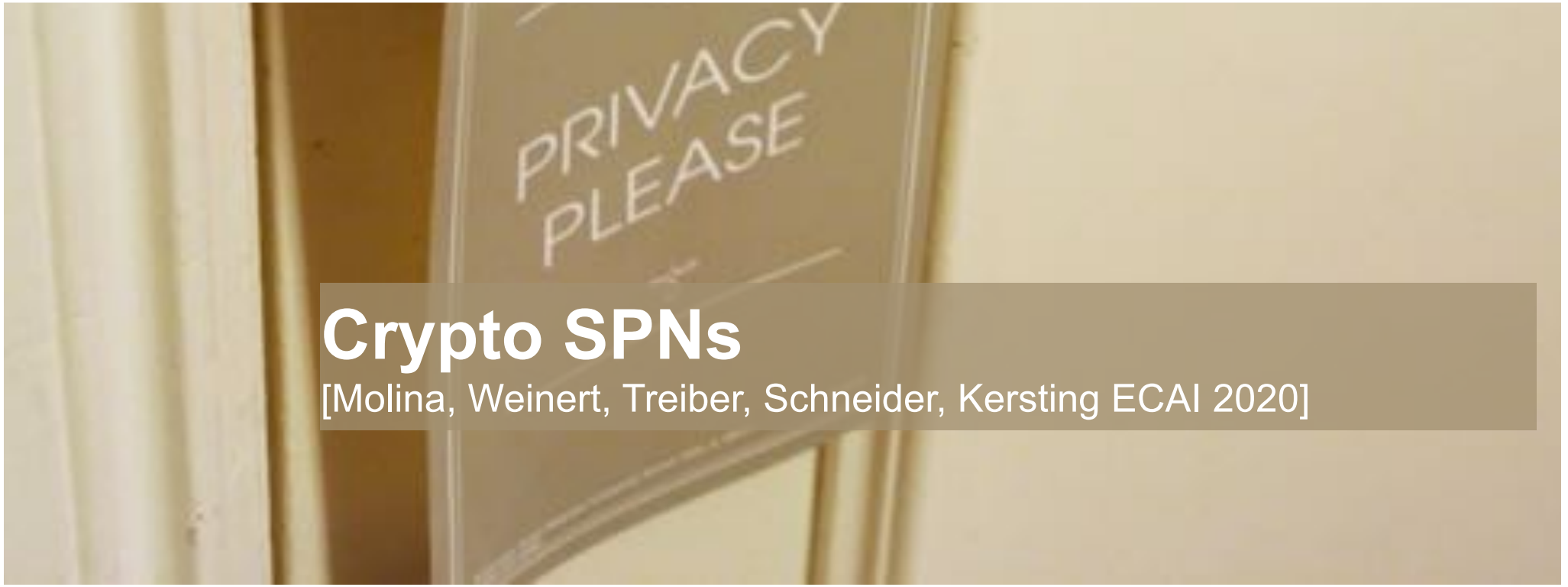


There are generic protocols to validate computations on authenticated data without knowledge of the secret key

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



Crypto SPNs

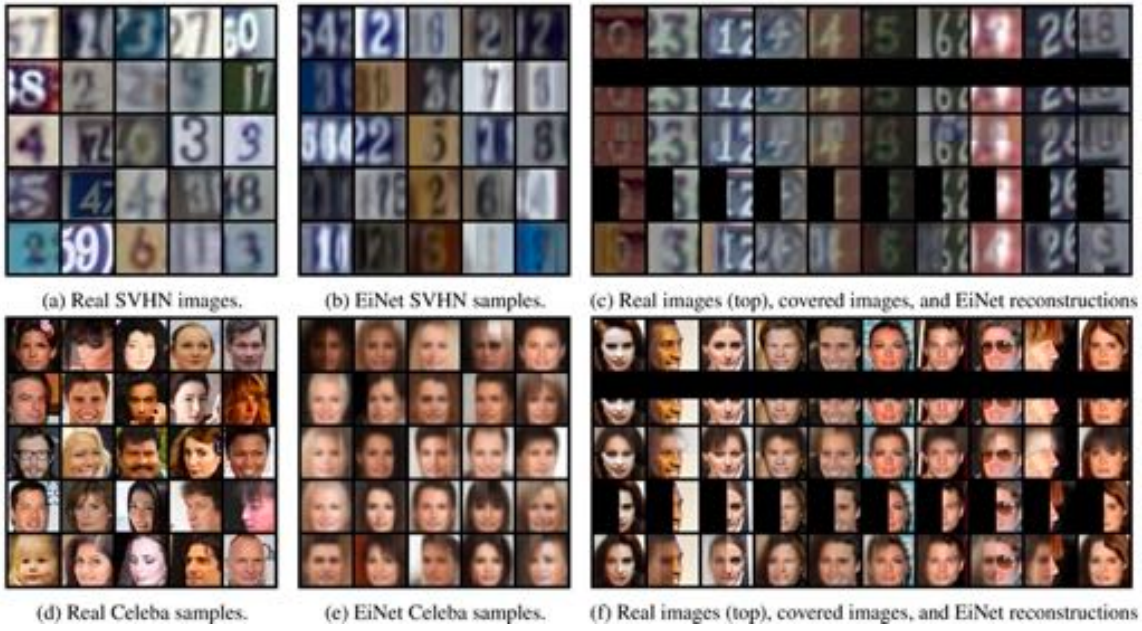
[Molina, Weinert, Treiber, Schneider, Kersting ECAI 2020]

Einsum Networks

[Peharz, Lang, Vergari, Stelzner, Molina, Trapp, Van den Broeck, Kersting, Ghahramani ICML 2020]

UNIVERSITY OF CAMBRIDGE · UBER AI Labs · TU/e · UCLA · TECHNISCHE UNIVERSITÄT DARMSTADT · UNI GRAZ

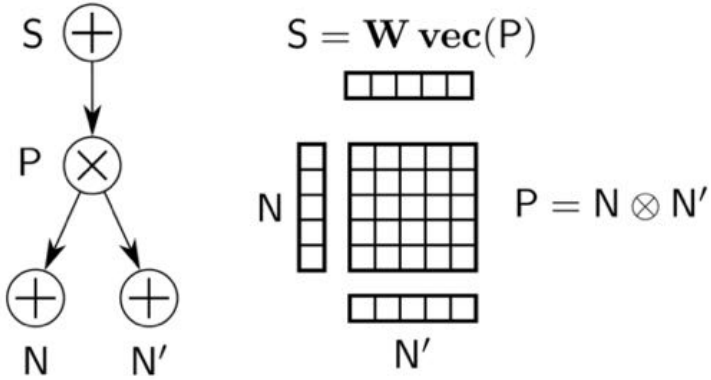
ICML | 2020
Thirty-seventh International Conference on Machine Learning

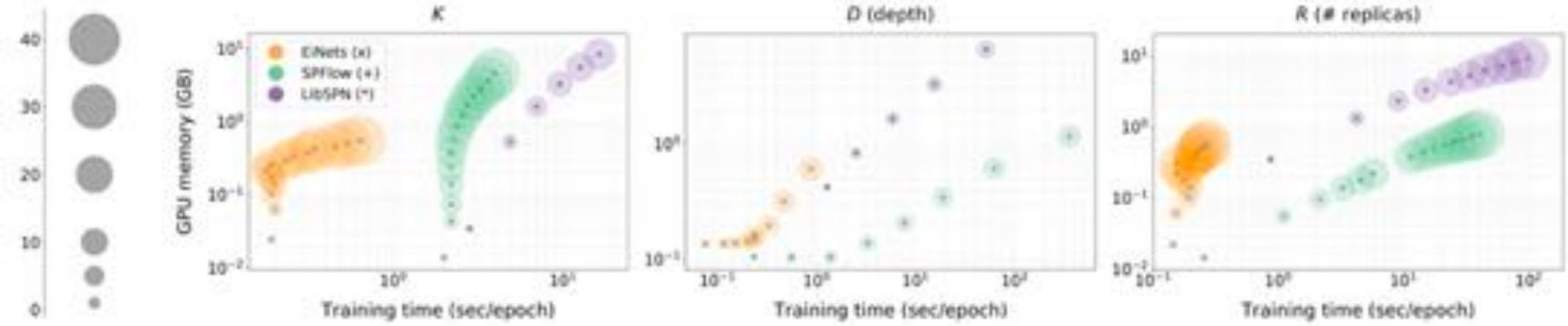


(a) Real SVHN images. (b) EiNet SVHN samples. (c) Real images (top), covered images, and EiNet reconstructions

(d) Real Celeba samples. (e) EiNet Celeba samples. (f) Real images (top), covered images, and EiNet reconstructions

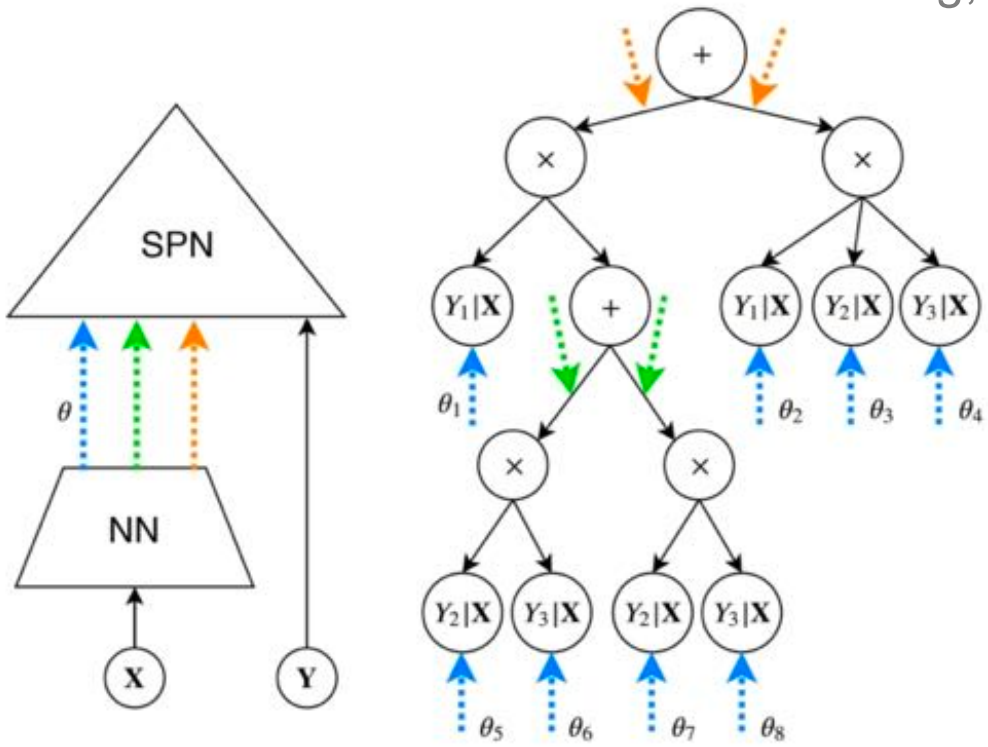
Compilation of PCs into efficient to evaluate computational graphs

