UCLA Computer Science

STAR AI RESEARCH LAB UCLA

# Tractable Probabilistic Circuits

Guy Van den Broeck

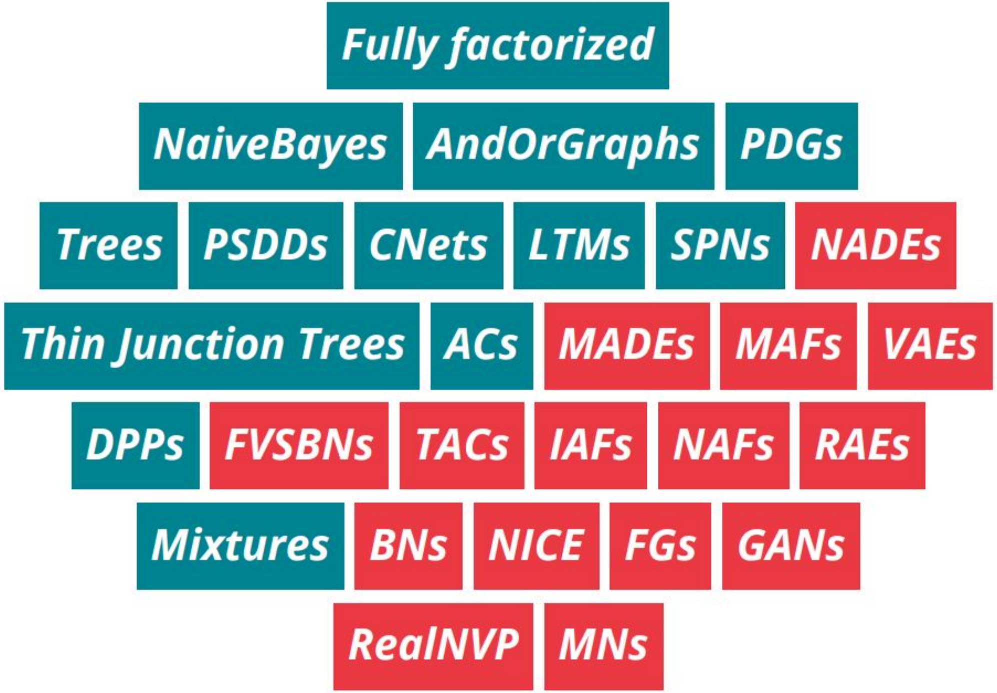
VinAI Research - May 27, 2022

# Outline

1. What are tractable probabilistic circuits?
2. Are these models any good?
3. How far can we push tractable inference?
4. What is their expressive power?

# Outline

1. **What are tractable probabilistic circuits?**
2. Are these models any good?
3. How far can we push tractable inference?
4. What is their expressive power?



*Fully factorized*

*NaiveBayes* *AndOrGraphs* *PDGs*

*Trees* *PSDDs* *CNets* *LTMs* *SPNs* *NADEs*

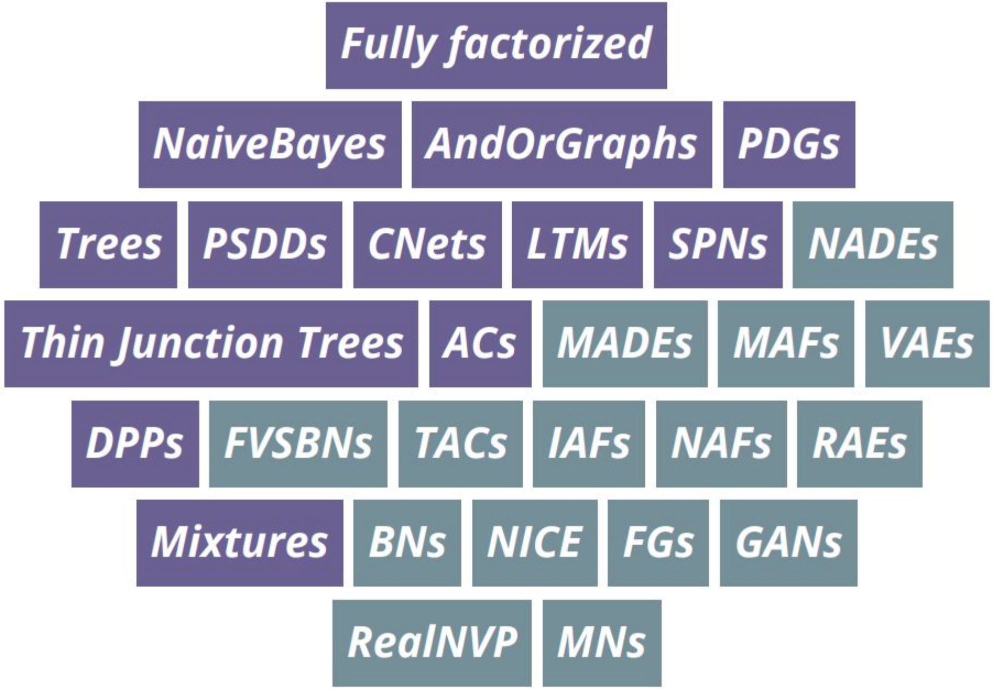
*Thin Junction Trees* *ACs* *MADEs* *MAFs* *VAEs*

*DPPs* *FVSBNs* *TACs* *IAFs* *NAFs* *RAEs*

*Mixtures* *BNs* *NICE* *FGs* *GANs*

*RealNVP* *MNs*

# *Intractable* and *tractable* models



Fully factorized

NaiveBayes AndOrGraphs PDGs

Trees PSDDs CNets LTMs SPNs NADEs

Thin Junction Trees ACs MADEs MAFs VAEs

DPPs FVSBNs TACs IAFs NAFs RAEs

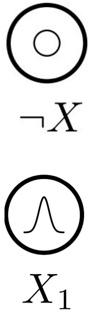
Mixtures BNs NICE FGs GANs

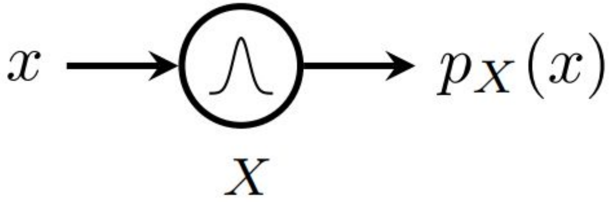
RealNVP MNs

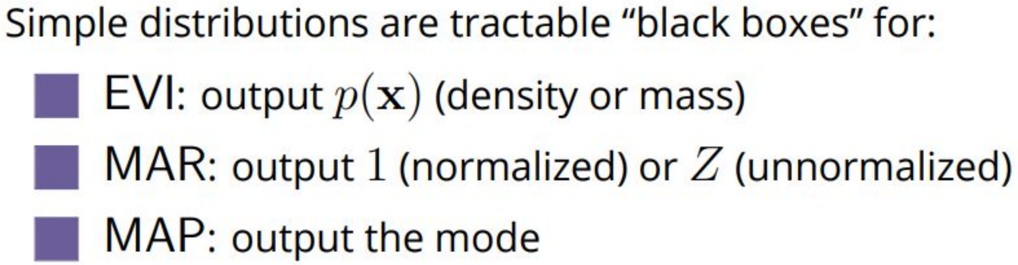
**a *unifying framework* for tractable models**

# Probabilistic circuits

*computational graphs* that recursively define distributions





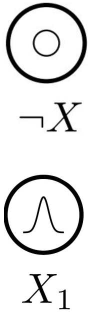


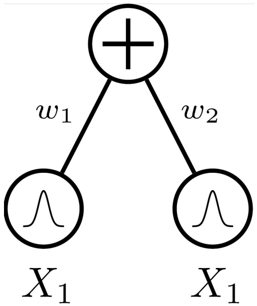
Simple distributions are tractable “black boxes” for:

- EVI: output $p(\mathbf{x})$ (density or mass)
- MAR: output $1$ (normalized) or $Z$ (unnormalized)
- MAP: output the mode

# Probabilistic circuits

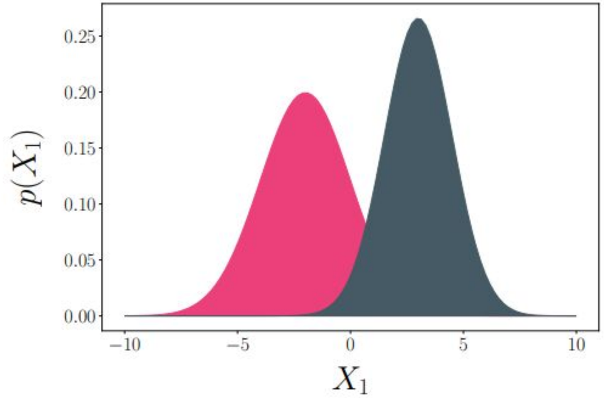
*computational graphs* that recursively define distributions





$$p(X_1) = w_1 p_1(X_1) + w_2 p_2(X_1)$$

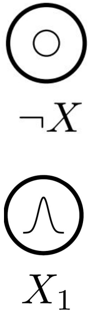
⇒
*mixtures*

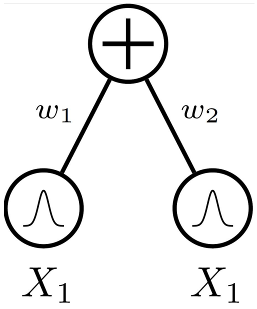


$$p(X) = p(Z = \boxed{1}) \cdot p_1(X|Z = \boxed{1}) + p(Z = \boxed{2}) \cdot p_2(X|Z = \boxed{2})$$

# Probabilistic circuits

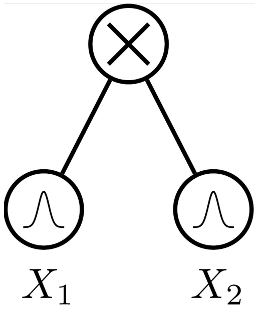
*computational graphs* that recursively define distributions