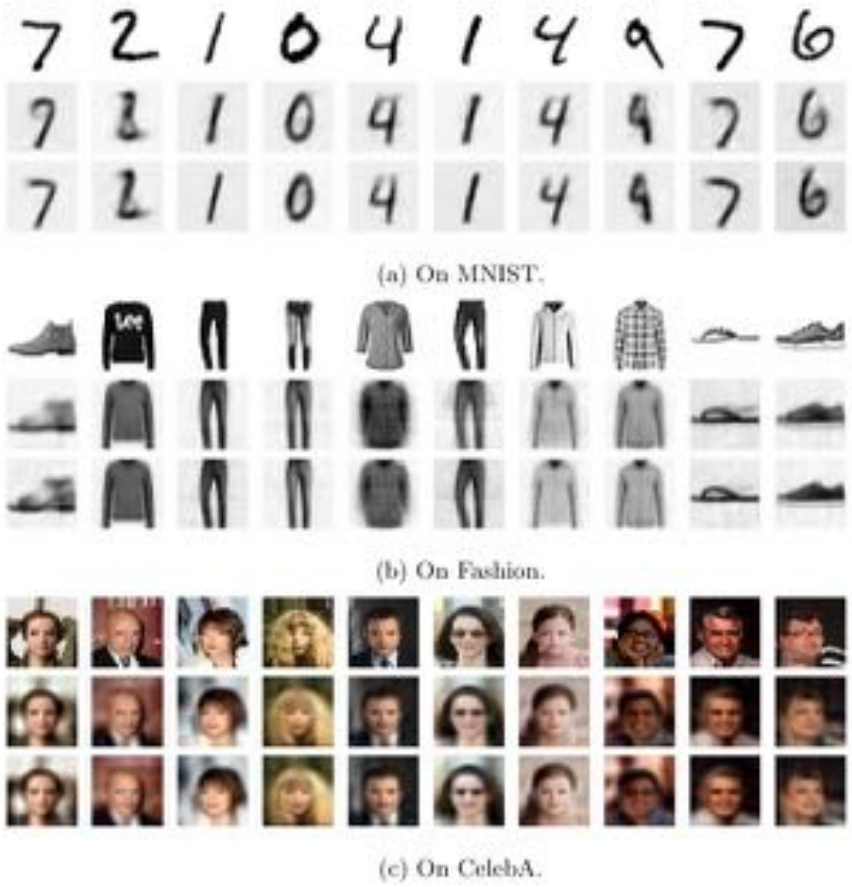




Conditional SPNs

[Shao, Molina, Vergari, Peharz, Liebig, Kersting PGM 2020]





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**

VAE using SPNs (top original, middle Vanilla VAE, bottom SPN-VAE)

SPNs for Time Series

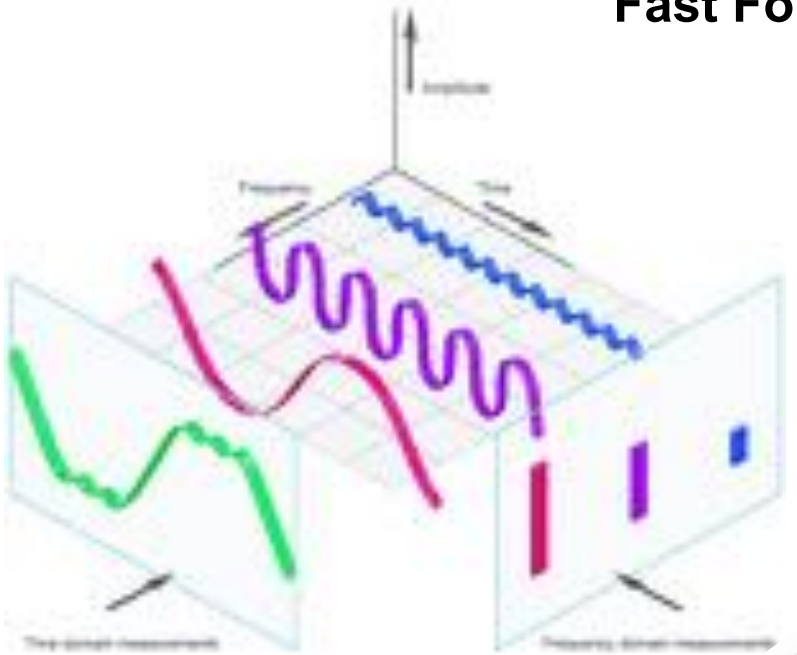
ICML 2021 38th International Conference on Machine Learning

TECHNISCHE UNIVERSITÄT DARMSTADT

TECHNISCHE UNIVERSITÄT DARMSTADT

[Yu, Ventola, Kersting: Whittle Networks: A Deep Likelihood Model for Time Series. ICML 2021]

Fast Fourier Transformation + Complex-values SPN



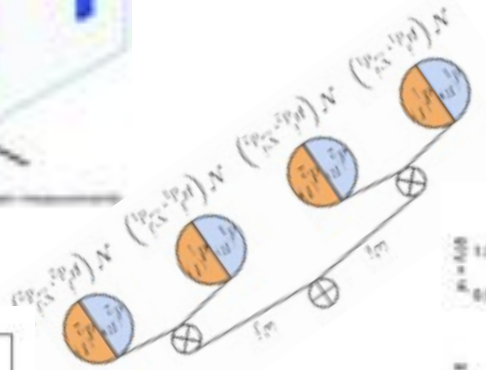
The Fourier coefficients are (assumed) independent complex normal RVs with mean zero $d_{n,k} \sim \mathcal{N}(0, S_k), \quad k = 0, \ldots, T-1$

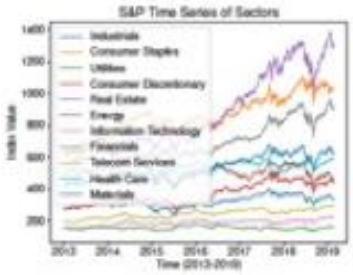
Where the spectral density matrix is independent from the time step

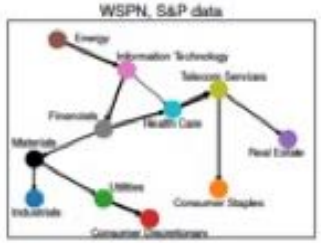
$$S_k = \sum_{h=-\infty}^{\infty} \mathbf{\Gamma}(h) e^{-i\lambda_k h}$$

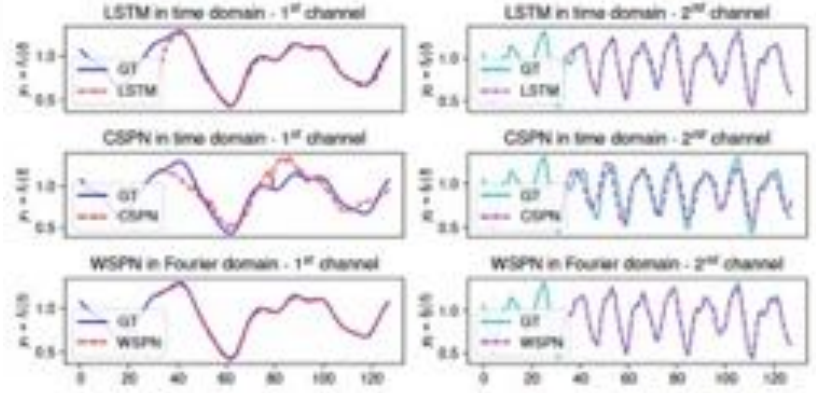
Thus, given N realizations, the Whittle likelihood is

$$p(X_{1:N} \mid S_{0:T-1}) \approx \prod_{n=1}^{N} \prod_{k=0}^{T-1} \frac{1}{\pi^p |S_k|} e^{-d_{n,k}^* S_k^{-1} d_{n,k}}$$





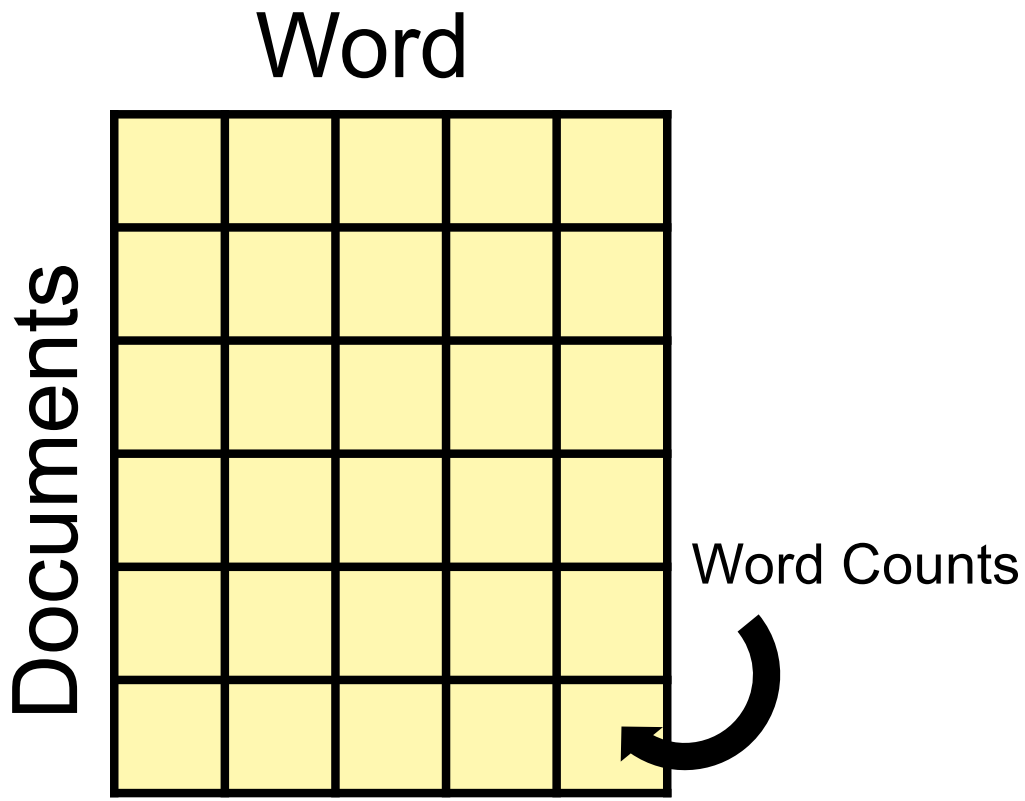




Or we learn directly (Tree-)SPNs

TECHNISCHE UNIVERSITÄT DARMSTADT

Testing independence of
Poisson random variables



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

Or we learn directly (Tree-)SPNs

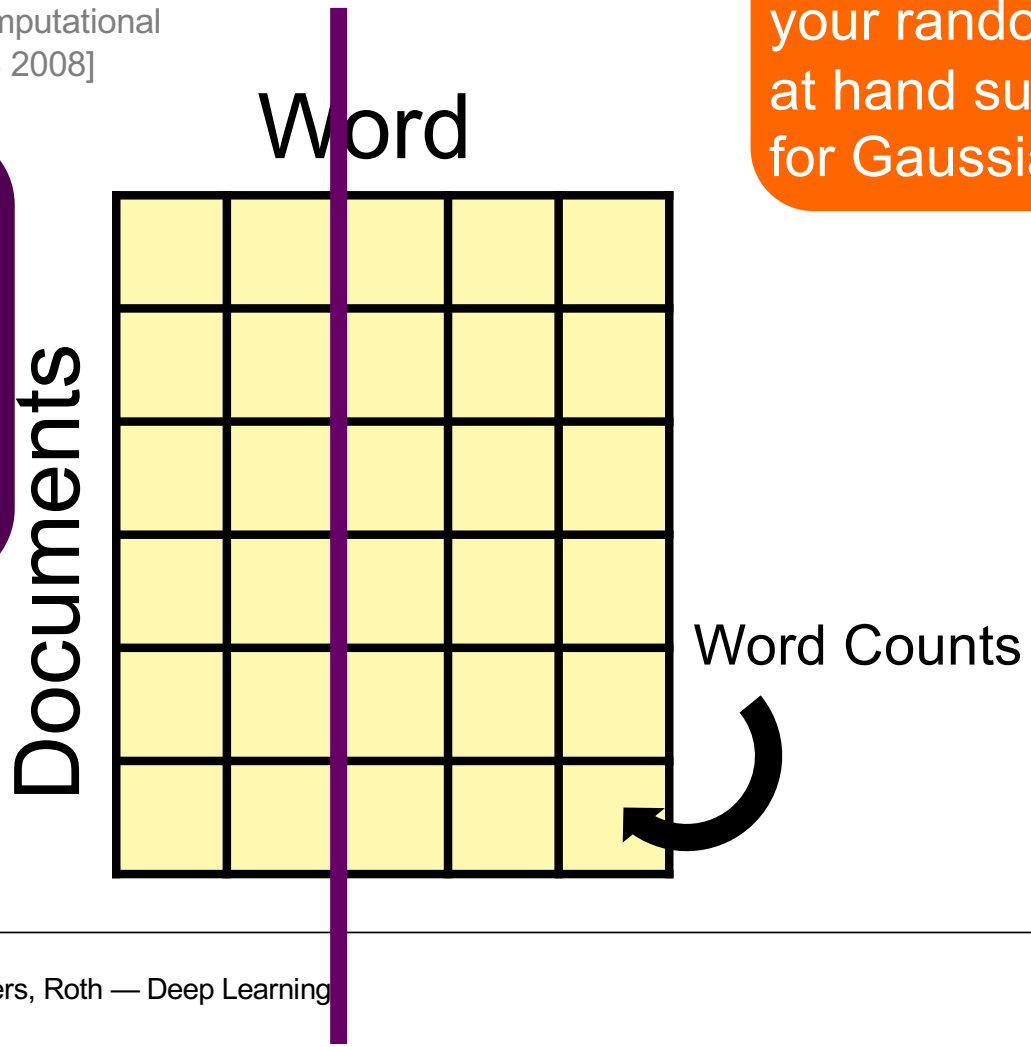
TECHNISCHE UNIVERSITÄT DARMSTADT

Testing independence of Poisson random variables

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

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

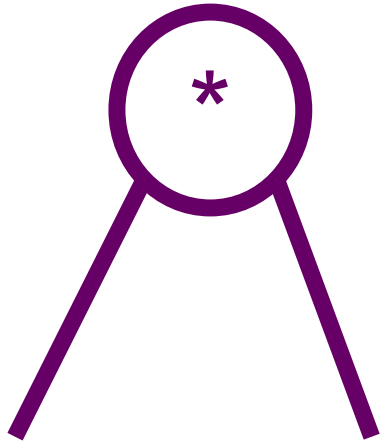
Learn Poisson model trees for P(x|V-x) and P(y|V-y). Check whether X resp. Y is significant in P(y|V-x) resp. P(x|V-y)



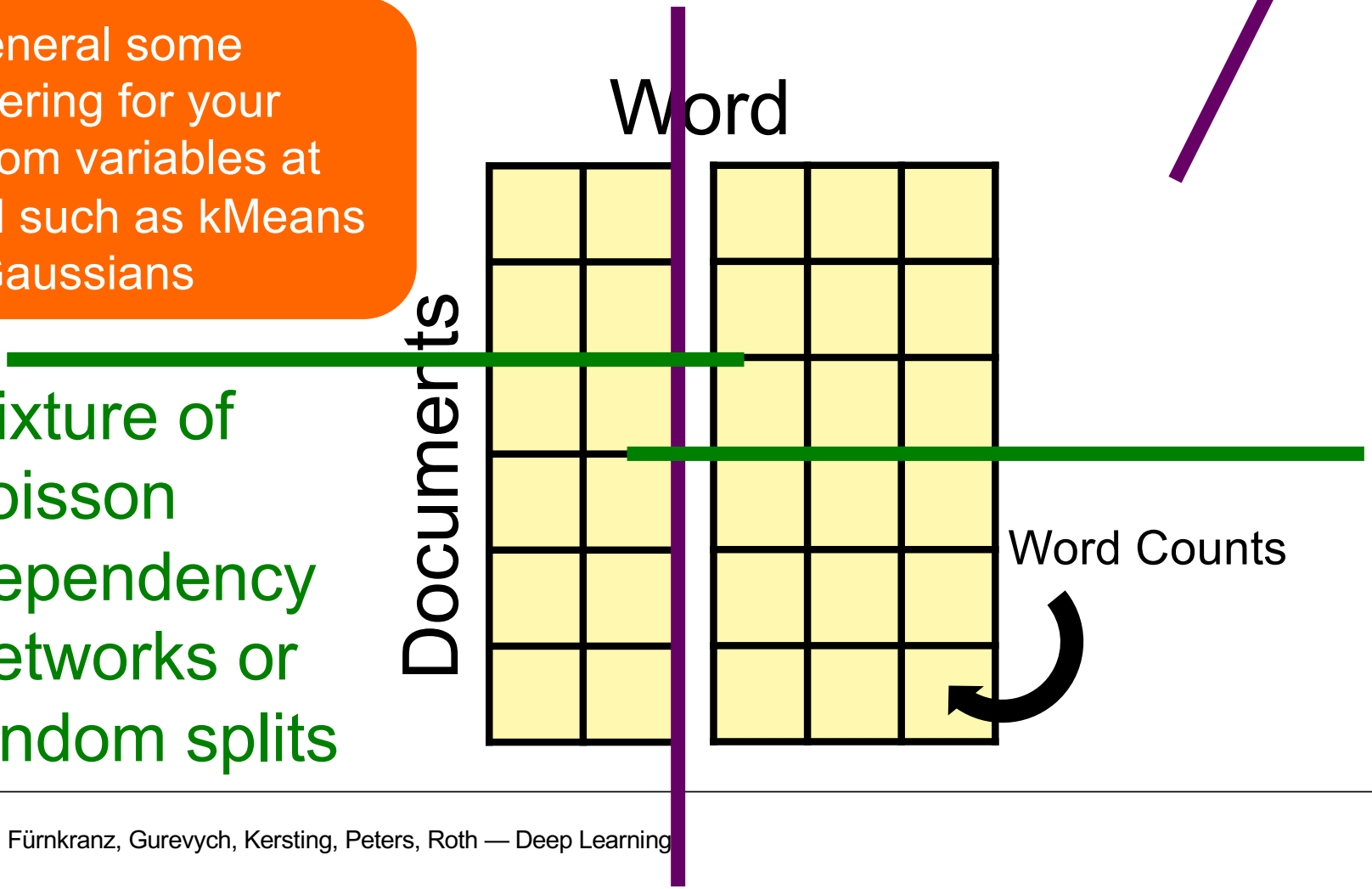
Or we learn directly (Tree-)SPNs

TECHNISCHE UNIVERSITÄT DARMSTADT

Testing independence of Poisson random variables



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



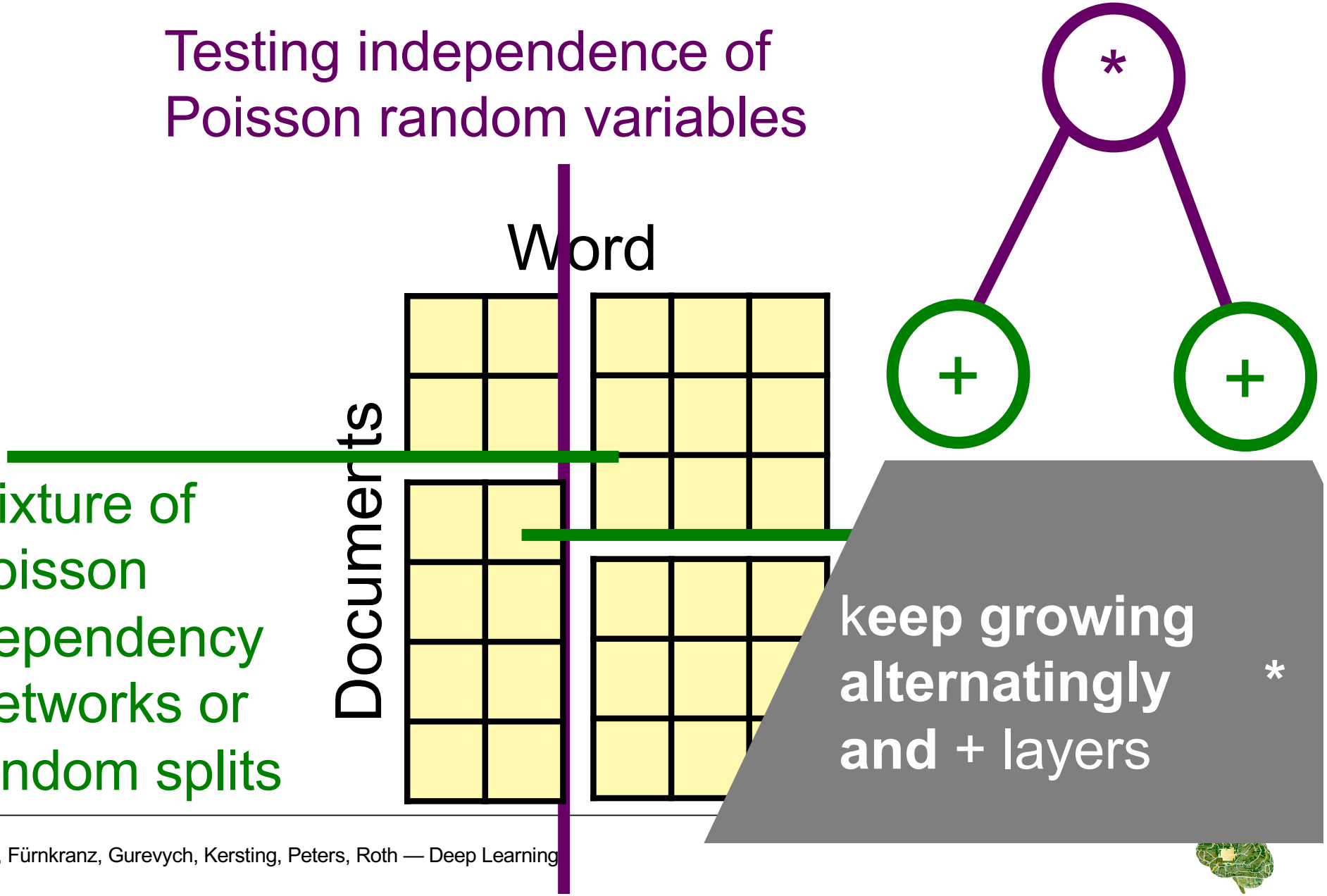
Mixture of Poisson Dependency Networks or random splits