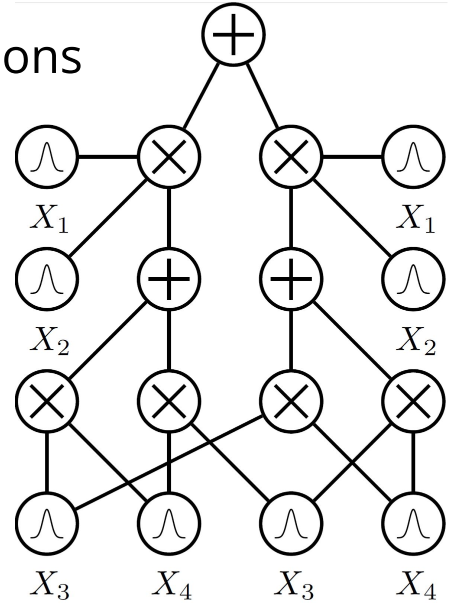
$p(X_1) = w_1 p_1(X_1) + w_2 p_2(X_1)$

⇒ *mixtures*



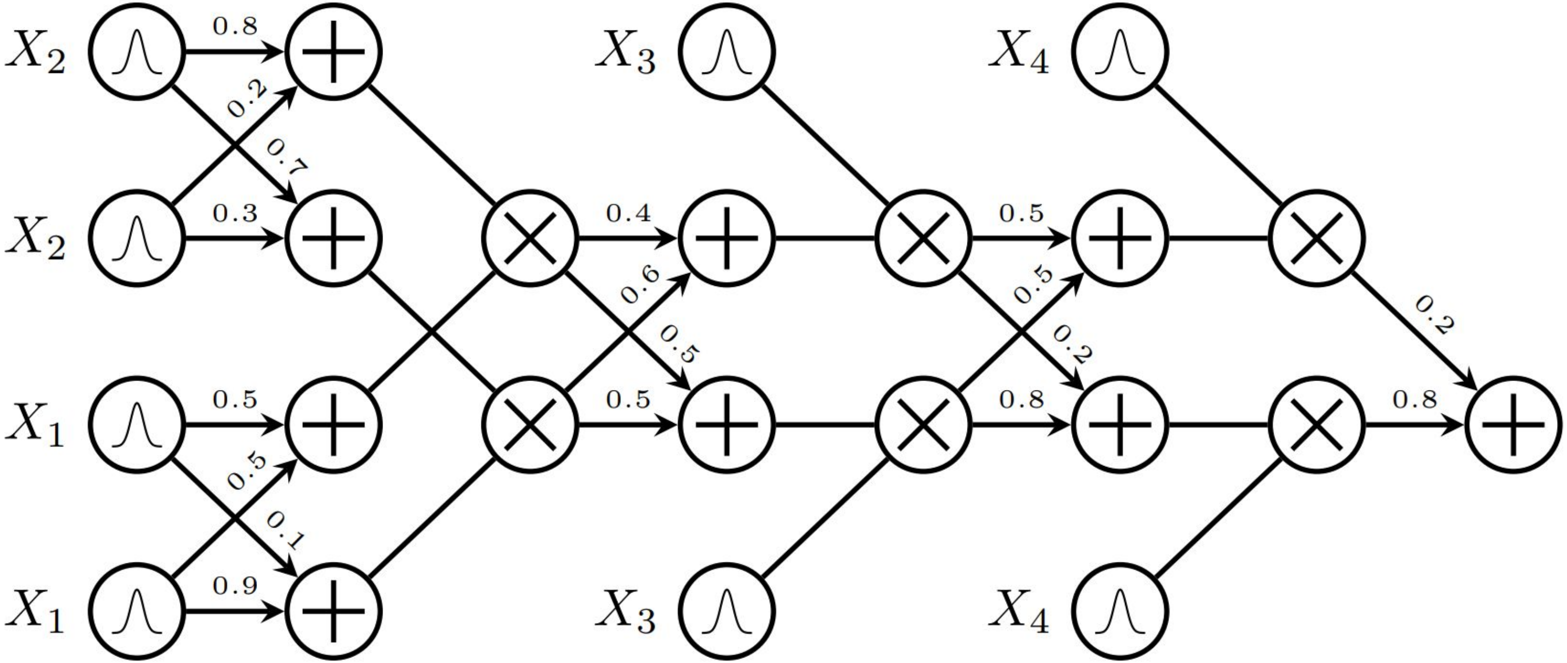
$p(X_1, X_2) = p(X_1) \cdot p(X_2)$

⇒ *factorizations*



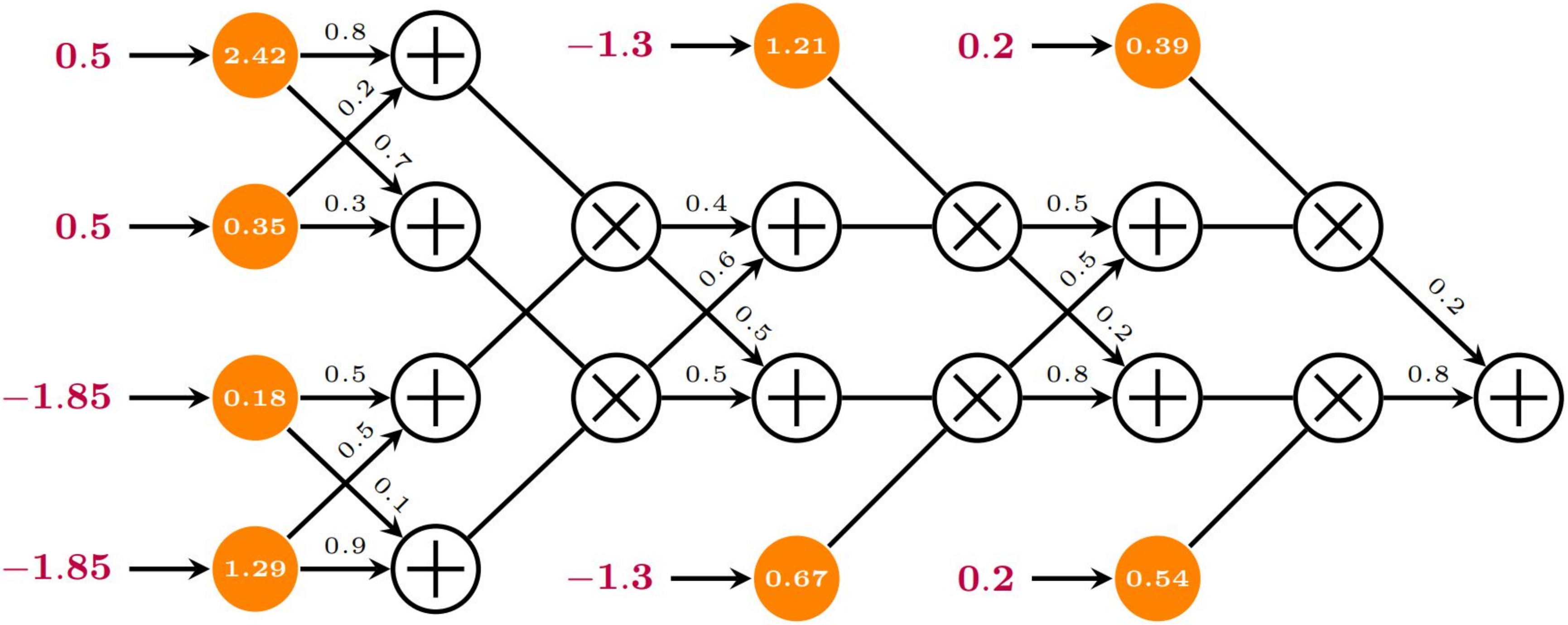
# Likelihood

$$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$$



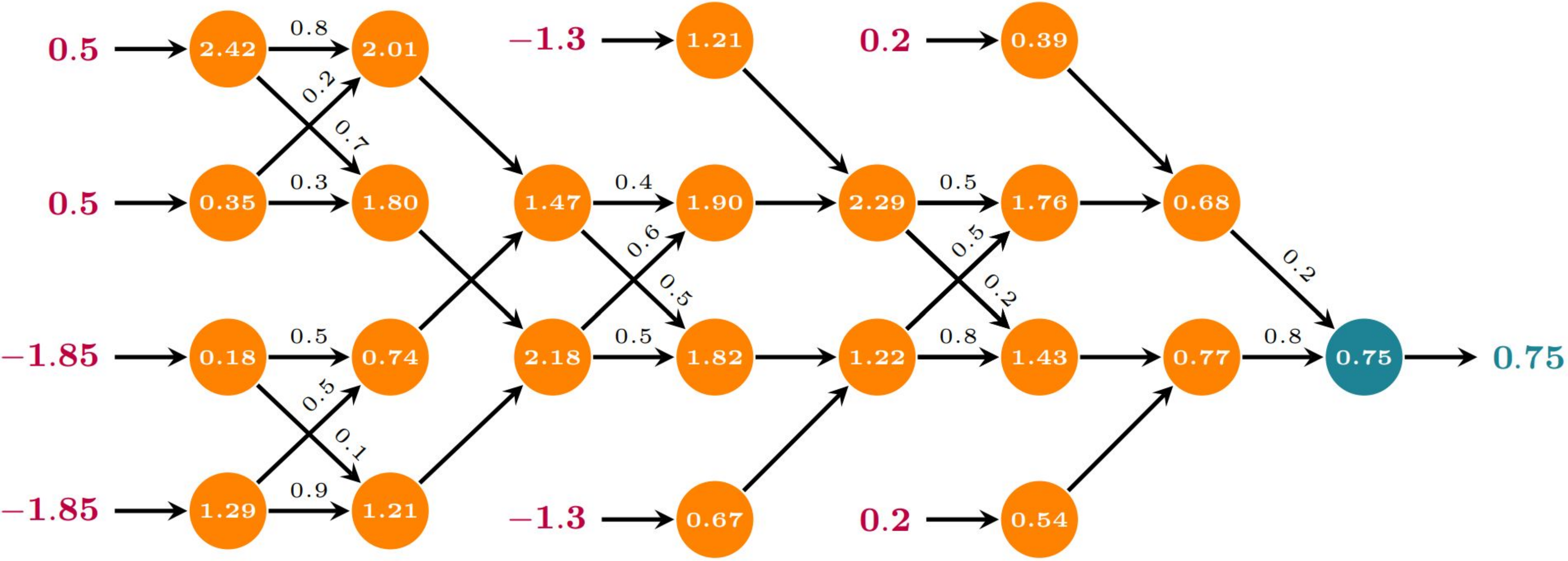
# Likelihood

$$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$$

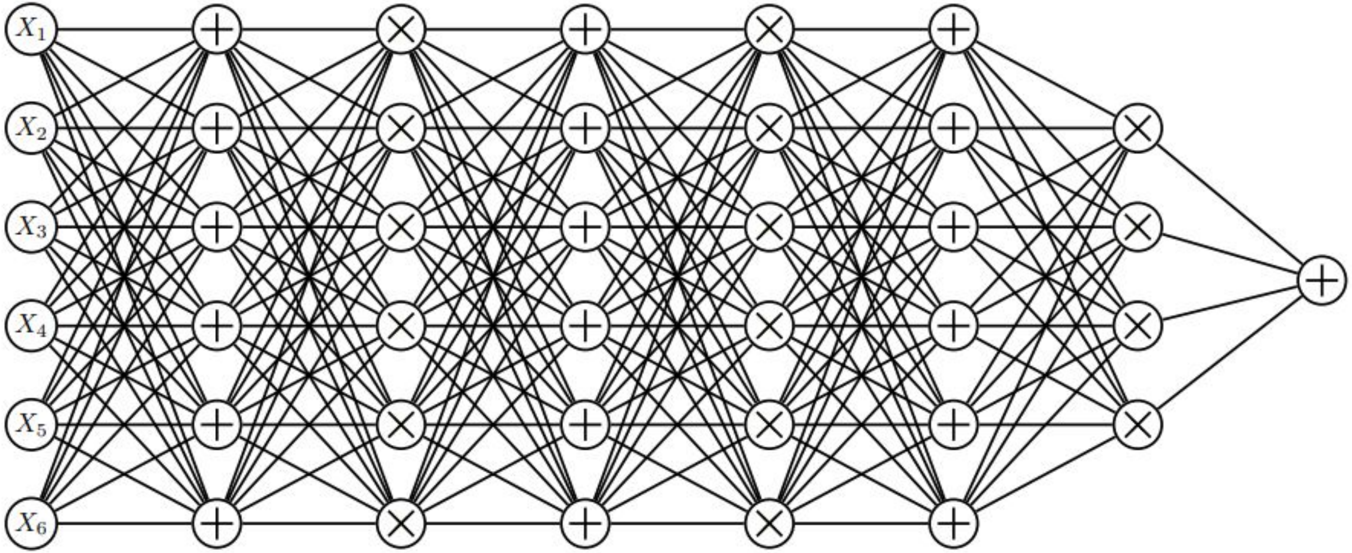


# Likelihood

$$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$$

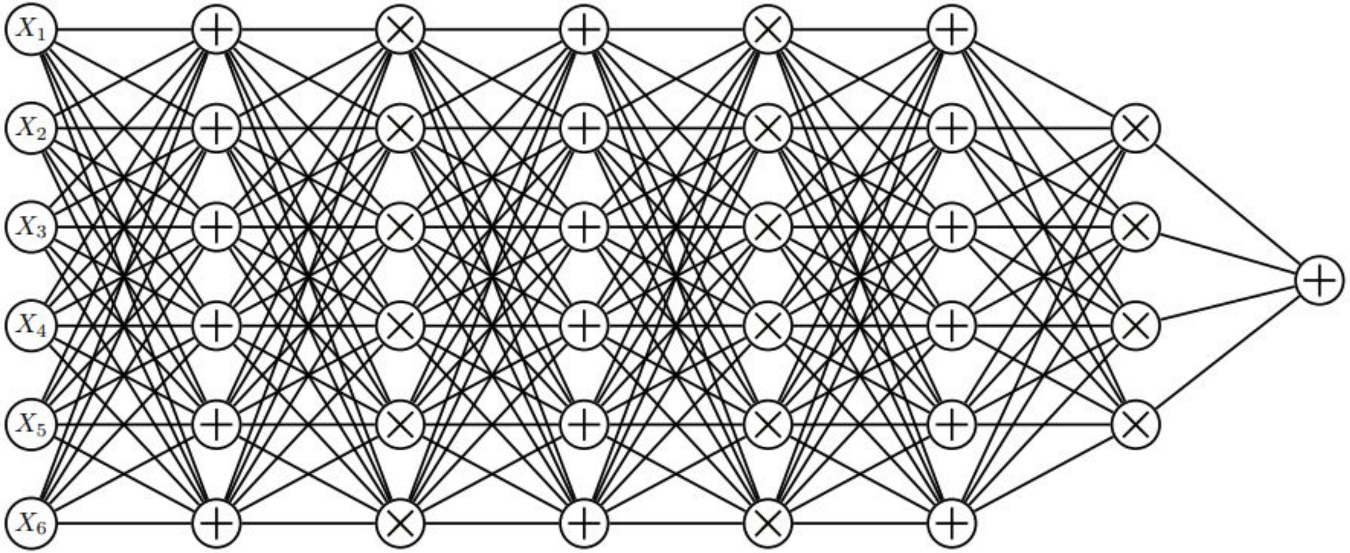


# Just sum, products and distributions?



***just arbitrarily compose them like a neural network!***

# Just sum, products and distributions?



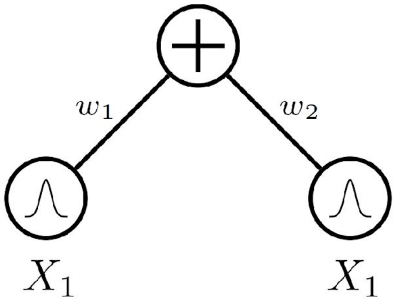
***~~just arbitrarily compose them like a neural network!~~***

⇒ structural constraints needed for tractability

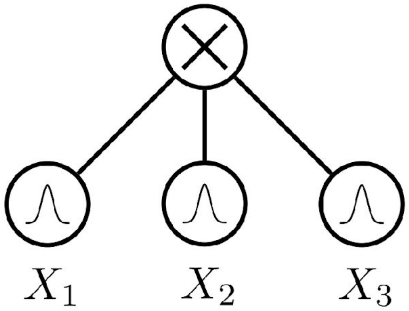
# Tractable marginals

A sum node is ***smooth*** if its children depend on the same set of variables.

A product node is ***decomposable*** if its children depend on disjoint sets of variables.



***smooth circuit***



***decomposable circuit***

Darwiche and Marquis, “A Knowledge Compilation Map”, 2002

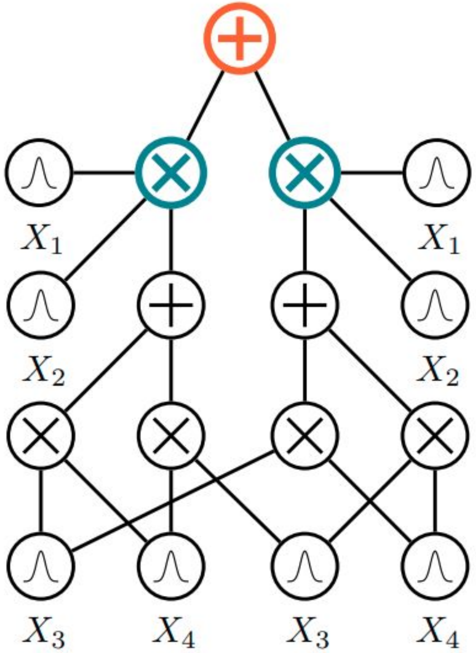
# Smoothness + decomposability = tractable MAR

If $p(\mathbf{x}) = \sum_i w_i p_i(\mathbf{x})$, (***smoothness***):

$$\int p(\mathbf{x})d\mathbf{x} = \int \sum_i w_i p_i(\mathbf{x})d\mathbf{x} =$$

$$= \sum_i w_i \int p_i(\mathbf{x})d\mathbf{x}$$

$\Rightarrow$ *integrals are "pushed down" to children*



[Darwiche & Marquis JAIR 2001, Poon & Domingos UAI11]

# Smoothness + decomposability = tractable MAR

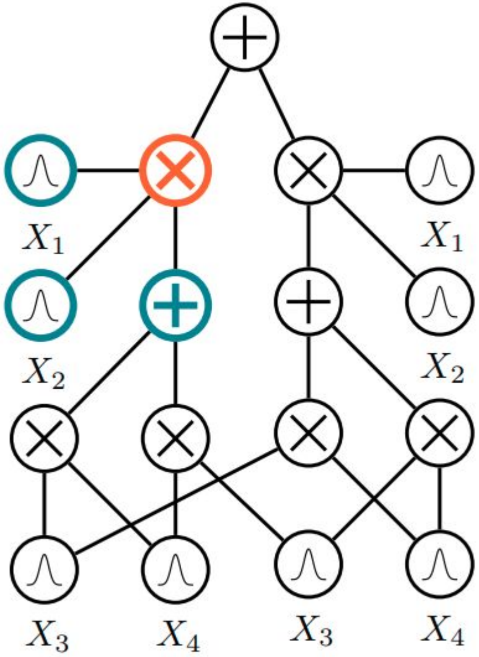
If $p(\mathbf{x},\mathbf{y},\mathbf{z}) = p(\mathbf{x})p(\mathbf{y})p(\mathbf{z})$, (***decomposability***):

$$\int\int\int p(\mathbf{x},\mathbf{y},\mathbf{z})d\mathbf{x}d\mathbf{y}d\mathbf{z} =$$

$$= \int\int\int p(\mathbf{x})p(\mathbf{y})p(\mathbf{z})d\mathbf{x}d\mathbf{y}d\mathbf{z} =$$

$$= \int p(\mathbf{x})d\mathbf{x}\int p(\mathbf{y})d\mathbf{y}\int p(\mathbf{z})d\mathbf{z}$$

$\Rightarrow$ *integrals decompose into easier ones*



# Smoothness + decomposability = tractable MAR

Forward pass evaluation for MAR

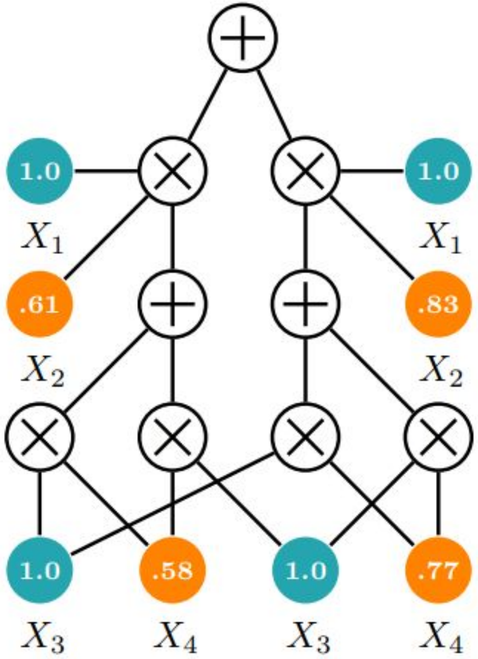
$\Rightarrow$ *linear in circuit size!*

E.g. to compute $p(x_2, x_4)$:

- leafs over $X_1$ and $X_3$ output $\boldsymbol{Z}_i = \int p(x_i)dx_i$

  $\Rightarrow$ *for normalized leaf distributions:* 1.0

- leafs over $X_2$ and $X_4$ output **EVI**
- feedforward evaluation (bottom-up)



# Smoothness + decomposability = tractable MAR

Forward pass evaluation for MAR

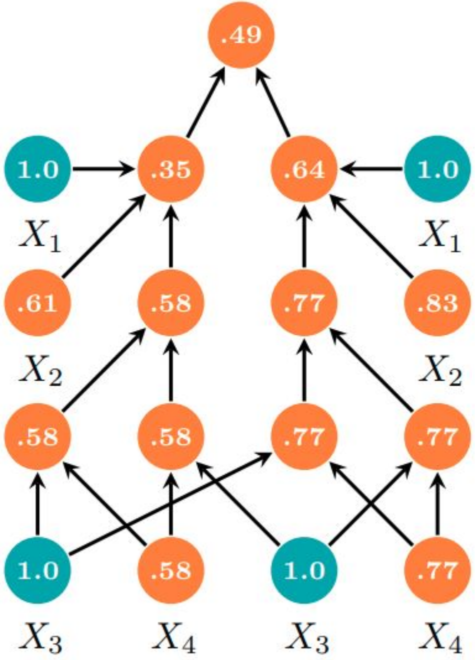
$\Rightarrow$ *linear in circuit size!*

E.g. to compute $p(x_2, x_4)$:

- leafs over $X_1$ and $X_3$ output $\boldsymbol{Z}_i = \int p(x_i) dx_i$

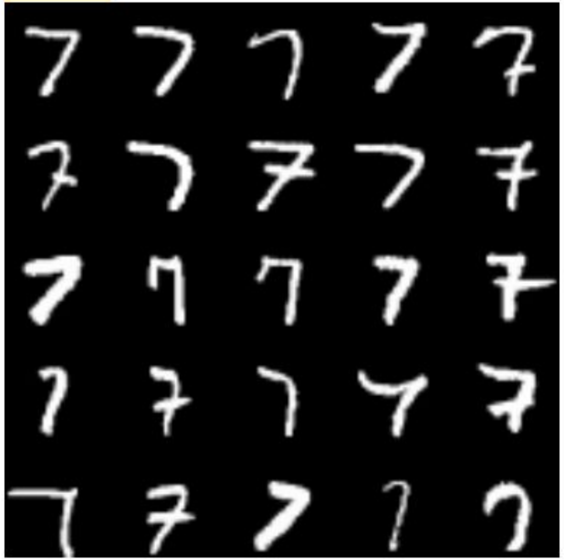
  $\Rightarrow$ *for normalized leaf distributions:* 1.0

- leafs over $X_2$ and $X_4$ output *EVI*
- feedforward evaluation (bottom-up)

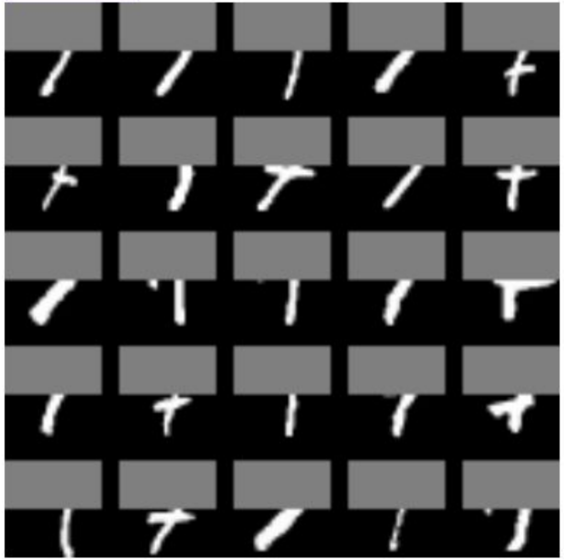


# Tractable MAR on PCs (Einsum Networks)

**EVI** $10,958.72\,\text{nats}$



**MAR** $5,387.55\,\text{nats}$



*Peharz et al., "Einsum Networks: Fast and Scalable Learning of Tractable Probabilistic Circuits", 2020*

# Smoothness + decomposability = ~~tractable MAP~~

We ***cannot*** decompose bottom-up a MAP query:

$$\max_{\mathbf{q}} p(\mathbf{q} \mid \mathbf{e})$$

since for a sum node we are marginalizing out a latent variable

$$\max_{\mathbf{q}} \sum_{i} w_i p_i(\mathbf{q}, \mathbf{e}) = \max_{\mathbf{q}} \sum_{\mathbf{z}} p(\mathbf{q}, \mathbf{z}, \mathbf{e}) \neq \sum_{\mathbf{z}} \max_{\mathbf{q}} p(\mathbf{q}, \mathbf{z}, \mathbf{e})$$

$\Longrightarrow$ *MAP for latent variable models is* **intractable** *[Conaty et al. 2017]*

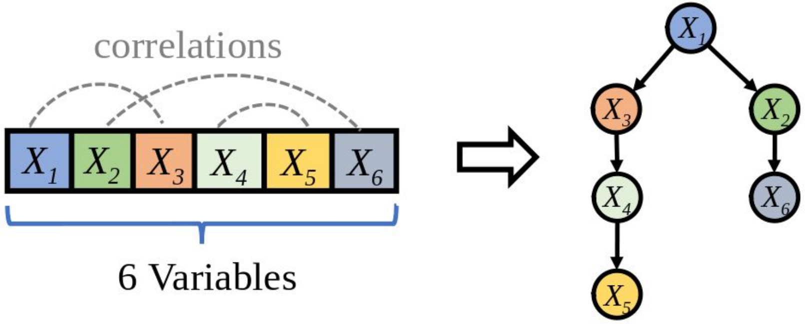
# Outline

1. What are tractable probabilistic circuits?
2. **Are these models any good?**
3. How far can we push tractable inference?
4. What is their expressive power?

# Learning Expressive Probabilistic Circuits

## Hidden Chow-Liu Trees

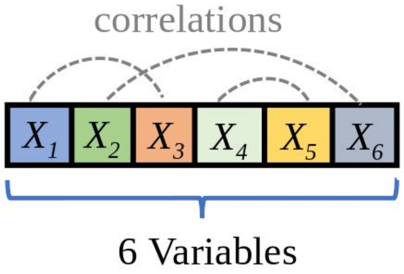
Learned **CLT structure** captures strong pairwise dependencies



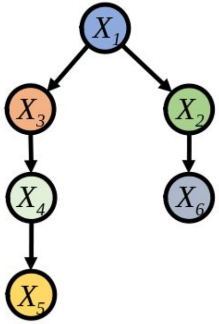
Anji Liu and Guy Van den Broeck. Tractable Regularization of Probabilistic Circuits, *NeurIPS*, 2021.

# Learning Expressive Probabilistic Circuits

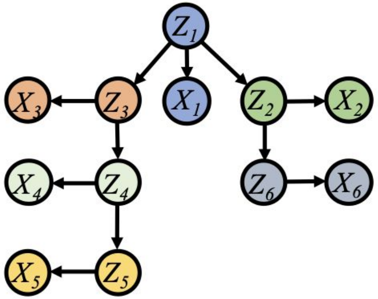
## Hidden Chow-Liu Trees



Learned **CLT structure** captures strong pairwise dependencies



Learned **HCLT structure**

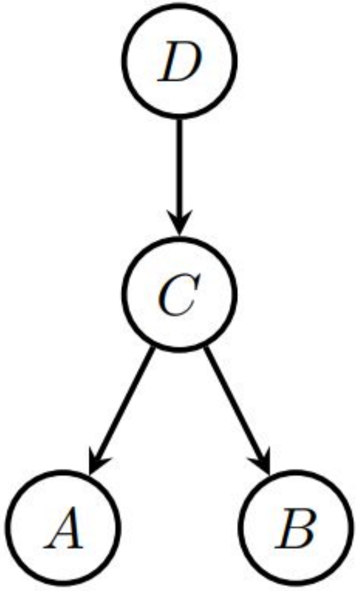


**Compile** into an equivalent PC

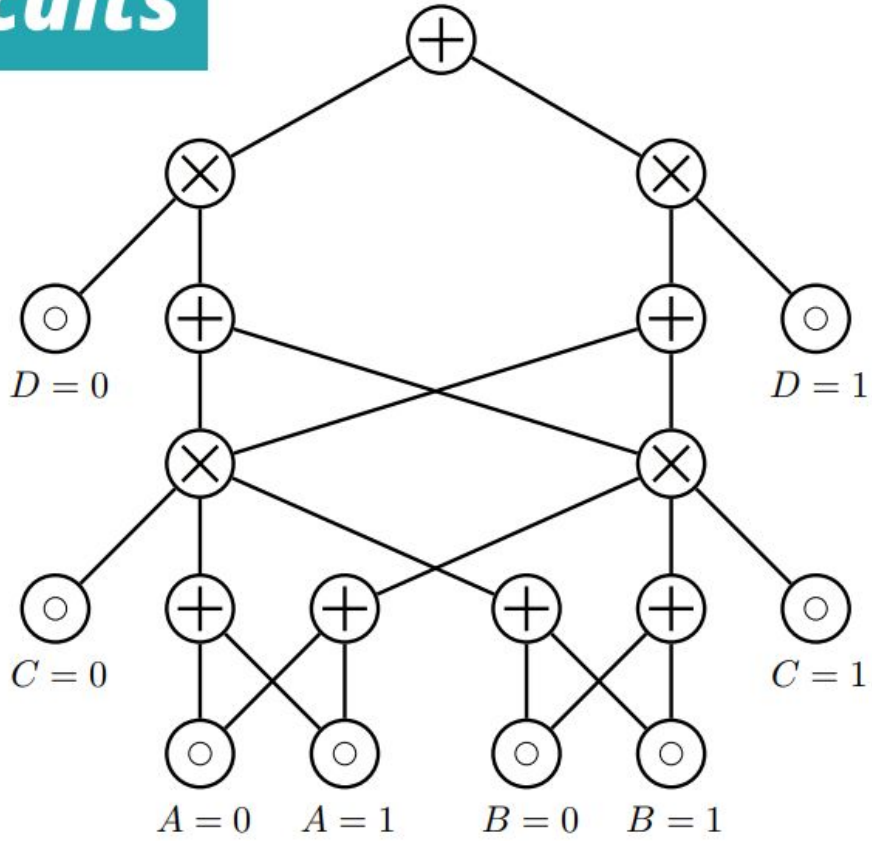
Anji Liu and Guy Van den Broeck. Tractable Regularization of Probabilistic Circuits, *NeurIPS*, 2021.

# From BN trees to circuits

*via compilation*

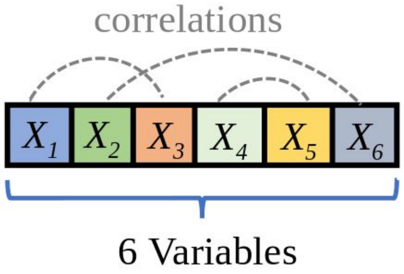


→

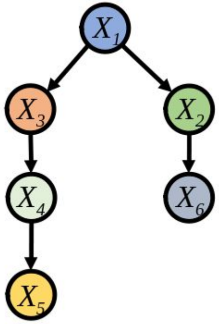


# Learning Expressive Probabilistic Circuits

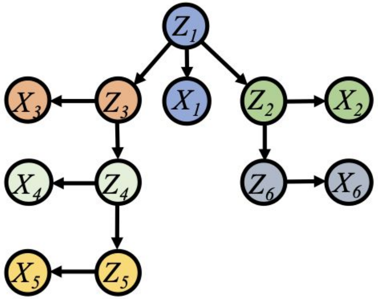
## Hidden Chow-Liu Trees



Learned **CLT structure** captures strong pairwise dependencies



Learned **HCLT structure**

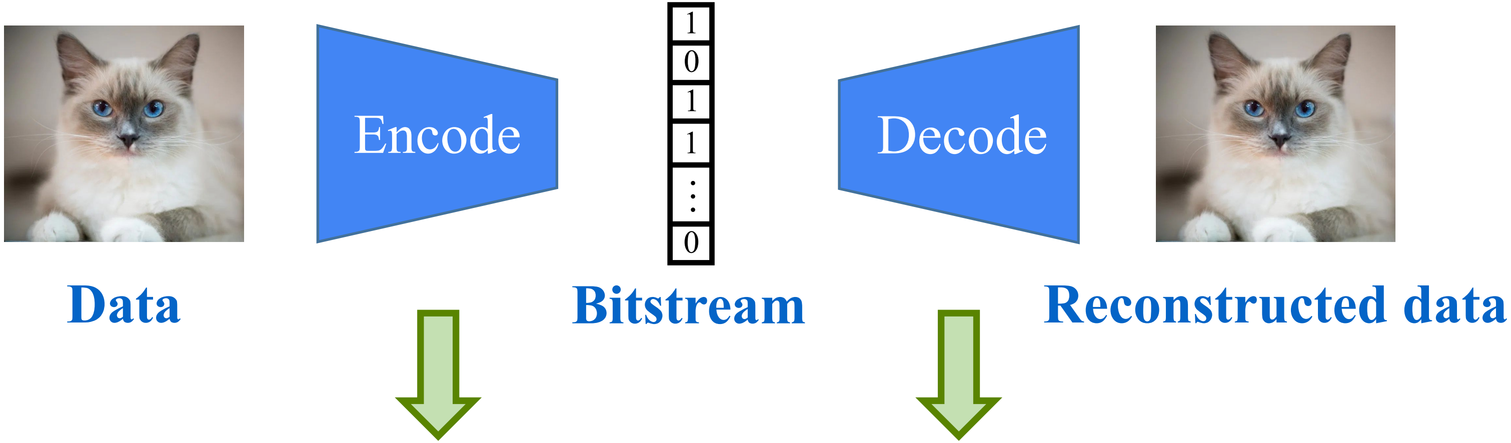


⇨ **Compile** into an equivalent PC

⇨ Mini-batch Stochastic **Expectation Maximization**

Anji Liu and Guy Van den Broeck. Tractable Regularization of Probabilistic Circuits, *NeurIPS*, 2021.