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

Or we learn directly (Tree-)SPNs

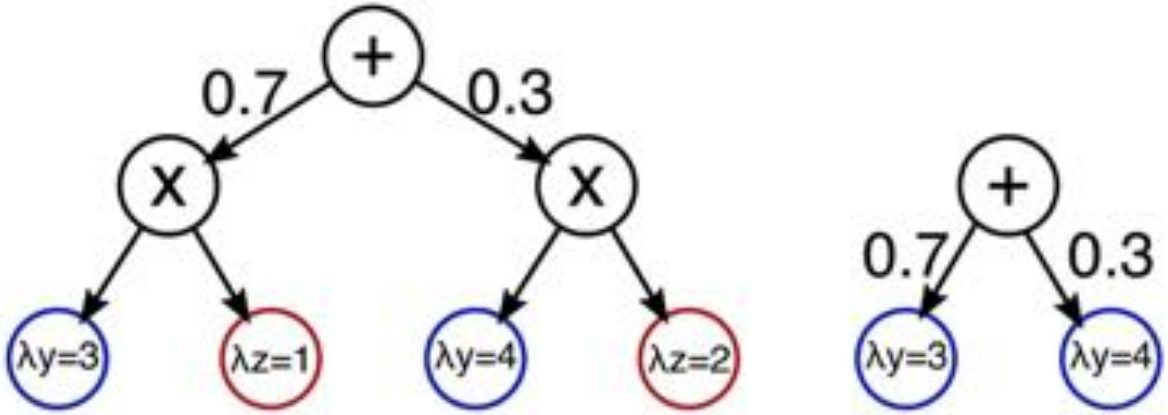
TECHNISCHE UNIVERSITÄT DARMSTADT

Testing independence of Poisson random variables



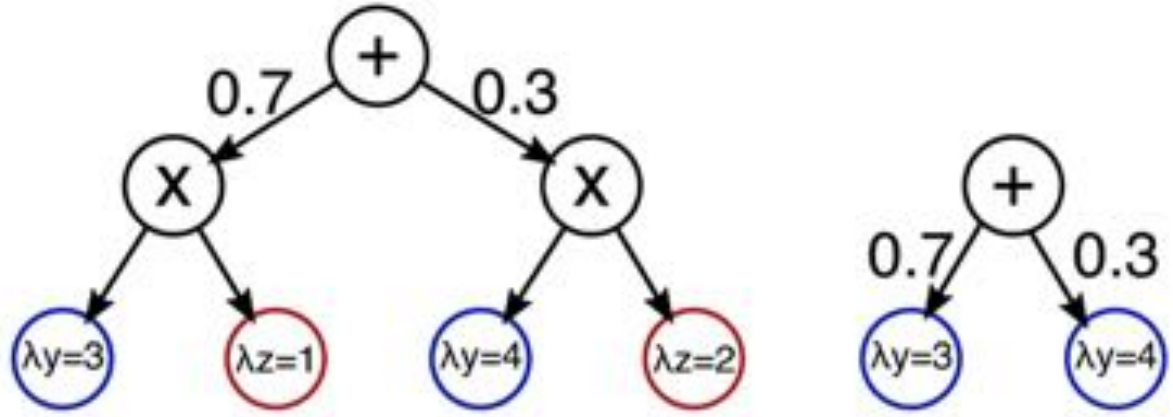
Tractable Inference via Symbolic Evaluation

TECHNISCHE UNIVERSITÄT DARMSTADT



SPNs encode polynomials over univariate distributions to build multivariate distributions. They can be fed into symbolic evaluation methods for inference

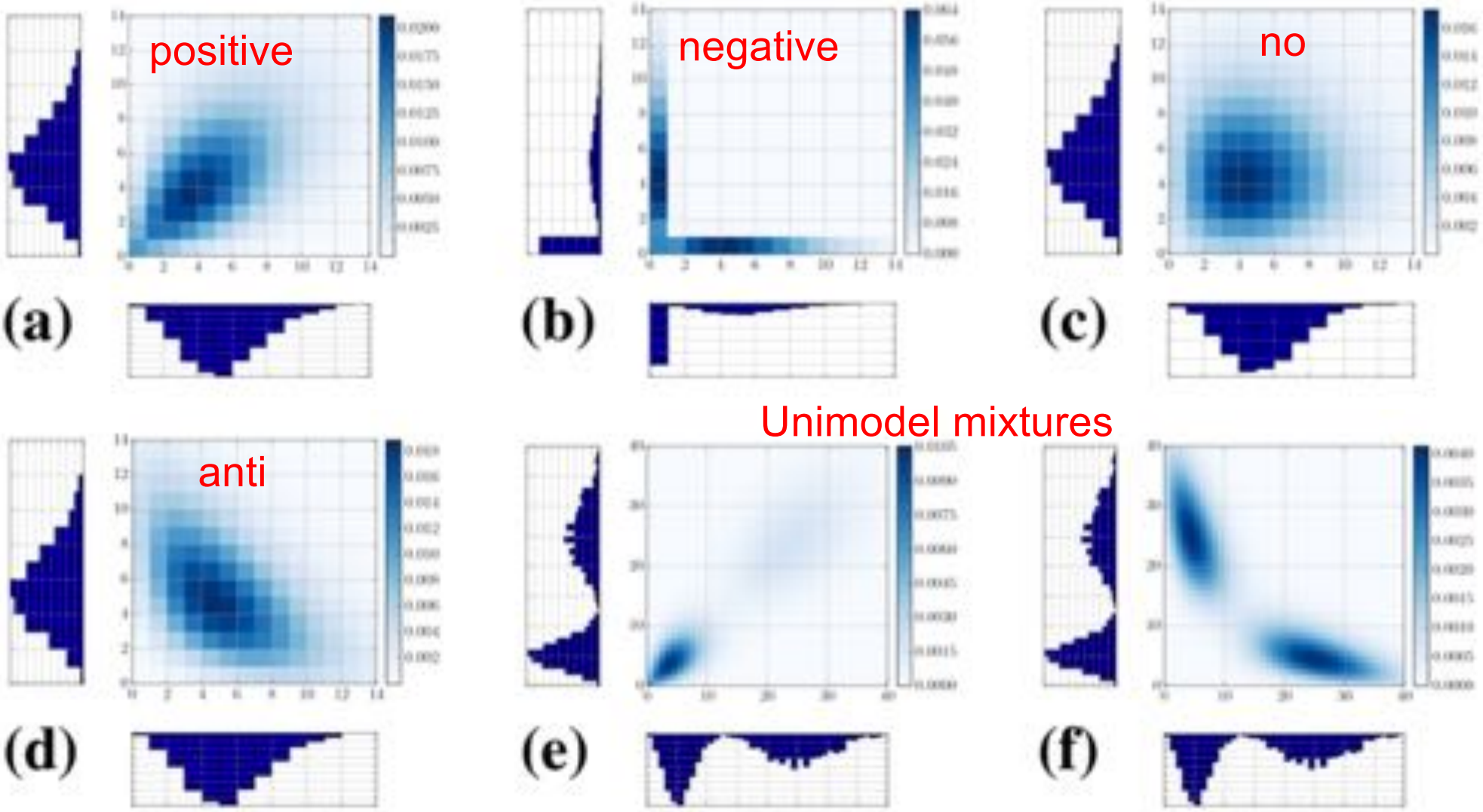
Inference on Devices



Ultimately, they may even be compiled into simple, flat, library-free code suitable for embedding in real-time applications and devices

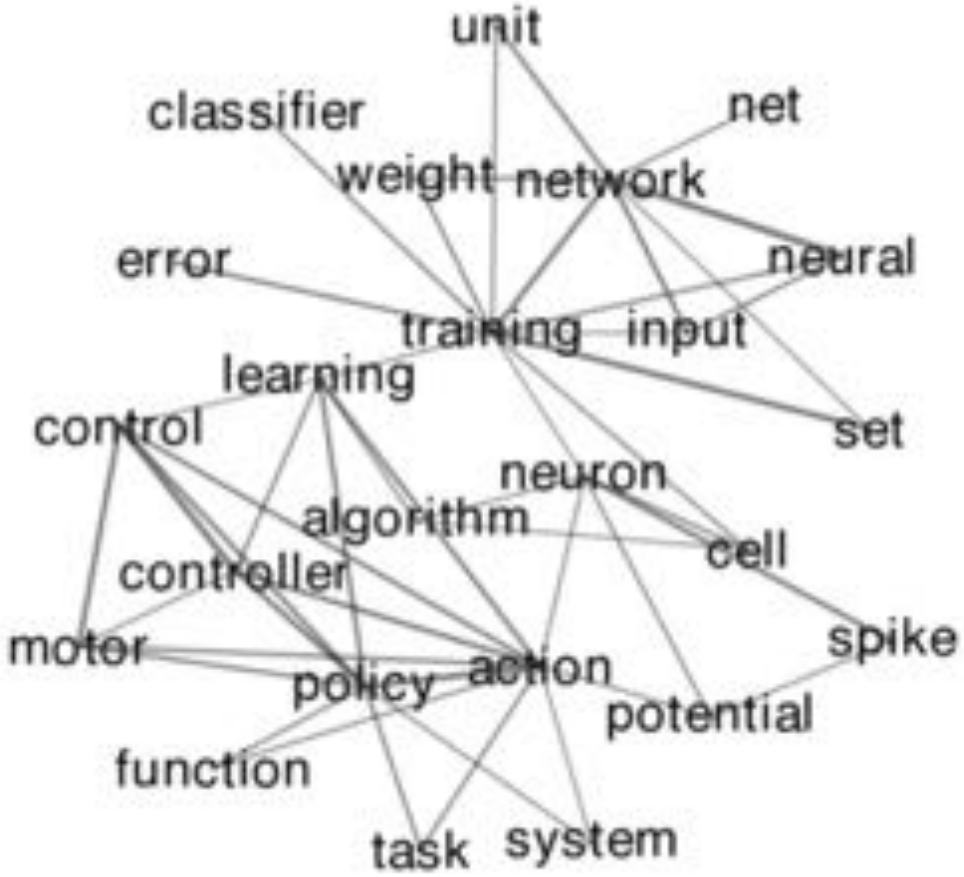
SPNs encode polynomials over univariate distributions to build multivariate distributions. They can be fed into symbolic evaluation methods for inference

Flexible Dependencies even for challenging distributions, here multivariate Poisson distribution

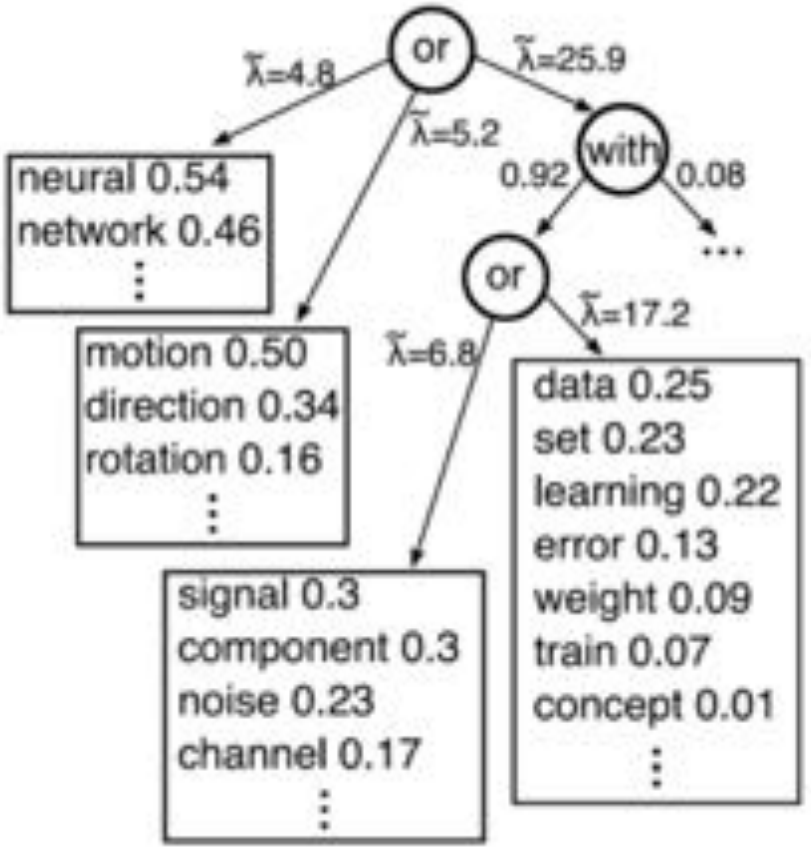


Multiple views on you data due to efficient inference

TECHNISCHE UNIVERSITÄT DARMSTADT



Mutual Information
(NIPS corpus)



Poisson Multinomial SPN
= hierachical topic model

Poisson SPNs provide topics:
From Topic Models to Poisson Mean Fields

[Bishop, Fienberg, Holland. Discrete Multivariate Analysis: Theory and Practice. Springer, 2007]

Topic-word distributions of LDA can be viewed as a product of independent Poissons

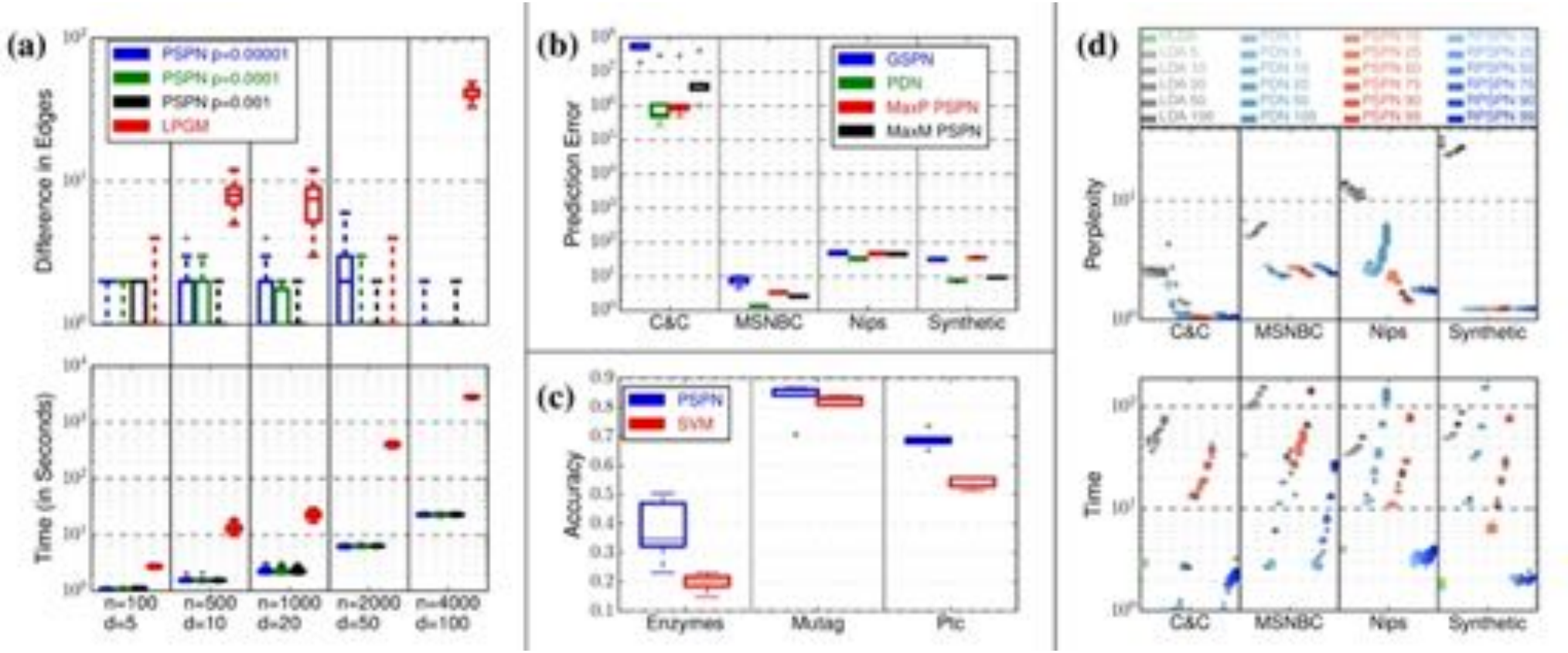
$$\begin{aligned}
&\Pr_{\text{Poiss}}\left(\tilde{x}\,|\,\tilde{\lambda}\right)\Pr_{\text{Mult}}\left(\mathbf{x}\,|\,\theta=(\lambda_1,\cdots,\lambda_p)\,/\tilde{\lambda},\mathsf{N}=\tilde{x}\right)\\
&\quad=\frac{e^{-\tilde{\lambda}}}{\tilde{x}!}\tilde{\lambda}^{\tilde{x}}\frac{\tilde{x}!}{\prod_{s=1}^{p}x_s!}\prod_{s=1}^{p}\left(\frac{\lambda_s}{\tilde{\lambda}}\right)^{x_s}\\
&\quad=\frac{\tilde{x}!}{\tilde{x}!}\frac{e^{-\tilde{\lambda}}}{\prod_{s=1}^{p}x_s!}\prod_{s=1}^{p}\left(\frac{\tilde{\lambda}\lambda_s}{\tilde{\lambda}}\right)^{x_s}\\
&\quad=\Pr_{\text{Ind. Poiss}}(\mathbf{x}\,|\,\lambda_1,\cdots,\lambda_p)=\prod_{s=1}^{p}\frac{e^{-\lambda_s}}{x_s!}\lambda_s^{x_s}
\end{aligned}$$

Competitive Predictive Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

Prediction error

LDA



Network discovery

Deep Graph Kernels (Weisfeiler Lehman)

[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

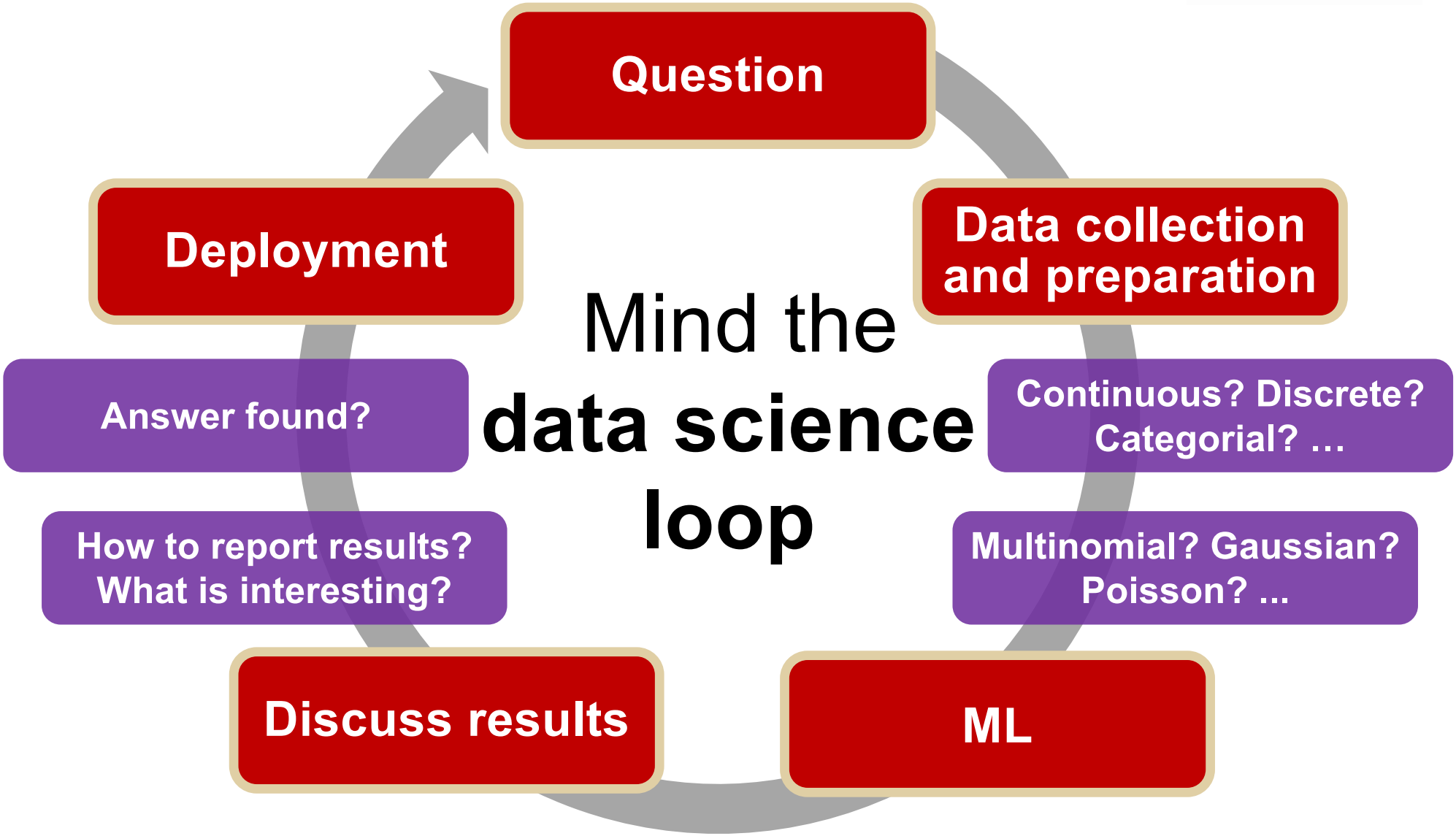
[Molina, Vergari, Stelzner, Peharz, Subramani, Poupart, Di Mauro, Kersting 2019]

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

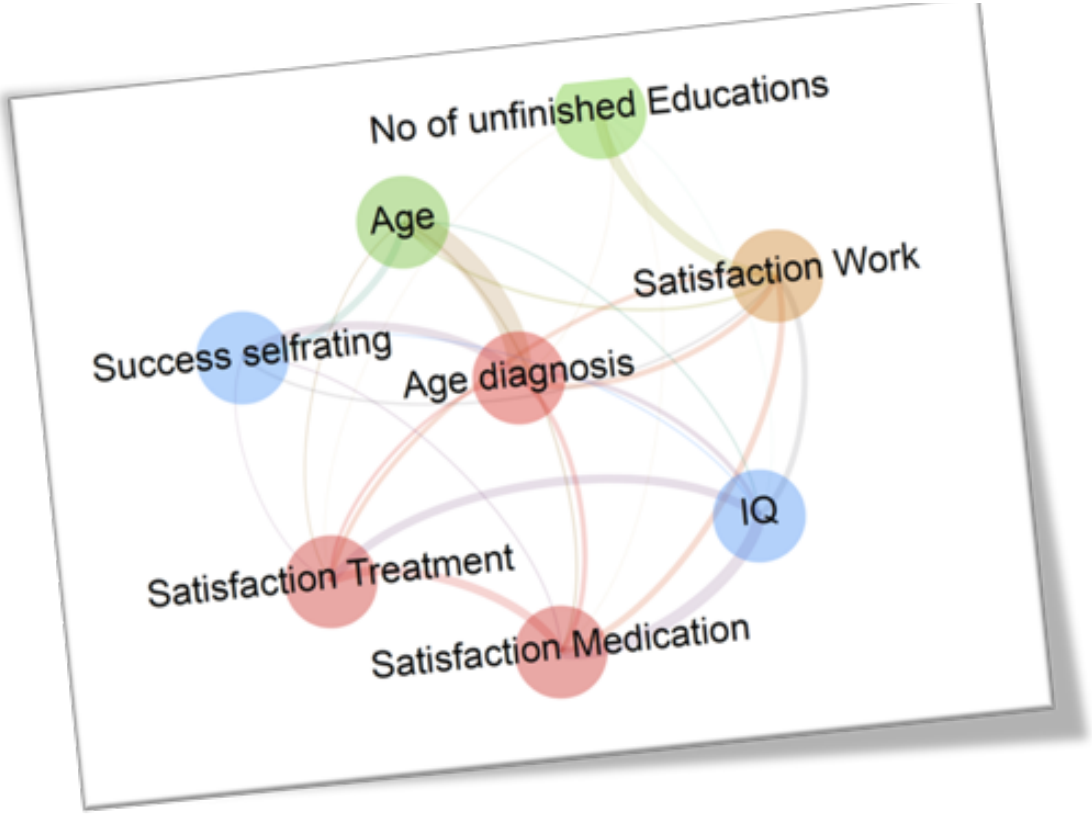
SPFlow, an open-source Python library providing a simple interface to inference, learning and manipulation routines for deep and tractable probabilistic models called Sum-Product Networks (SPNs). The library allows one to quickly create SPNs both from data and through a domain specific language (DSL).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