# Lossless Data Compression



Encode

Decode

1 0 1 1 ⋮ 0

Data

Bitstream

Reconstructed data

Expressive probabilistic model $p(\boldsymbol{x})$ ⇨ Determines the theoretical limit of compression rate

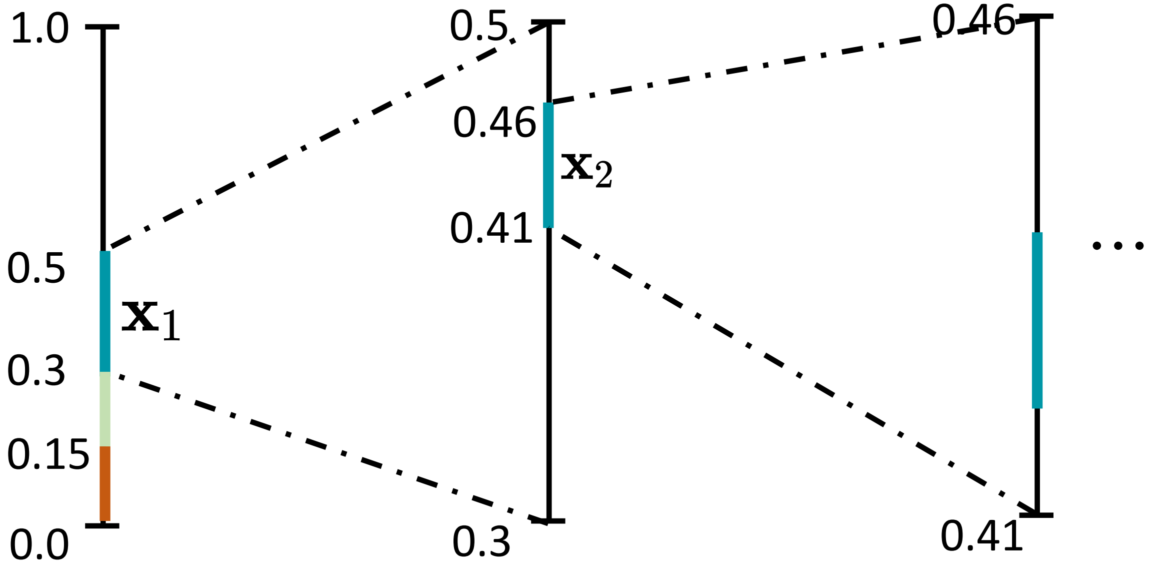
+

Efficient coding algorithm ⇨ How close we can approach the theoretical limit

Anji Liu, Stephan Mandt and Guy Van den Broeck. Lossless Compression with Probabilistic Circuits, 2021.

# A Typical Streaming Code – Arithmetic Coding

We want to compress a set of variables (e.g., pixels, letters) $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$



Compress $\mathbf{x}_1$ with $-\log p(\mathbf{x}_1)$ bits

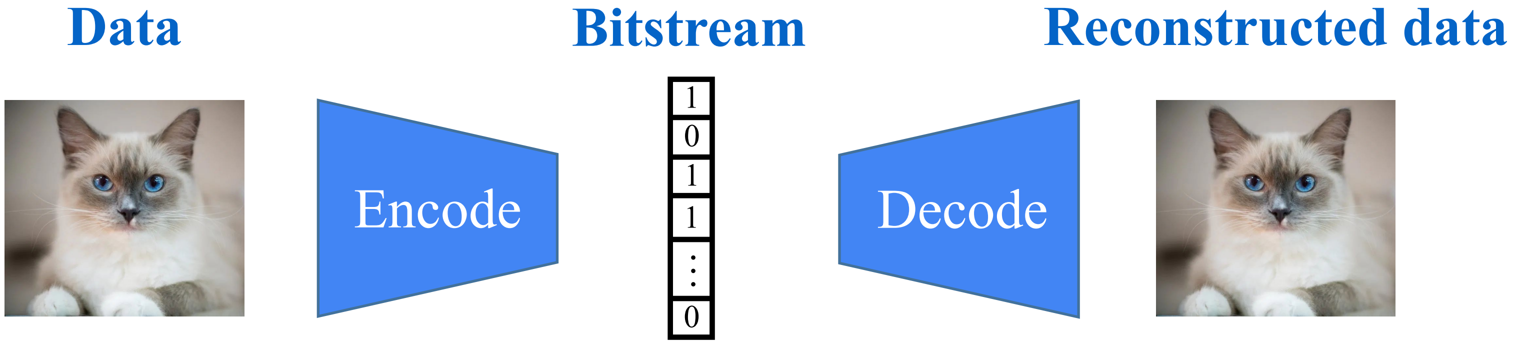
Compress $\mathbf{x}_2$ with $-\log p(\mathbf{x}_2|\mathbf{x}_1)$ bits

Compress $\mathbf{x}_3$ with $-\log p(\mathbf{x}_3|\mathbf{x}_1, \mathbf{x}_2)$ bits

Need to compute

$$p(X_1 < x_1)$$
$$p(X_1 \leq x_1)$$
$$p(X_2 < x_2|x_1)$$
$$p(X_2 \leq x_2|x_1)$$
$$p(X_3 < x_3|x_1, x_2)$$
$$p(X_3 \leq x_3|x_1, x_2)$$
$$\vdots$$

# Lossless Neural Compression with Probabilistic Circuits



**Data** **Bitstream** **Reconstructed data**

Encode

1
0
1
1
⋮
0

Decode

Probabilistic Circuits

- Expressive → SoTA likelihood on MNIST.
- Fast → Time complexity of en/decoding is **O( |p| log(D) )**, where D is the # variables and |p| is the size of the PC.

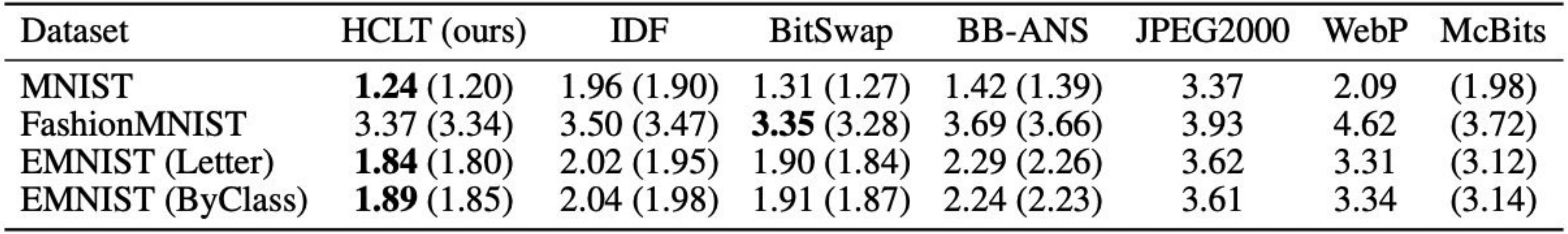
Arithmetic Coding:

$$p(X_1 < x_1)$$
$$p(X_1 \leq x_1)$$
$$p(X_2 < x_2|x_1)$$
$$p(X_2 \leq x_2|x_1)$$
$$p(X_3 < x_3|x_1, x_2)$$
$$p(X_3 \leq x_3|x_1, x_2)$$
$$\vdots$$

Anji Liu, Stephan Mandt and Guy Van den Broeck. Lossless Compression with Probabilistic Circuits, 2021.

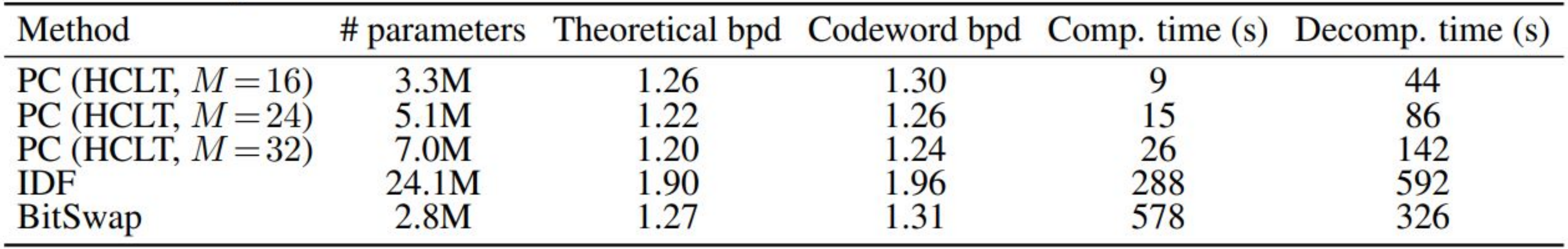
# Lossless Neural Compression with Probabilistic Circuits

SoTA compression rates



| Dataset | HCLT (ours) | IDF | BitSwap | BB-ANS | JPEG2000 | WebP | McBits |
|---|---|---|---|---|---|---|---|
| MNIST | **1.24** (1.20) | 1.96 (1.90) | 1.31 (1.27) | 1.42 (1.39) | 3.37 | 2.09 | (1.98) |
| FashionMNIST | 3.37 (3.34) | 3.50 (3.47) | **3.35** (3.28) | 3.69 (3.66) | 3.93 | 4.62 | (3.72) |
| EMNIST (Letter) | **1.84** (1.80) | 2.02 (1.95) | 1.90 (1.84) | 2.29 (2.26) | 3.62 | 3.31 | (3.12) |
| EMNIST (ByClass) | **1.89** (1.85) | 2.04 (1.98) | 1.91 (1.87) | 2.24 (2.23) | 3.61 | 3.34 | (3.14) |

Compress and decompress 5-40x faster than NN methods with similar bitrates

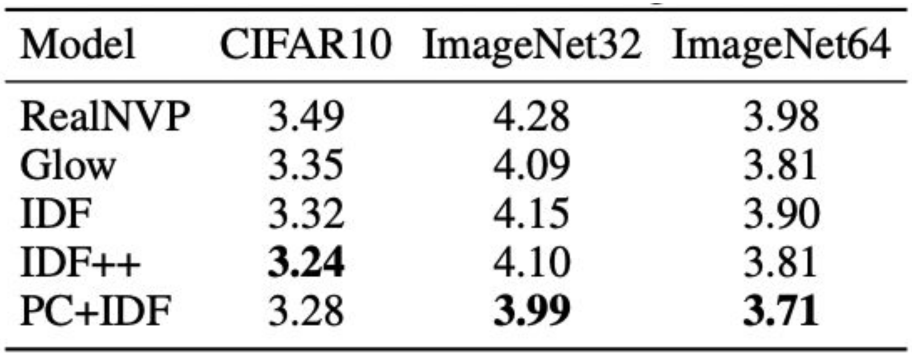


| Method | # parameters | Theoretical bpd | Codeword bpd | Comp. time (s) | Decomp. time (s) |
|---|---|---|---|---|---|
| PC (HCLT, $M=16$) | 3.3M | 1.26 | 1.30 | 9 | 44 |
| PC (HCLT, $M=24$) | 5.1M | 1.22 | 1.26 | 15 | 86 |
| PC (HCLT, $M=32$) | 7.0M | 1.20 | 1.24 | 26 | 142 |
| IDF | 24.1M | 1.90 | 1.96 | 288 | 592 |
| BitSwap | 2.8M | 1.27 | 1.31 | 578 | 326 |

Anji Liu, Stephan Mandt and Guy Van den Broeck. Lossless Compression with Probabilistic Circuits, 2021.

# Lossless Neural Compression with Probabilistic Circuits

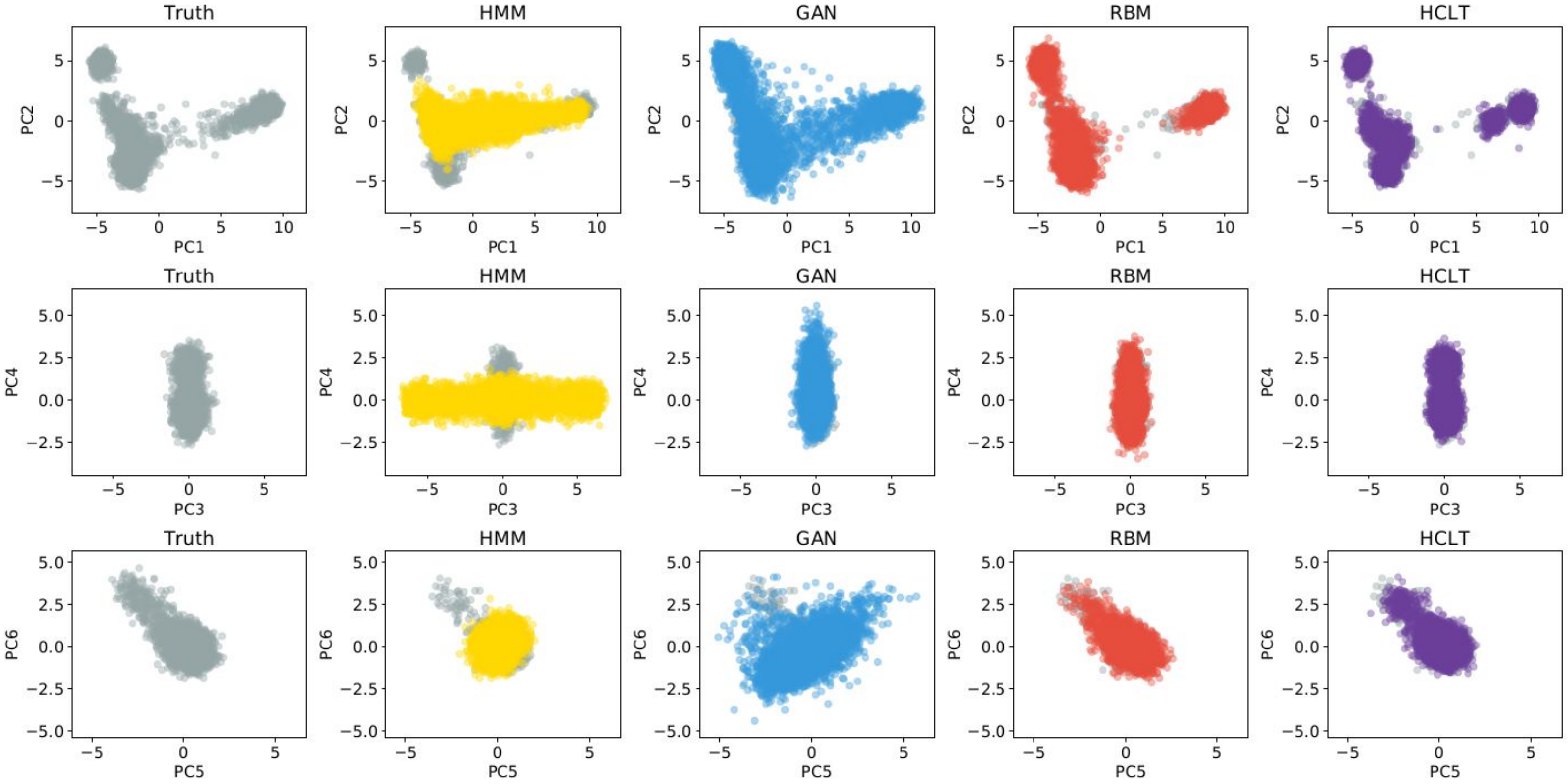
Can be effectively combined with Flow models to achieve better generative performance



| Model | CIFAR10 | ImageNet32 | ImageNet64 |
|---|---|---|---|
| RealNVP | 3.49 | 4.28 | 3.98 |
| Glow | 3.35 | 4.09 | 3.81 |
| IDF | 3.32 | 4.15 | 3.90 |
| IDF++ | **3.24** | 4.10 | 3.81 |
| PC+IDF | 3.28 | **3.99** | **3.71** |

Anji Liu, Stephan Mandt and Guy Van den Broeck. Lossless Compression with Probabilistic Circuits, 2021.