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

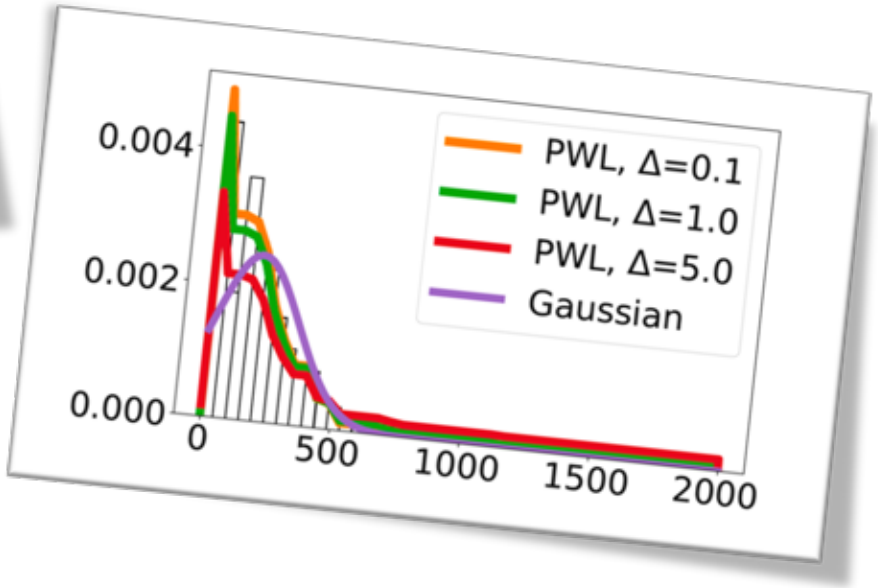
[Molina, Natarajan, Vergari, Di Mauro, Esposito, Kersting AAAI 2018]

TECHNISCHE UNIVERSITÄT DARMSTADT
THE UNIVERSITY OF TEXAS AT DALLAS
UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO

Distribution-agnostic PC



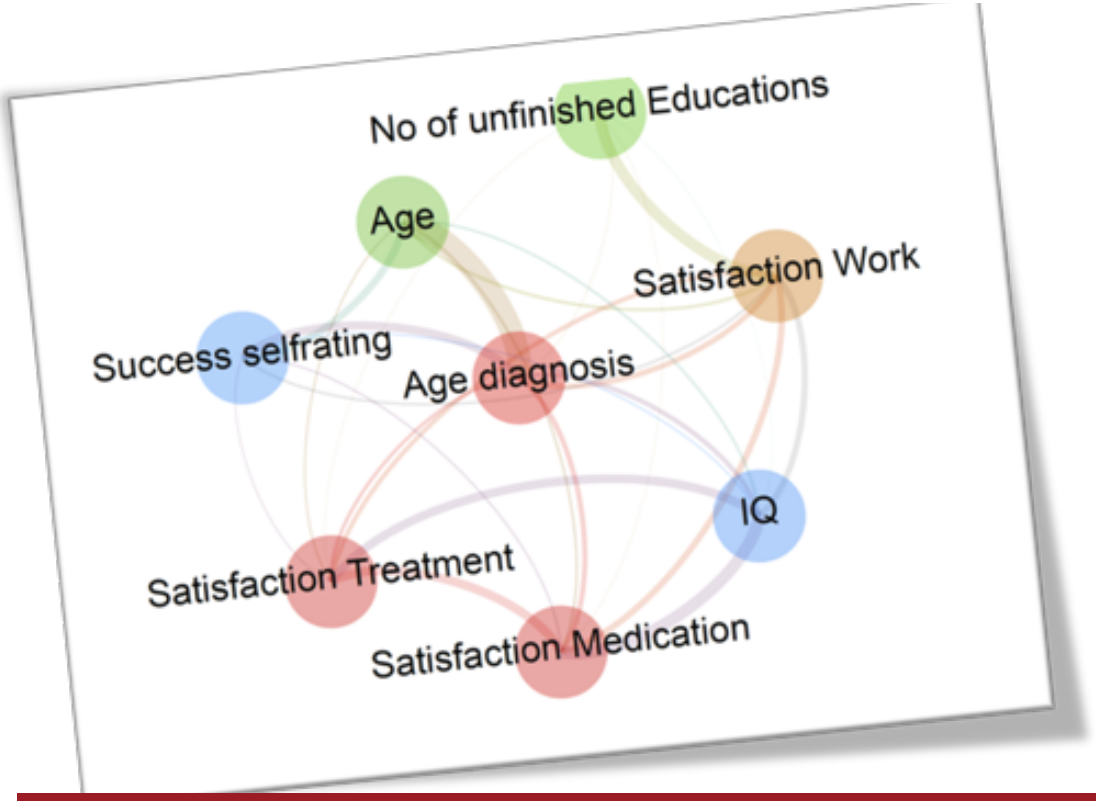
Use nonparametric independency tests and piece-wise linear approximations



[Molina, Natarajan, Vergari, Di Mauro, Esposito, Kersting AAAI 2018]

TECHNISCHE UNIVERSITÄT DARMSTADT
THE UNIVERSITY OF TEXAS AT DALLAS
UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO

Distribution-agnostic PC



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

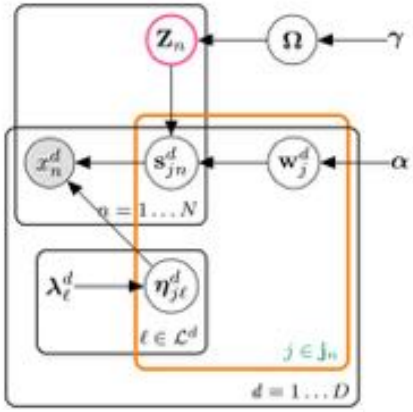
UBER AI Labs

UNIVERSITY OF CAMBRIDGE

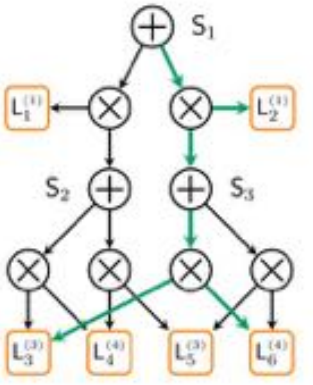
Max Planck Institute for Intelligent Systems

TECHNISCHE UNIVERSITÄT DARMSTADT

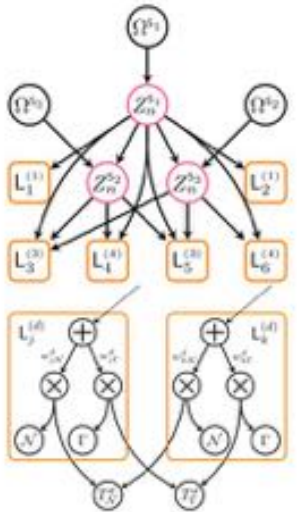
We can even automatically discovers the statistical types and parametric forms of the variables



Bayesian Type Discovery



Mixed Sum-Product Network



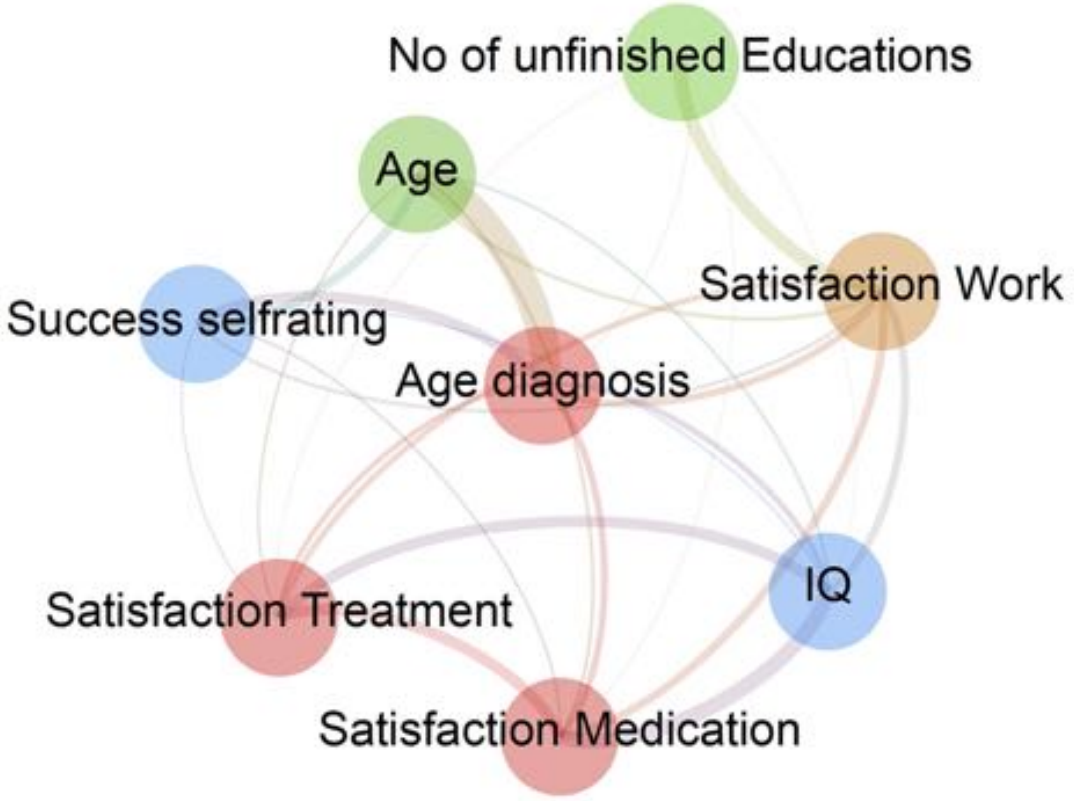
Automatic Statistician

[Molina, Natarajan, Vergari, Di Mauro, Esposito, Kersting Submitted 2017]

TECHNISCHE UNIVERSITÄT DARMSTADT

Towards Automated Statisticians:

make use of nonparametric (in)dependency tests and distributions to build nonparametric SPNs that are agnostic to the statistical type of the random variables



Visualization the gist of a Mixed SPN (learned on the hybrid Autism dataset) using normalized mutual information (the thicker, the higher). **The machine learning tool does not know anything about the data types.** Node colors encode different feature groups: Demographics (green), Psychological (blue), Social Environment (organge) and Medical (red).

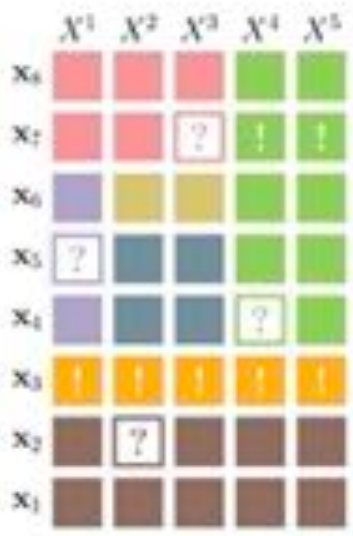
[Vergari, Molina, Peharz, Ghahramani, Kersting, Valera AAAI 2019]

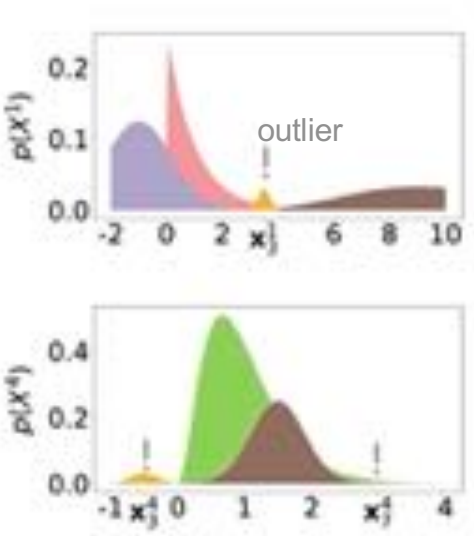
The Automatic Data Scientist

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

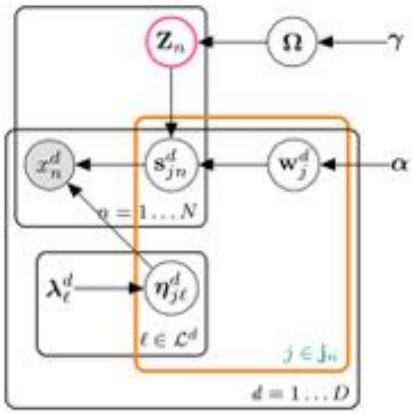
AAAI-19: Thirty-Third AAAI Conference on Artificial Intelligence
January 27 – February 1, 2019, Hilton Hawaiian Village, Honolulu, Hawaii, USA

CAML · DFG MADESI · Federal Ministry of Education and Research

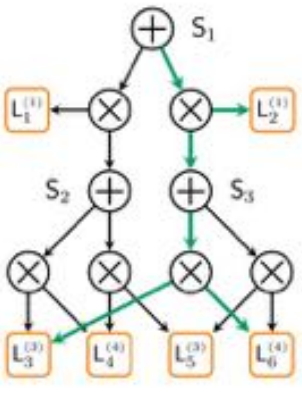




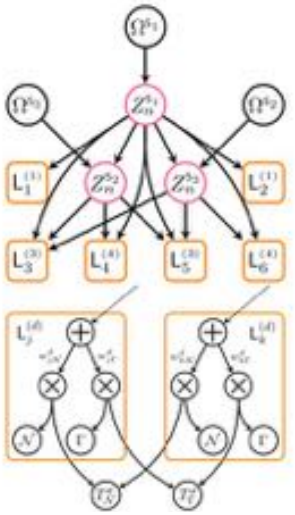
We can even automatically discovers the statistical types and parametric forms of the variables



Bayesian Type Discovery

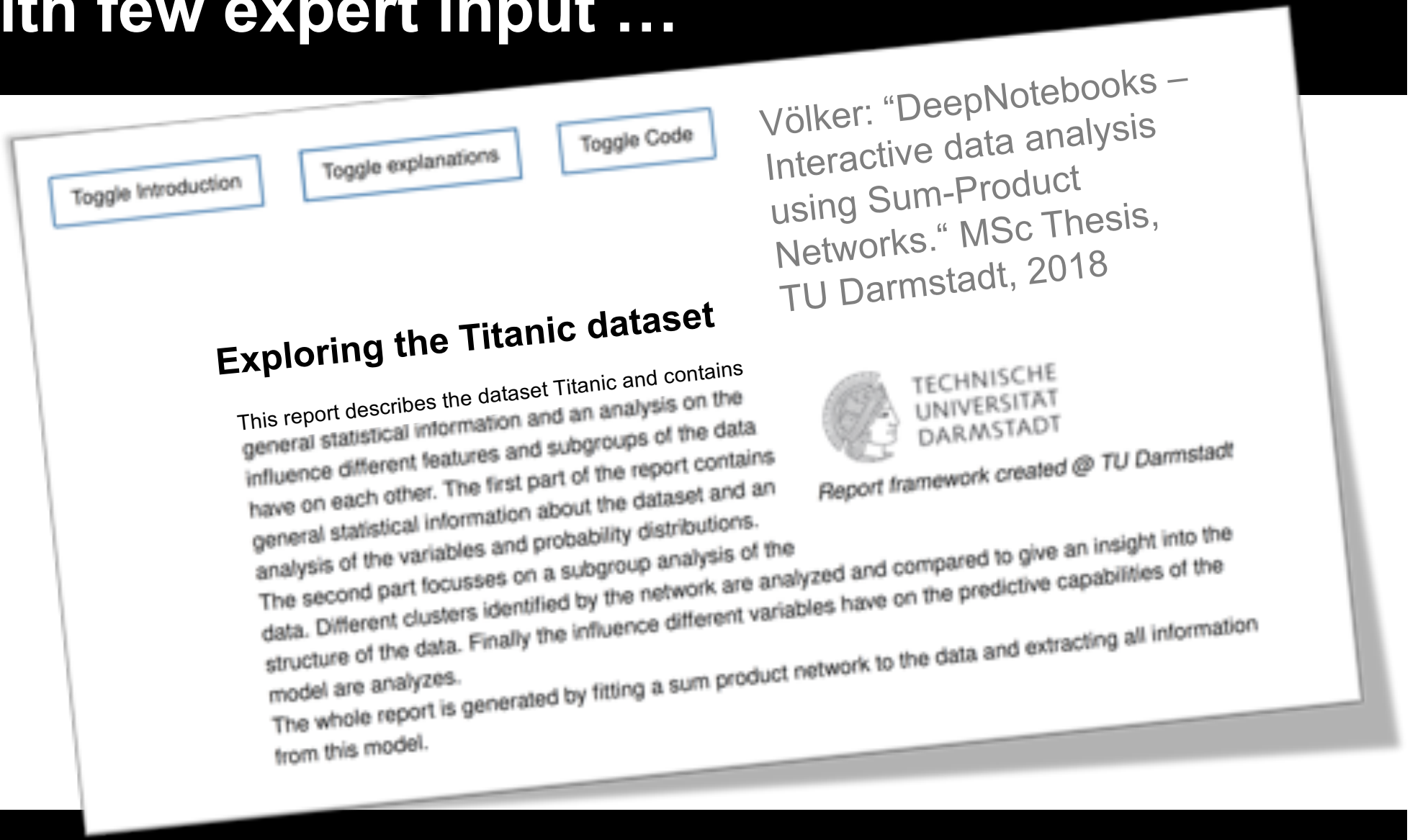


Mixed Sum-Product Network



Automatic Statistician

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



…and can compile data reports automatically

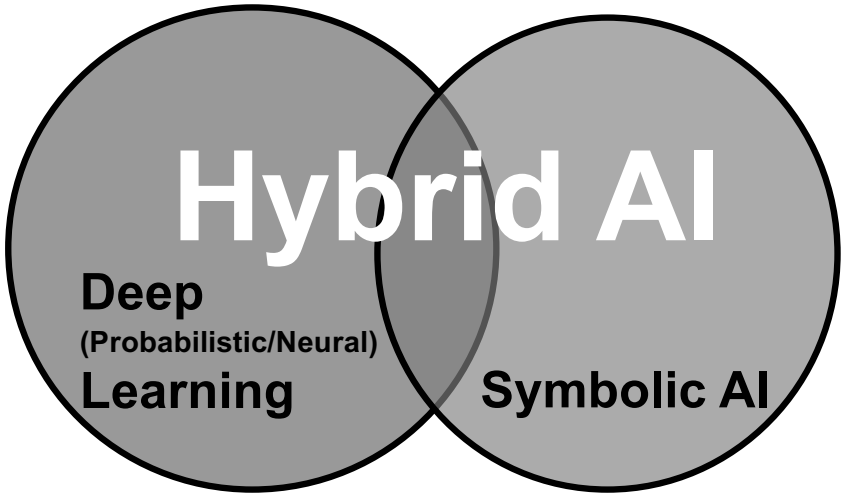
Overall we need languages for Systems AI,

the computational and mathematical modeling of complex AI systems.

[Kordjamshidi, Roth, Kersting: "Systems AI: A Declarative Learning Based Programming Perspective." IJCAI-ECAI 2018] following the motivation of Christopher Re

The next breakthrough in AI may not just be a new ML/AI algorithm…

…but may be in the ability to rapidly combine, deploy, and maintain existing AI algorithms



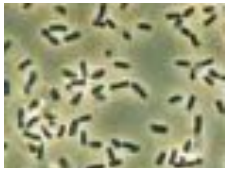
Hybrid AI: Unsupervised scene understanding

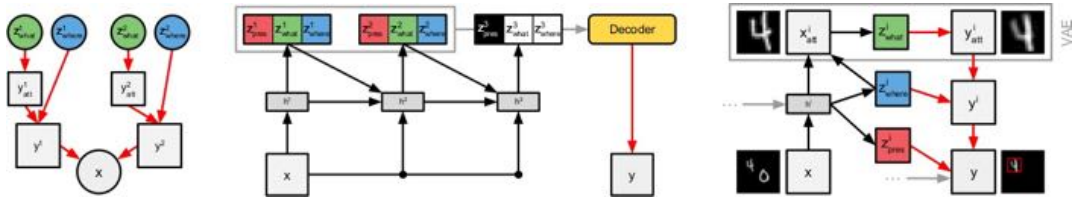
[Stelzner, Peharz, Kersting ICML 2019, Best Paper Award at TPM@ICML2019] https://github.com/stelzner/supair

UNIVERSITY OF CAMBRIDGE TECHNISCHE UNIVERSITÄT DARMSTADT

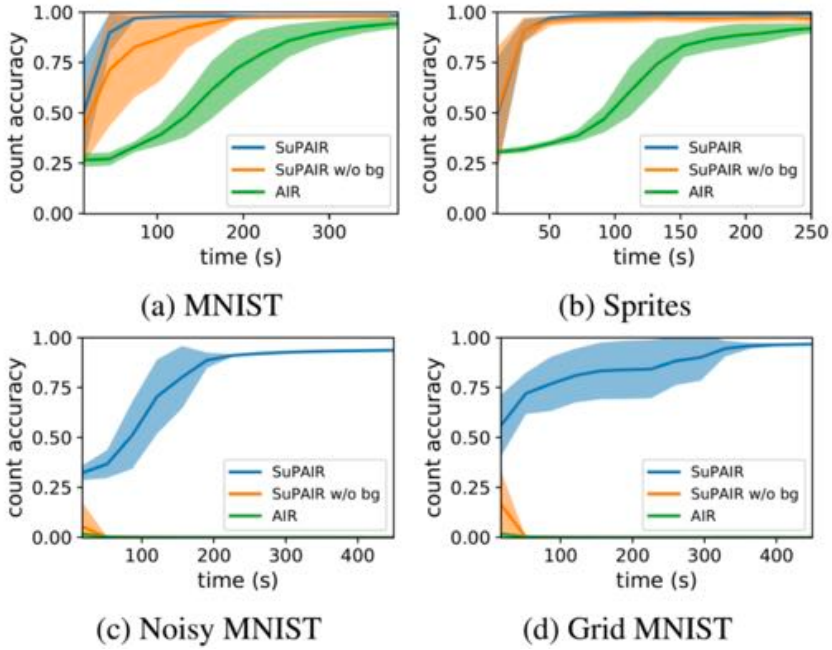
ICML | 2019
Thirty-sixth International Conference on Machine Learning

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

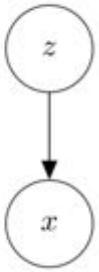




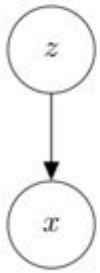
[Attend-Infer-Repeat (AIR) model, Hinton et al. NIPS 2016]