# Tractable and expressive generative models of genetic variation data



Meihua Dang , Anji Liu , Xinzhu Wei , Sriram Sankararaman, and Guy Van den Broeck, Tractable and expressive generative models of genetic variation data, RECOMB 2022

# *PC Learners keep getting better!* ***… stay tuned …***

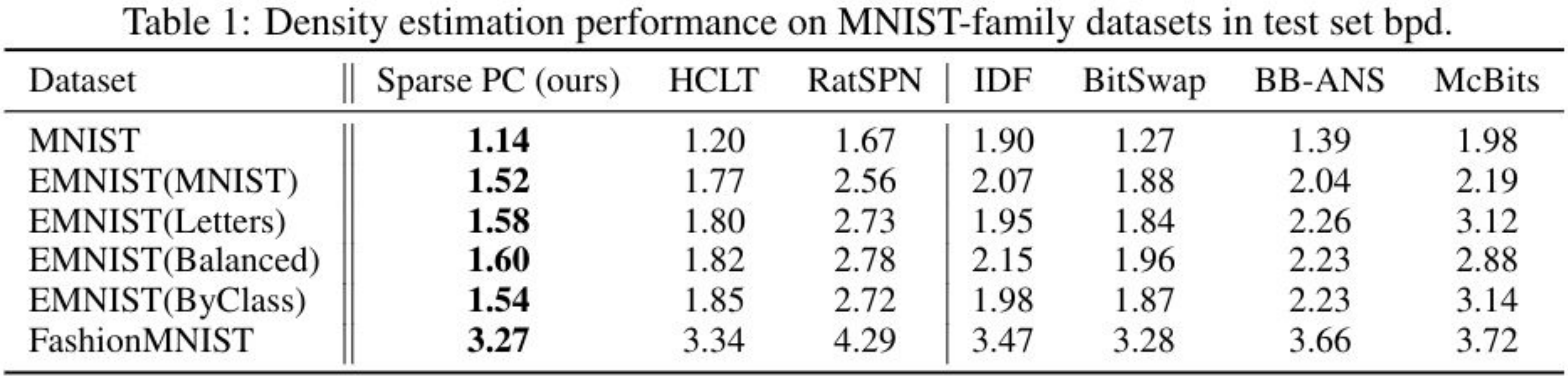
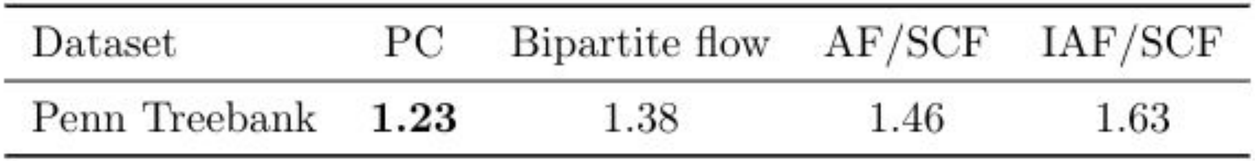


Table 1: Density estimation performance on MNIST-family datasets in test set bpd.

| Dataset | Sparse PC (ours) | HCLT | RatSPN | IDF | BitSwap | BB-ANS | McBits |
|---|---|---|---|---|---|---|---|
| MNIST | **1.14** | 1.20 | 1.67 | 1.90 | 1.27 | 1.39 | 1.98 |
| EMNIST(MNIST) | **1.52** | 1.77 | 2.56 | 2.07 | 1.88 | 2.04 | 2.19 |
| EMNIST(Letters) | **1.58** | 1.80 | 2.73 | 1.95 | 1.84 | 2.26 | 3.12 |
| EMNIST(Balanced) | **1.60** | 1.82 | 2.78 | 2.15 | 1.96 | 2.23 | 2.88 |
| EMNIST(ByClass) | **1.54** | 1.85 | 2.72 | 1.98 | 1.87 | 2.23 | 3.14 |
| FashionMNIST | **3.27** | 3.34 | 4.29 | 3.47 | 3.28 | 3.66 | 3.72 |

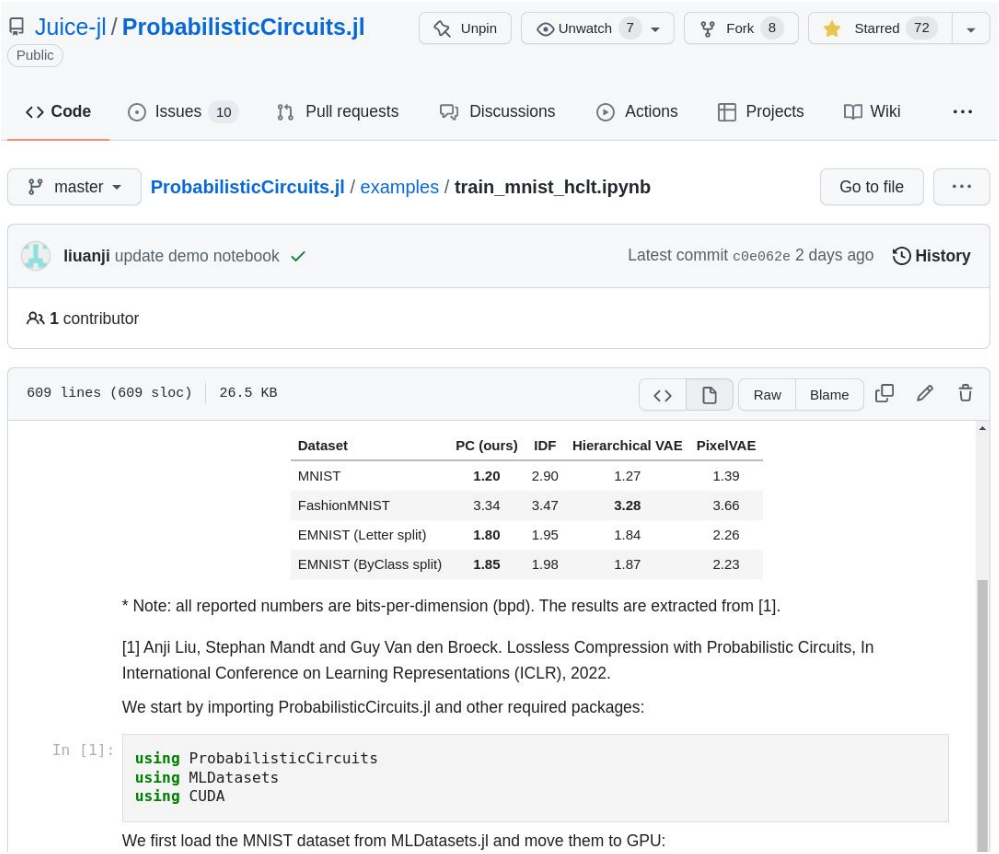


| Dataset | PC | Bipartite flow | AF/SCF | IAF/SCF |
|---|---|---|---|---|
| Penn Treebank | **1.23** | 1.38 | 1.46 | 1.63 |

work in progress by Honghua Zhang, Meihua Dang, Anji Liu

# Training SotA likelihood full MNIST probabilistic circuit model in ~7 minutes on GPU:

https://github.com/Juice-jl/ProbabilisticCircuits.jl/blob/master/examples/train_mnist_hclt.ipynb



ProbabilisticCircuits.jl / examples / train_mnist_hclt.ipynb

liuanji update demo notebook ✓ Latest commit c0e062e 2 days ago History

1 contributor

609 lines (609 sloc) 26.5 KB

| Dataset | PC (ours) | IDF | Hierarchical VAE | PixelVAE |
|---|---|---|---|---|
| MNIST | **1.20** | 2.90 | 1.27 | 1.39 |
| FashionMNIST | 3.34 | 3.47 | **3.28** | 3.66 |
| EMNIST (Letter split) | **1.80** | 1.95 | 1.84 | 2.26 |
| EMNIST (ByClass split) | **1.85** | 1.98 | 1.87 | 2.23 |

* Note: all reported numbers are bits-per-dimension (bpd). The results are extracted from [1].

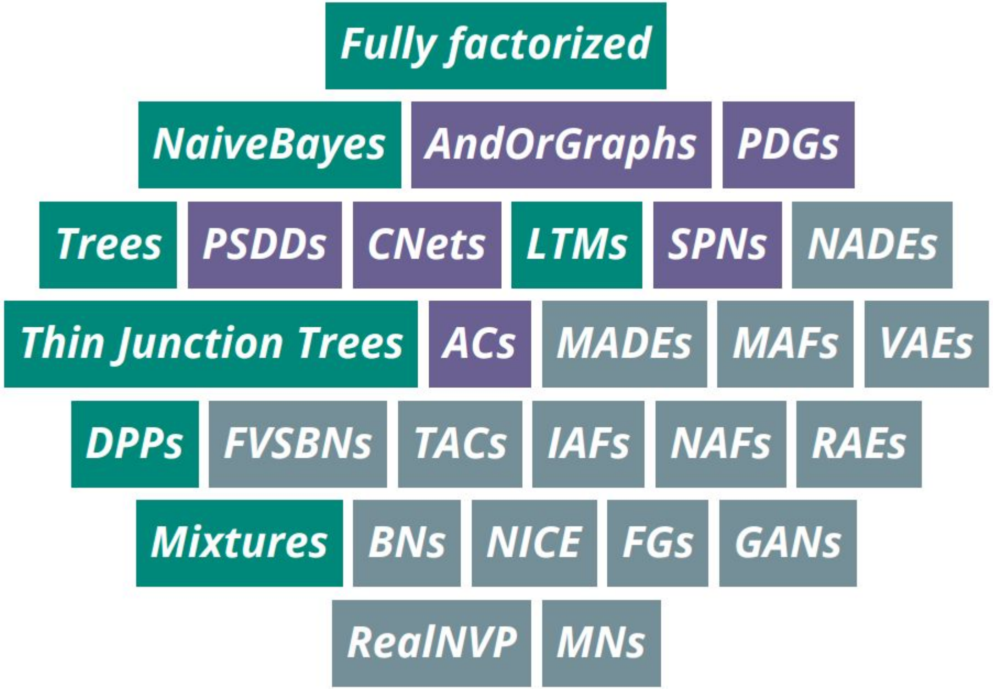
[1] Anji Liu, Stephan Mandt and Guy Van den Broeck. Lossless Compression with Probabilistic Circuits, In International Conference on Learning Representations (ICLR), 2022.

We start by importing ProbabilisticCircuits.jl and other required packages:

In [1]:

```julia
using ProbabilisticCircuits
using MLDatasets
using CUDA
```

We first load the MNIST dataset from MLDatasets.jl and move them to GPU:



Fully factorized

NaiveBayes AndOrGraphs PDGs

Trees PSDDs CNets LTMs SPNs NADEs

Thin Junction Trees ACs MADEs MAFs VAEs

DPPs FVSBNs TACs IAFs NAFs RAEs

Mixtures BNs NICE FGs GANs

RealNVP MNs

**Expressive models without compromises**

# Outline

1. What are tractable probabilistic circuits?
2. Are these models any good?
3. **How far can we push tractable inference?**
4. What is their expressive power?

# Smoothness + decomposability = ~~tractable MAP~~

We ***cannot*** decompose bottom-up a MAP query:

$$\max_{\mathbf{q}} p(\mathbf{q} \mid \mathbf{e})$$

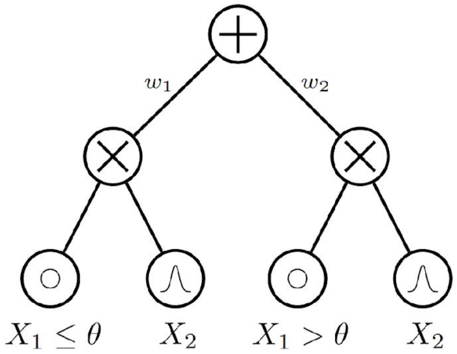
since for a sum node we are marginalizing out a latent variable

$$\max_{\mathbf{q}} \sum_{i} w_i p_i(\mathbf{q}, \mathbf{e}) = \max_{\mathbf{q}} \sum_{\mathbf{z}} p(\mathbf{q}, \mathbf{z}, \mathbf{e}) \neq \sum_{\mathbf{z}} \max_{\mathbf{q}} p(\mathbf{q}, \mathbf{z}, \mathbf{e})$$

$\Longrightarrow$ *MAP for latent variable models is* **intractable** *[Conaty et al. 2017]*

# Determinism

A sum node is ***deterministic*** if only one of its children outputs non-zero for any input



***deterministic circuit***

⇒ *allows **tractable MAP** inference*

$$argmax_{\boldsymbol{x}}\, p(\boldsymbol{x})$$

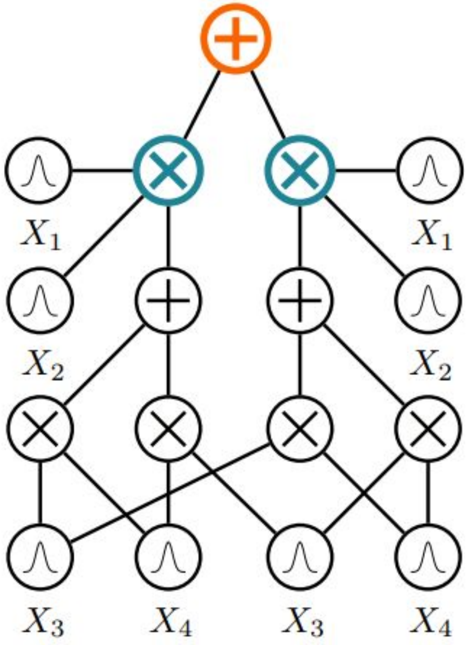
Darwiche and Marquis, “A Knowledge Compilation Map”, 2002

# Determinism + decomposability = tractable MAP

If $\boldsymbol{p}(\mathbf{q},\mathbf{e}) = \sum_i w_i \boldsymbol{p}_i(\mathbf{q},\mathbf{e}) = \max_i w_i \boldsymbol{p}_i(\mathbf{q},\mathbf{e})$,
(***deterministic*** sum node):

$$\begin{aligned}
\max_{\mathbf{q}} \boldsymbol{p}(\mathbf{q},\mathbf{e}) &= \max_{\mathbf{q}} \sum_i w_i \boldsymbol{p}_i(\mathbf{q},\mathbf{e}) \\
&= \max_{\mathbf{q}} \max_i w_i \boldsymbol{p}_i(\mathbf{q},\mathbf{e}) \\
&= \max_i \max_{\mathbf{q}} w_i \boldsymbol{p}_i(\mathbf{q},\mathbf{e})
\end{aligned}$$

$\Rightarrow$ *one non-zero child term, thus sum is max*

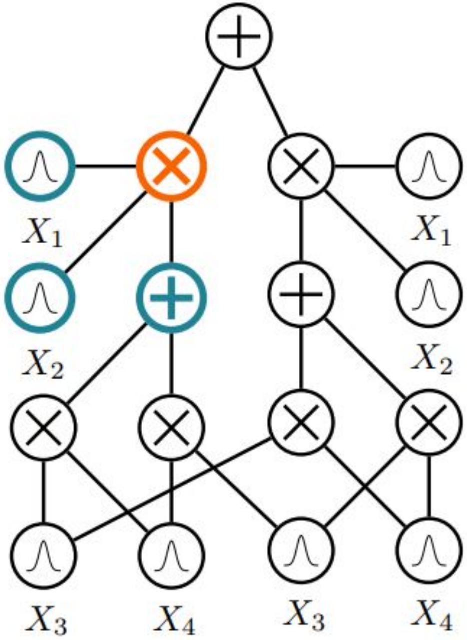


# Determinism + decomposability = tractable MAP

If $p(\mathbf{q}, \mathbf{e}) = p(\mathbf{q}_\mathbf{x}, \mathbf{e}_\mathbf{x}, \mathbf{q}_\mathbf{y}, \mathbf{e}_\mathbf{y}) = p(\mathbf{q}_\mathbf{x}, \mathbf{e}_\mathbf{x}) p(\mathbf{q}_\mathbf{y}, \mathbf{e}_\mathbf{y})$
(***decomposable*** product node):

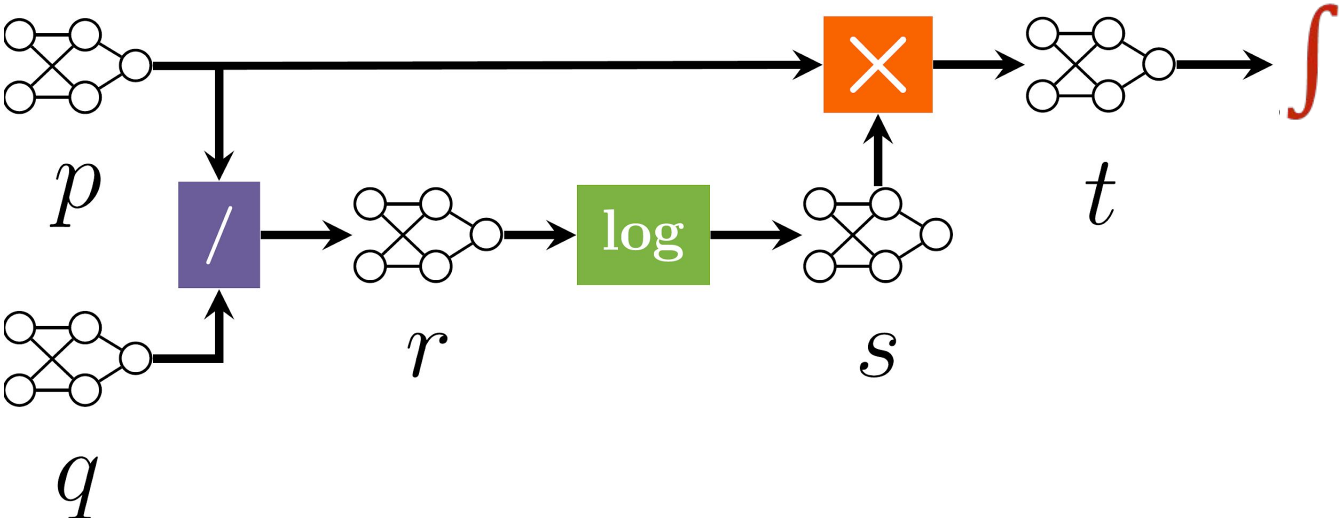
$$\begin{aligned}\max_{\mathbf{q}} p(\mathbf{q} \mid \mathbf{e}) &= \max_{\mathbf{q}} p(\mathbf{q}, \mathbf{e}) \\ &= \max_{\mathbf{q}_\mathbf{x}, \mathbf{q}_\mathbf{y}} p(\mathbf{q}_\mathbf{x}, \mathbf{e}_\mathbf{x}, \mathbf{q}_\mathbf{y}, \mathbf{e}_\mathbf{y}) \\ &= \max_{\mathbf{q}_\mathbf{x}} p(\mathbf{q}_\mathbf{x}, \mathbf{e}_\mathbf{x}) \cdot \max_{\mathbf{q}_\mathbf{y}} p(\mathbf{q}_\mathbf{y}, \mathbf{e}_\mathbf{y})\end{aligned}$$

$\Longrightarrow$ *solving optimization independently*



# Queries as pipelines: KLD

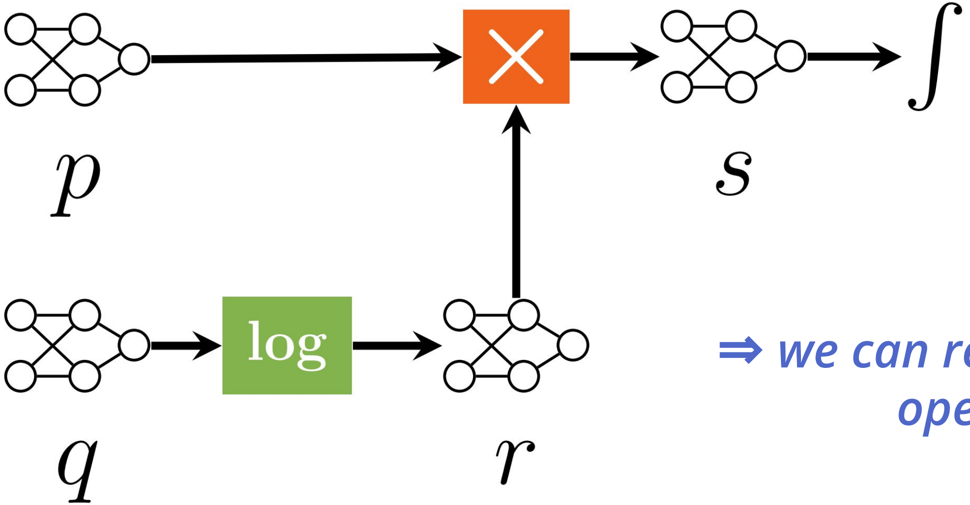
$$\mathbb{KLD}(p \parallel q) = \int p(\boldsymbol{x}) \times \log((p(\boldsymbol{x})/q(\boldsymbol{x}))d\boldsymbol{X}$$



Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso and Guy Van den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference, *NeurIPS*, 2021.

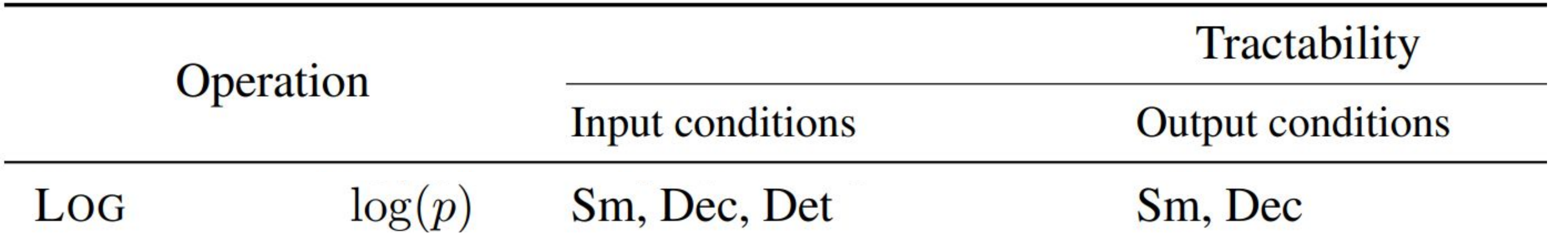
# Queries as pipelines: Cross Entropy

$$H(p,q) = \int p(\boldsymbol{x}) \times \log(q(\boldsymbol{x}))d\boldsymbol{X}$$

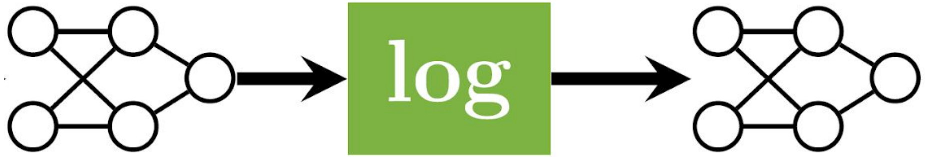


⇒ *we can reuse the operations!*

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso and Guy Van den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference, *NeurIPS*, 2021.



| Operation | | Tractability | |
|---|---|---|---|
| | | Input conditions | Output conditions |
| LOG | $\log(p)$ | Sm, Dec, Det | Sm, Dec |

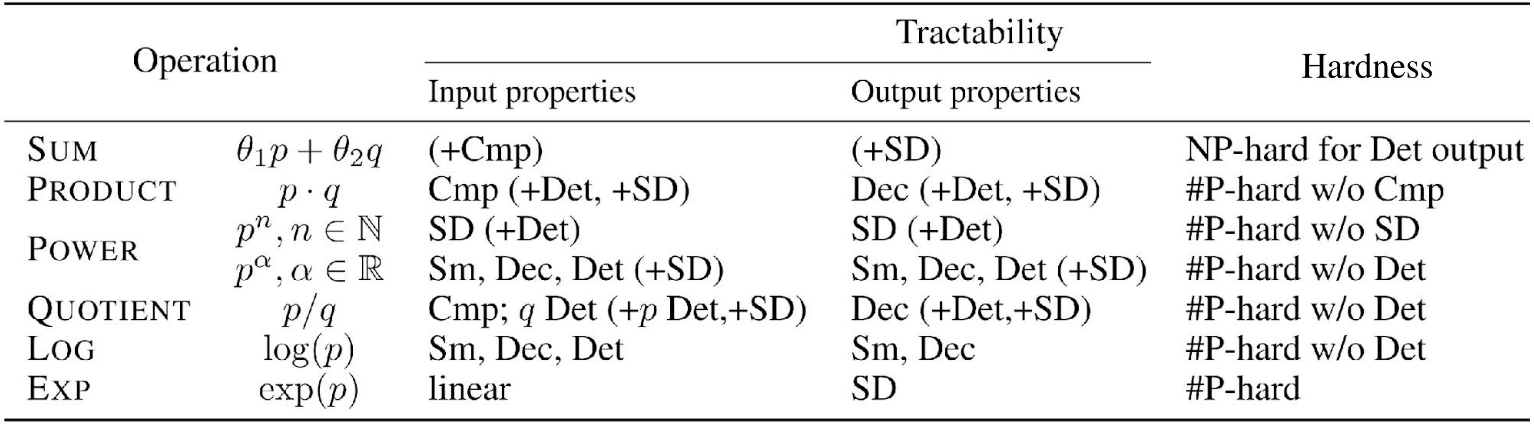


*smooth, decomposable, deterministic*

*smooth, decomposable*

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso and Guy Van den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference, *NeurIPS*, 2021.

# Tractable circuit operations

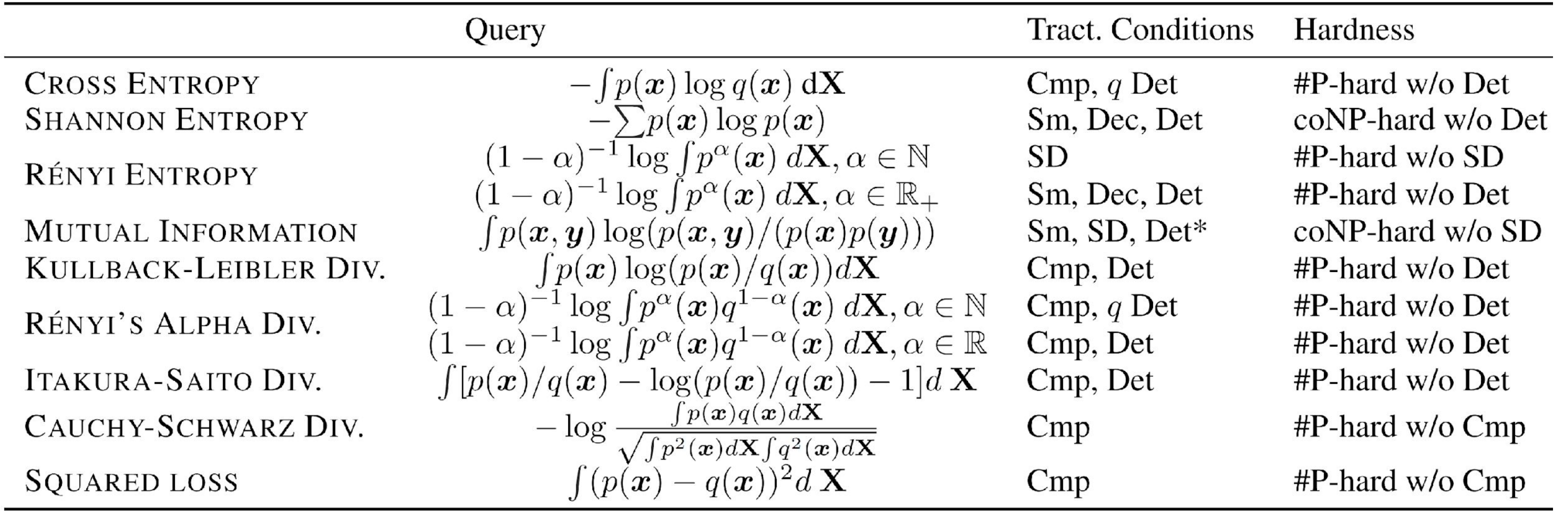


| Operation | | Tractability | | Hardness |
|---|---|---|---|---|
| | | Input properties | Output properties | |
| SUM | $\theta_1 p + \theta_2 q$ | (+Cmp) | (+SD) | NP-hard for Det output |
| PRODUCT | $p \cdot q$ | Cmp (+Det, +SD) | Dec (+Det, +SD) | #P-hard w/o Cmp |
| POWER | $p^n, n \in \mathbb{N}$ | SD (+Det) | SD (+Det) | #P-hard w/o SD |
| | $p^\alpha, \alpha \in \mathbb{R}$ | Sm, Dec, Det (+SD) | Sm, Dec, Det (+SD) | #P-hard w/o Det |
| QUOTIENT | $p/q$ | Cmp; $q$ Det (+$p$ Det,+SD) | Dec (+Det,+SD) | #P-hard w/o Det |
| LOG | $\log(p)$ | Sm, Dec, Det | Sm, Dec | #P-hard w/o Det |
| EXP | $\exp(p)$ | linear | SD | #P-hard |

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso and Guy Van den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference, *NeurIPS*, 2021.

# Inference by tractable operations

*systematically derive* tractable inference algorithm of complex queries



| | Query | Tract. Conditions | Hardness |
|---|---|---|---|
| Cross Entropy | $-\int p(\boldsymbol{x}) \log q(\boldsymbol{x})\, \mathrm{d}\mathbf{X}$ | Cmp, $q$ Det | #P-hard w/o Det |
| Shannon Entropy | $-\sum p(\boldsymbol{x}) \log p(\boldsymbol{x})$ | Sm, Dec, Det | coNP-hard w/o Det |
| Rényi Entropy | $(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x})\, d\mathbf{X}, \alpha \in \mathbb{N}$ | SD | #P-hard w/o SD |
| | $(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x})\, d\mathbf{X}, \alpha \in \mathbb{R}_+$ | Sm, Dec, Det | #P-hard w/o Det |
| Mutual Information | $\int p(\boldsymbol{x},\boldsymbol{y}) \log(p(\boldsymbol{x},\boldsymbol{y})/(p(\boldsymbol{x})p(\boldsymbol{y})))$ | Sm, SD, Det* | coNP-hard w/o SD |
| Kullback-Leibler Div. | $\int p(\boldsymbol{x}) \log(p(\boldsymbol{x})/q(\boldsymbol{x}))d\mathbf{X}$ | Cmp, Det | #P-hard w/o Det |
| Rényi's Alpha Div. | $(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x}) q^{1-\alpha}(\boldsymbol{x})\, d\mathbf{X}, \alpha \in \mathbb{N}$ | Cmp, $q$ Det | #P-hard w/o Det |
| | $(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x}) q^{1-\alpha}(\boldsymbol{x})\, d\mathbf{X}, \alpha \in \mathbb{R}$ | Cmp, Det | #P-hard w/o Det |
| Itakura-Saito Div. | $\int [p(\boldsymbol{x})/q(\boldsymbol{x}) - \log(p(\boldsymbol{x})/q(\boldsymbol{x})) - 1] d\,\mathbf{X}$ | Cmp, Det | #P-hard w/o Det |
| Cauchy-Schwarz Div. | $-\log \frac{\int p(\boldsymbol{x})q(\boldsymbol{x})d\mathbf{X}}{\sqrt{\int p^2(\boldsymbol{x})d\mathbf{X} \int q^2(\boldsymbol{x})d\mathbf{X}}}$ | Cmp | #P-hard w/o Cmp |
| Squared loss | $\int (p(\boldsymbol{x}) - q(\boldsymbol{x}))^2 d\,\mathbf{X}$ | Cmp | #P-hard w/o Cmp |

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso and Guy Van den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference, *NeurIPS*, 2021.