VAE



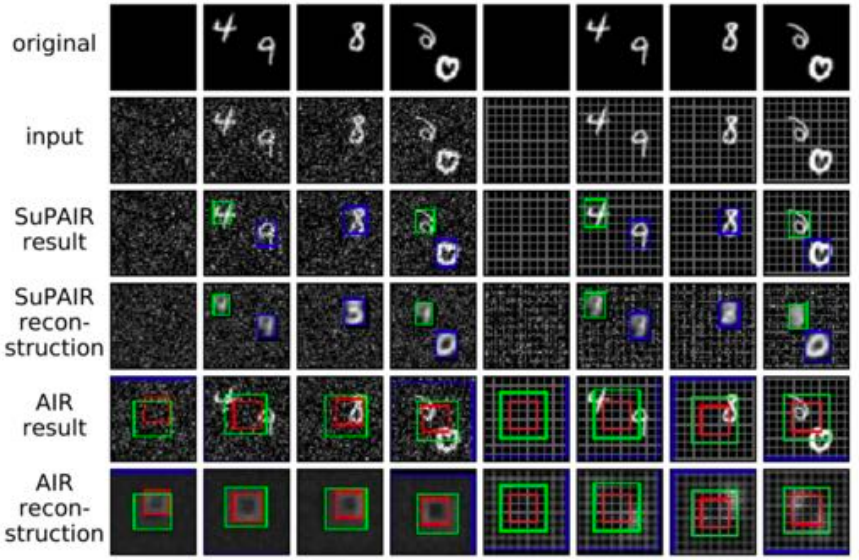
SPN



Replace VAE by SPN as object model

- 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]

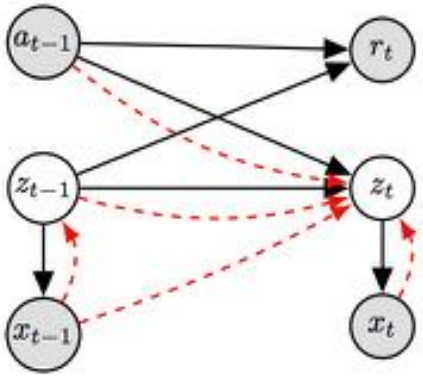


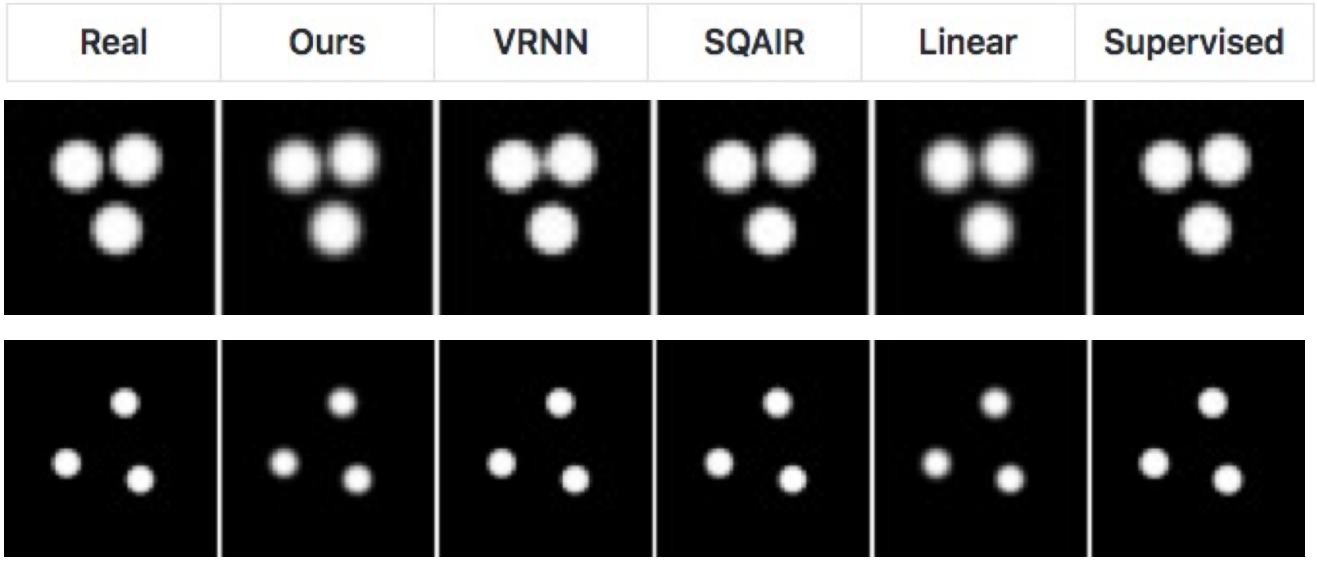
Hybrid AI: Unsupervised physics learning

[Kossen, Stelzner, Hussing, Voelcker, Kersting ICLR 2020]

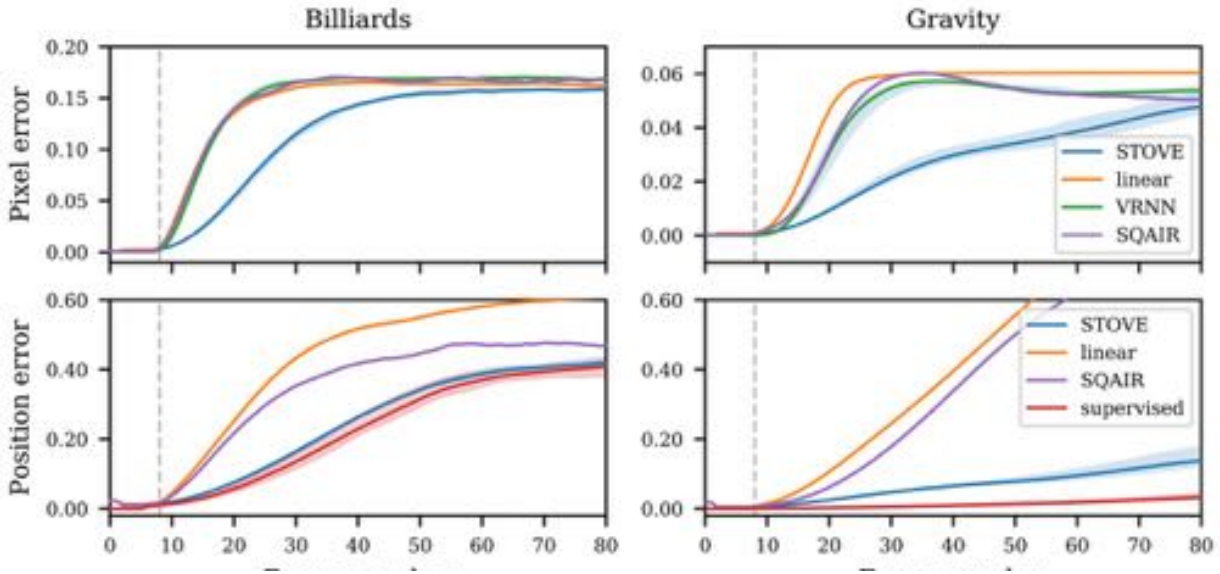
ICLR | 2020
Eighth International Conference on Learning Representations

TECHNISCHE UNIVERSITÄT DARMSTADT



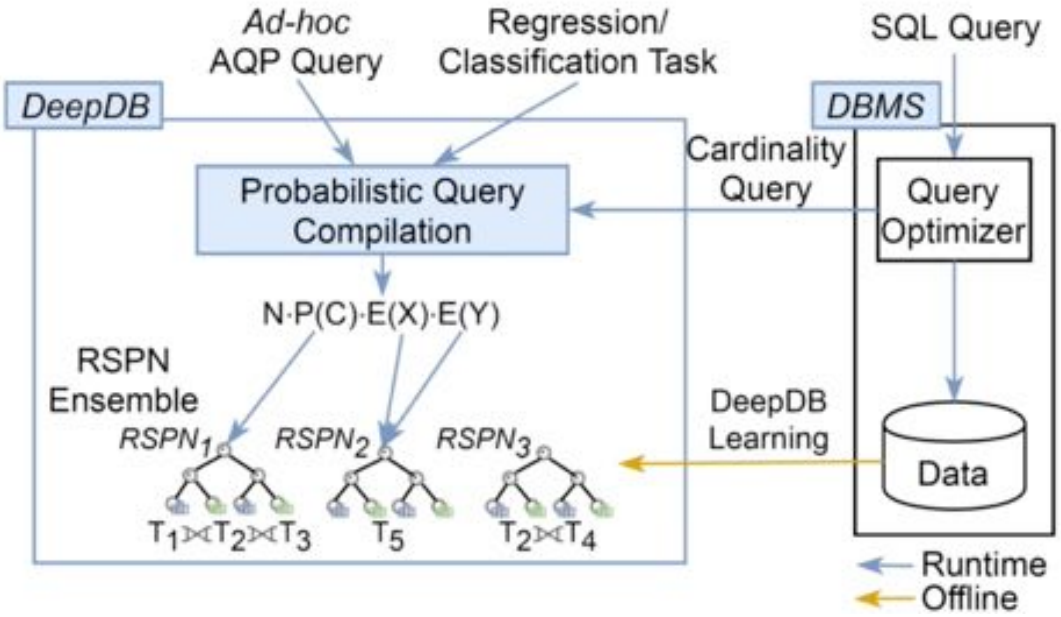


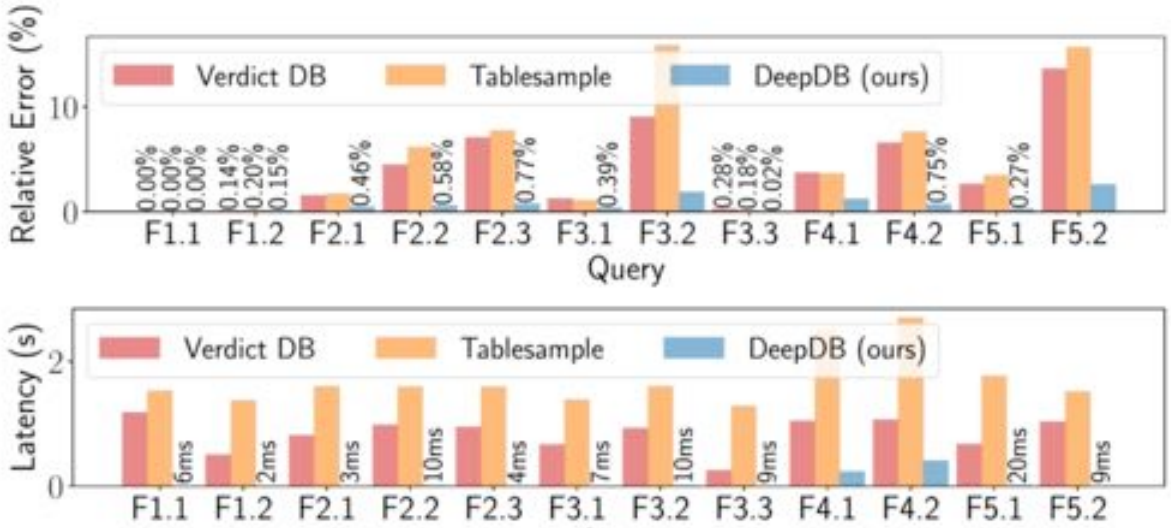
putting structure and tractable inference into deep models



Hybrid AI: Deep Databases

[Hilprecht, Schmidt, Kulessa, Molina, Kersting, Binnig VLDB 2020]





What have we learnt about SPNs?

Sum-product networks (SPNs)

- DAG of sums and products
- They are instances of Arithmetic Circuits (ACs)
- Compactly represent partition function
- Learn many layers of hidden variables

Efficient marginal inference

Easy learning

Can outperform well-known alternatives