# Even harder queries

***Marginal MAP***

Given a set of query variables $\boldsymbol{Q} \subset \boldsymbol{X}$ and evidence $\boldsymbol{e}$,

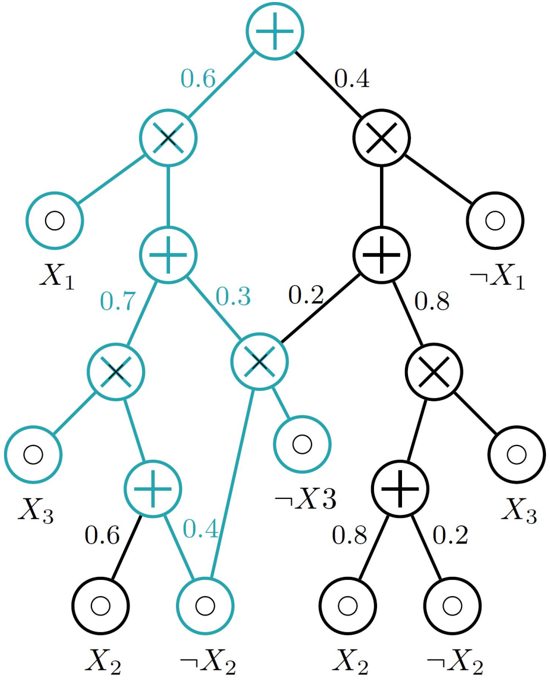
find: $argmax_{\mathbf{q}}\, p(\boldsymbol{q}|\boldsymbol{e})$

⇒ *i.e. MAP of a marginal distribution on* ***Q***

! ***NP$^{PP}$-complete*** *for PGMs*

! ***NP-hard*** *even for PCs tractable for marginals, MAP & entropy*

# Pruning circuits



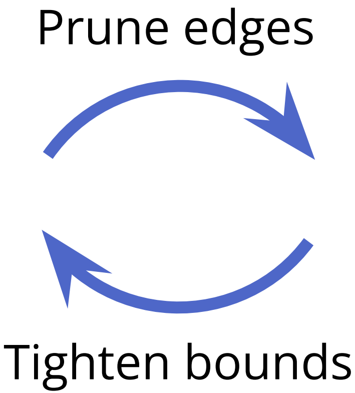
Any parts of circuit not relevant for MMAP state can be pruned away

e.g. $p(X_1 = 1, X_2 = 0)$

We can find such edges in *linear time*

YooJung Choi, Tal Friedman and Guy Van den Broeck. Solving Marginal MAP Exactly by Probabilistic Circuit Transformations, 2021.

# Iterative MMAP solver


Prune edges

Tighten bounds

|  | runtime (# solved) |  |
|---|---|---|
| **Dataset** | **search** | **pruning** |
| NLTCS | **0.01** (10) | 0.63 (10) |
| MSNBC | **0.03** (10) | 0.73 (10) |
| KDD | **0.04** (10) | 0.68 (10) |
| Plants | 2.95 (10) | **2.72** (10) |
| Audio | 2041.33 (6) | **13.70 (10)** |
| Jester | 2913.04 (2) | **14.74 (10)** |
| Netflix | – (0) | **47.18 (10)** |
| Accidents | 109.56 (10) | **15.86** (10) |
| Retail | **0.06** (10) | 0.81 (10) |
| Pumsb-star | 2208.27 (7) | **20.88 (10)** |
| DNA | – (0) | **505.75 (9)** |
| Kosarek | 48.74 (10) | **3.41** (10) |
| MSWeb | 1543.49 (10) | **1.28** (10) |
| Book | – (0) | **46.50 (10)** |
| EachMovie | – (0) | **1216.89 (8)** |
| WebKB | – (0) | **575.68 (10)** |
| Reuters-52 | – (0) | **120.58 (10)** |
| 20 NewsGrp. | – (0) | **504.52 (9)** |
| BBC | – (0) | **2757.18 (3)** |
| Ad | – (0) | **1254.37 (8)** |

YooJung Choi, Tal Friedman and Guy Van den Broeck. Solving Marginal MAP Exactly by Probabilistic Circuit Transformations, 2021.

# Probabilistic Sufficient Explanations

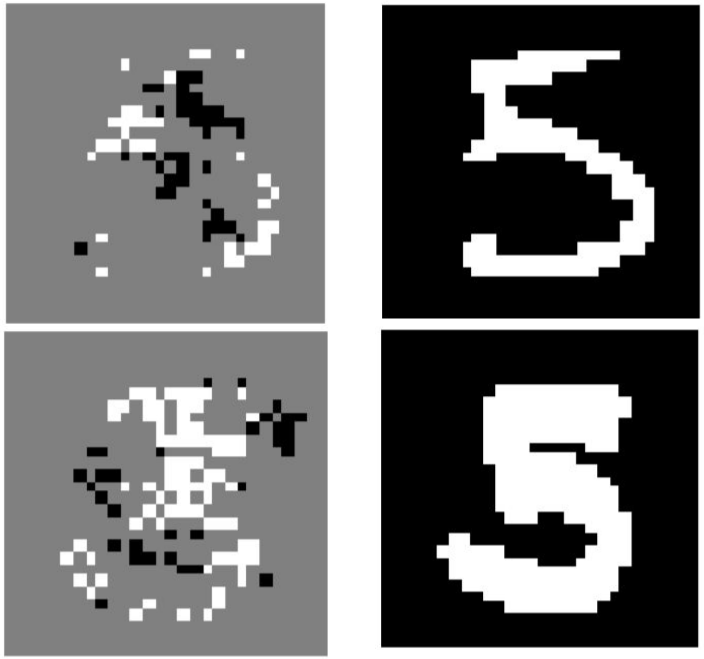
Goal: explain an instance of classification (a specific prediction)

Explanation is a subset of features, s.t.

1. The explanation is "probabilistically sufficient"

   *Under the feature distribution, given the explanation, the classifier is likely to make the observed prediction.*

2. It is minimal and "simple"



[Khosravi et al. IJCAI19, Wang et al. XXAI20]

# Model-Based Algorithmic Fairness: FairPC

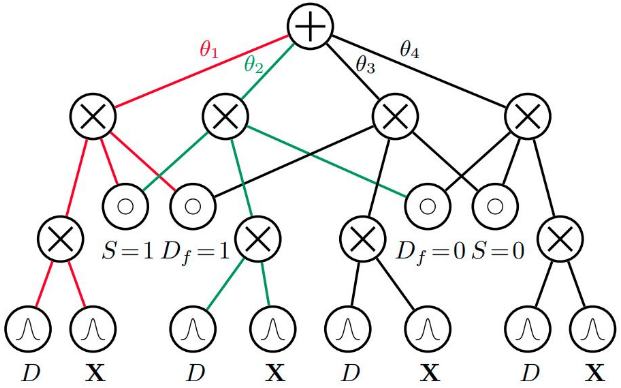
Learn classifier given

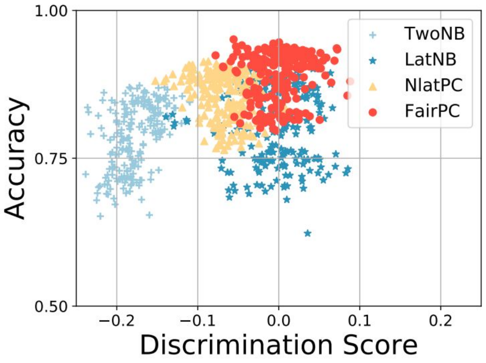
- features S and X
- training labels/decisions D

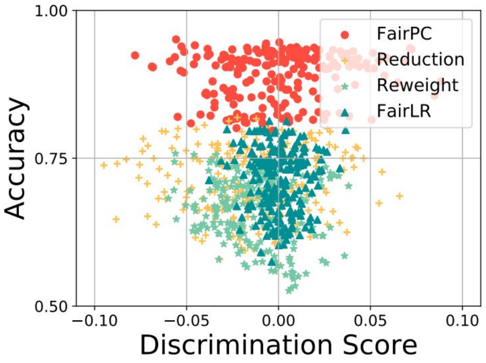
Group fairness by demographic parity:

*Fair decision $D_f$ should be independent of the sensitive attribute S*

Discover the **latent fair decision** $D_f$ by learning a PC.

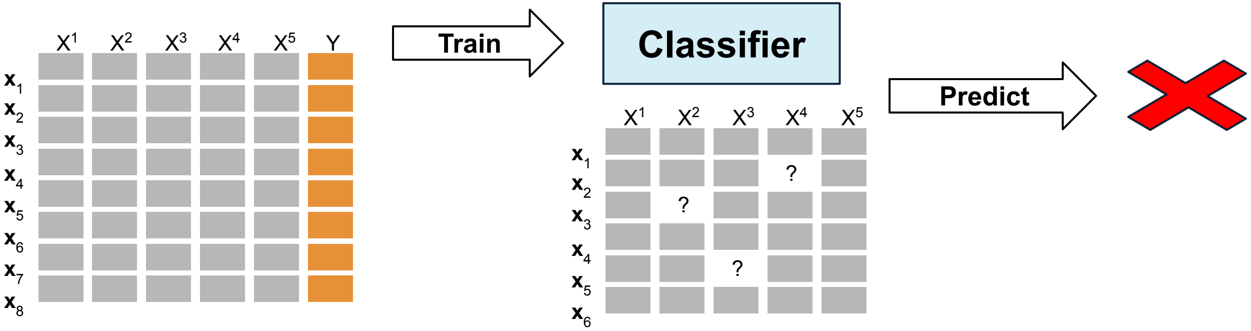






[Choi et al. AAAI21]

# Prediction with Missing Features



Test with missing features

See work on

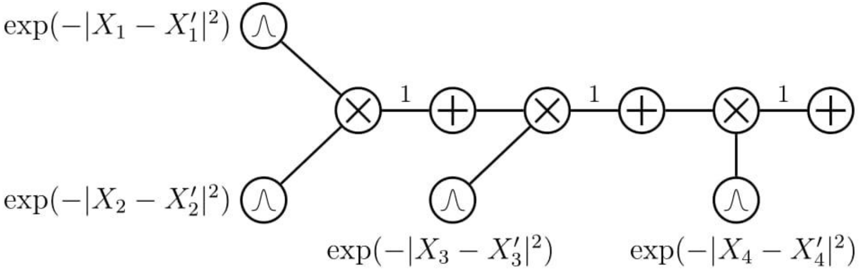
- Expected predictions / conformant learning [Khosravi et al.]
- Generative forests [Correia et al.]

# Tractable Computation of Expected Kernels

- How to compute the expected kernel given two distributions **p**, **q**?

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{p},\mathbf{x}'\sim\mathbf{q}}[\mathbf{k}(\mathbf{x},\mathbf{x}')]$$

- Circuit representation for kernel functions, e.g., $\mathbf{k}(\mathbf{x},\mathbf{x}') = \exp\left(-\sum_{i=1}^{4} \mid X_i - X_i' \mid^2\right)$



Wenzhe Li, Zhe Zeng, Antonio Vergari and Guy Van den Broeck. Tractable Computation of Expected Kernels, *UAI*, 2021.

# Tractable Computation of Expected Kernels: Applications

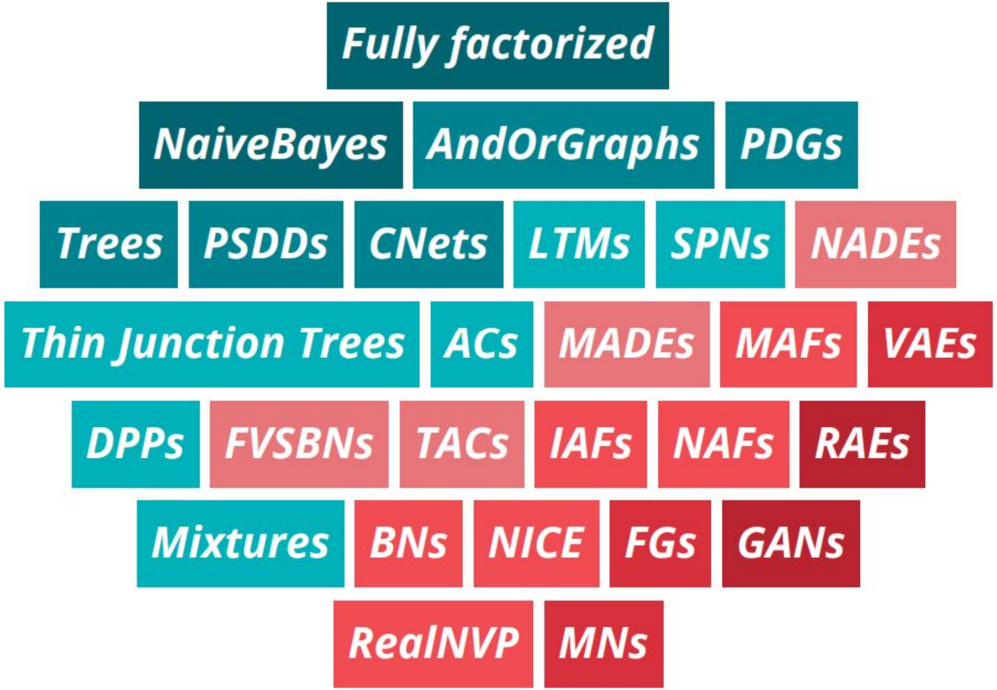
- Reasoning about support vector regression (SVR) with missing features

$$\underset{\substack{\downarrow \\ \text{missing} \\ \text{features}}}{\mathbb{E}_{\mathbf{x}_m \sim \mathbf{p}(\mathbf{X}_m | \mathbf{x}_o)}} \left[ \underbrace{\sum_{i=1}^{m} w_i \mathbf{k}(\mathbf{x}_i, \mathbf{x}) + b}_{\text{SVR model}} \right]$$

- Collapsed Black-box Importance Sampling: minimize kernelized Stein discrepancy

importance weights $$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}} \left\{ \boldsymbol{w}^\top \underset{\substack{\downarrow \\ \text{expected kernel} \\ \text{matrix}}}{\boldsymbol{K}_{p,\mathbf{s}}} \boldsymbol{w} \;\middle|\; \sum_{i=1}^{n} w_i = 1,\ w_i \geq 0 \right\}$$

Wenzhe Li, Zhe Zeng, Antonio Vergari and Guy Van den Broeck. Tractable Computation of Expected Kernels, *UAI*, 2021.



Fully factorized

NaiveBayes AndOrGraphs PDGs

Trees PSDDs CNets LTMs SPNs NADEs

Thin Junction Trees ACs MADEs MAFs VAEs

DPPs FVSBNs TACs IAFs NAFs RAEs

Mixtures BNs NICE FGs GANs

RealNVP MNs

# tractability is a spectrum

# Outline

1. What are tractable probabilistic circuits?
2. Are these models any good?
3. How far can we push tractable inference?
4. **What is their expressive power?**

Probabilistic circuits seem awfully general.

*Are all tractable probabilistic models probabilistic circuits?*

# Enter: Determinantal Point Processes (DPPs)

DPPs are models where probabilities are specified by (sub)determinants

$$L = \begin{bmatrix} 1 & 0.9 & 0.8 & 0 \\ 0.9 & 0.97 & 0.96 & 0 \\ 0.8 & 0.96 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

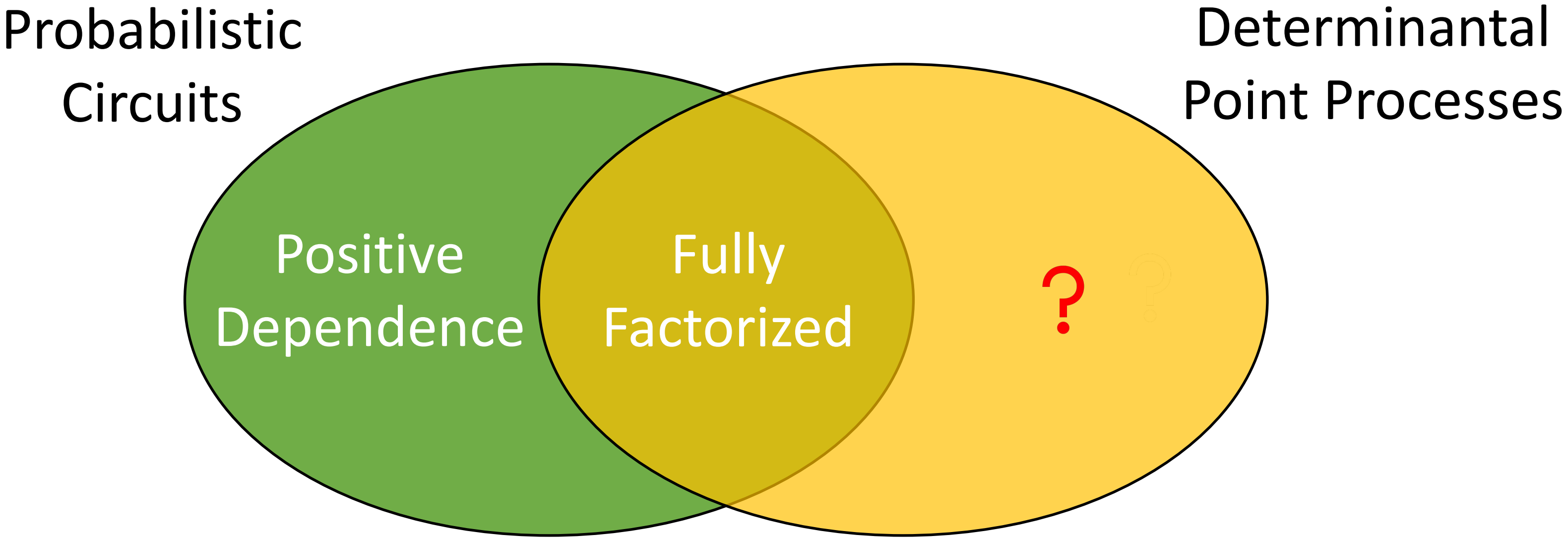
Tractable likelihoods and marginals

Global Negative Dependence

Diversity in recommendation systems

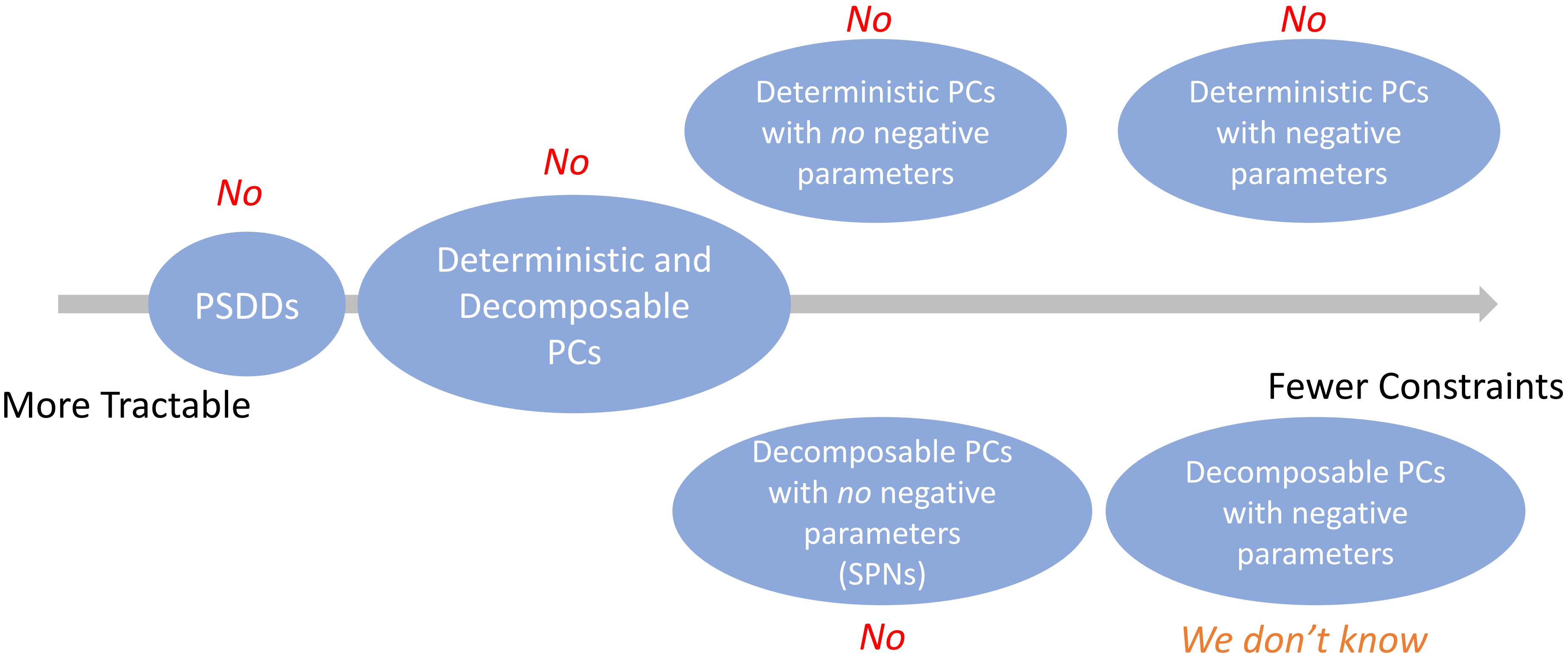
$$\Pr_L(X_1 = 1, X_2 = 0, X_3 = 1, X_4 = 0) = \frac{1}{\det(L + I)} \det(L_{\{1,2\}})$$

# Relationship between PCs and DPPs



Honghua Zhang, Steven Holtzen and Guy Van den Broeck. On the Relationship Between Probabilistic Circuits and Determinantal Point Processes, *UAI*, 2020.

# We cannot tractably represent DPPs with subclasses of PCs



More Tractable

Fewer Constraints

PSDDs

*No*

Deterministic and Decomposable PCs

*No*

Deterministic PCs with *no* negative parameters

*No*

Deterministic PCs with negative parameters

*No*

Decomposable PCs with *no* negative parameters (SPNs)

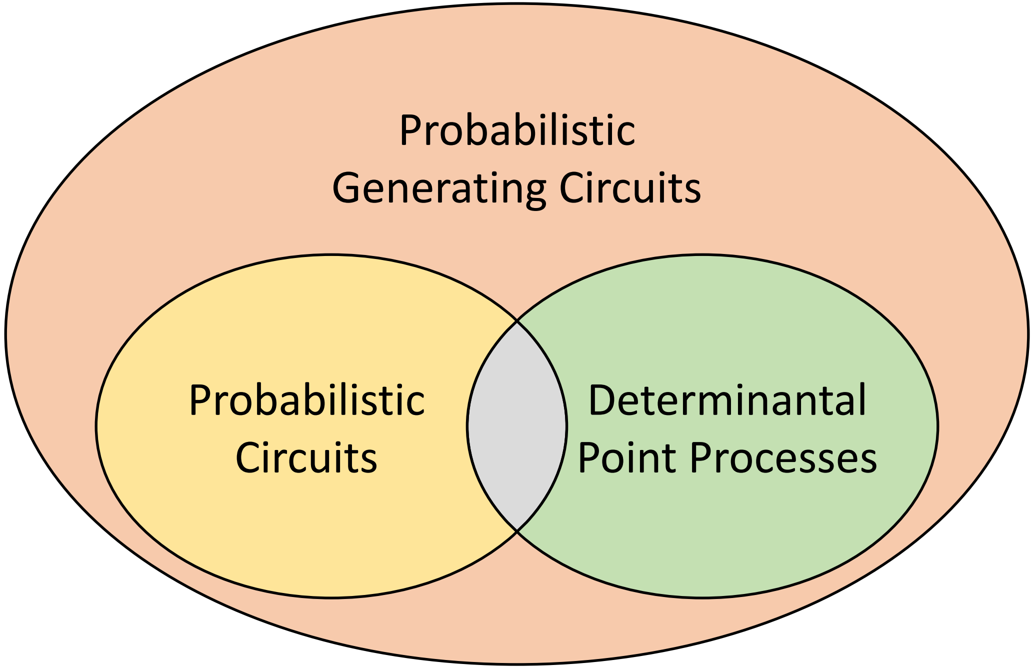
*No*

Decomposable PCs with negative parameters

*We don't know*

Honghua Zhang, Steven Holtzen and Guy Van den Broeck. On the Relationship Between Probabilistic Circuits and Determinantal Point Processes, *UAI*, 2020.

# Probabilistic Generating Circuits


Probabilistic Generating Circuits
Probabilistic Circuits
Determinantal Point Processes

A Tractable Unifying Framework for PCs and DPPs

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# Probability Generating Functions

| $X_1$ | $X_2$ | $X_3$ | $\Pr_\beta$ |
|---|---|---|---|
| 0 | 0 | 0 | 0.02 |
| 0 | 0 | 1 | 0.08 |
| 0 | 1 | 0 | 0.12 |
| 0 | 1 | 1 | 0.48 |
| 1 | 0 | 0 | 0.02 |
| 1 | 0 | 1 | 0.08 |
| 1 | 1 | 0 | 0.04 |
| 1 | 1 | 1 | 0.16 |

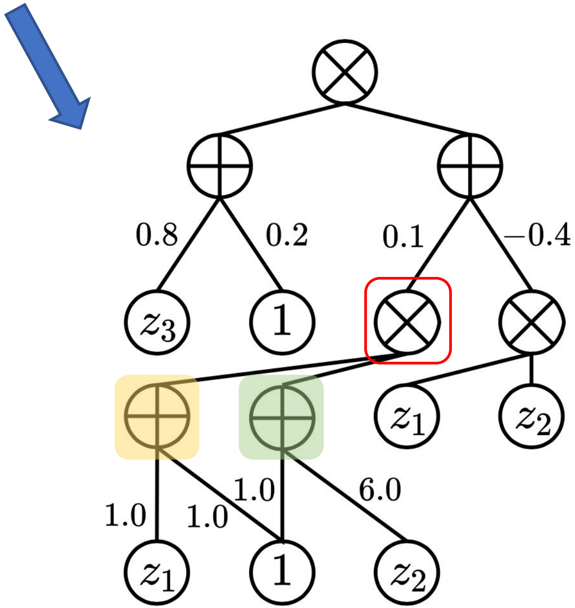
$$g_\beta = \boxed{0.16 z_1 z_2 z_3} + 0.04 z_1 z_2 + 0.08 z_1 z_3 + 0.02 z_1 \\ + 0.48 z_2 z_3 + 0.12 z_2 + 0.08 z_3 + 0.02.$$

$$g_\beta = (0.1(z_1 + 1)(6 z_2 + 1) - 0.4 z_1 z_2)(0.8 z_3 + 0.2)$$

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# Probabilistic Generating Circuits (PGCs)

$$g_\beta = (0.1\boxed{(z_1+1)(6z_2+1)} - 0.4z_1z_2)(0.8z_3+0.2)$$

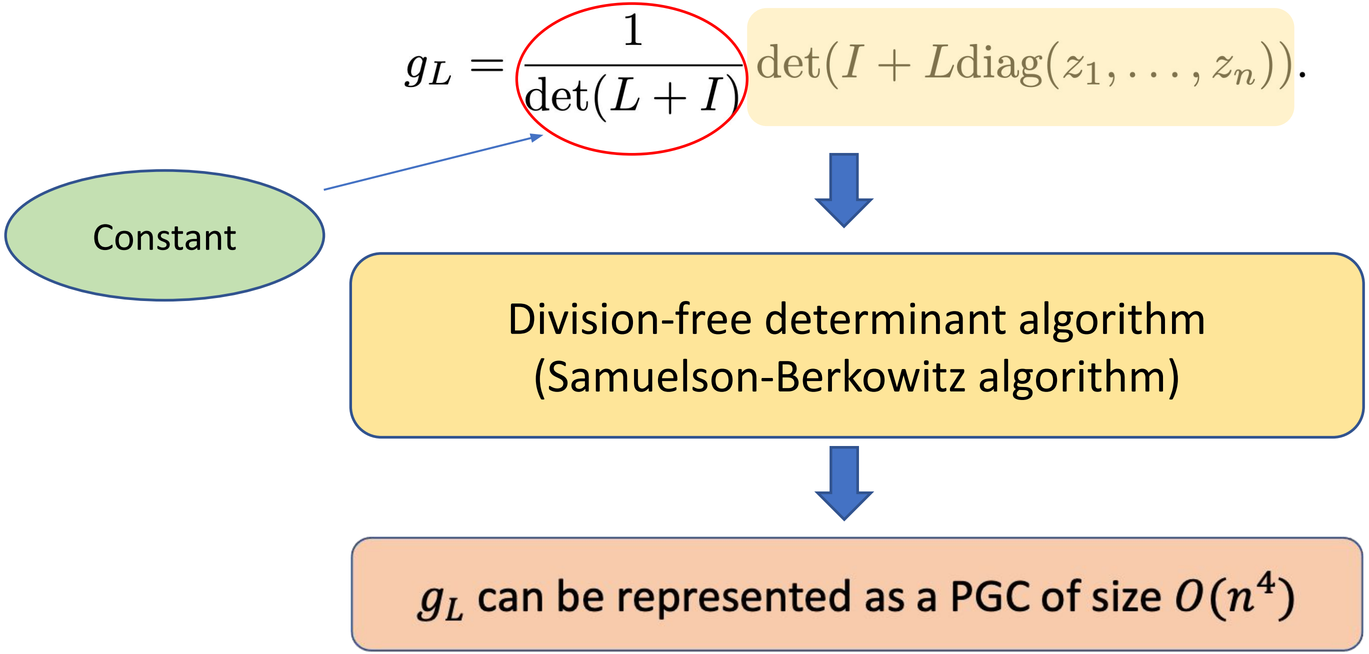


1. Sum nodes ⊕ with weighted edges to children.
2. Product nodes ⊗ with unweighted edges to children.
3. Leaf nodes: z_i or constant.

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# DPPs as PGCs

The generating polynomial for a DPP with kernel $L$ is given by:



$$g_L = \frac{1}{\det(L+I)} \det(I + L\mathrm{diag}(z_1, \ldots, z_n)).$$

Constant

Division-free determinant algorithm (Samuelson-Berkowitz algorithm)

$g_L$ can be represented as a PGC of size $O(n^4)$

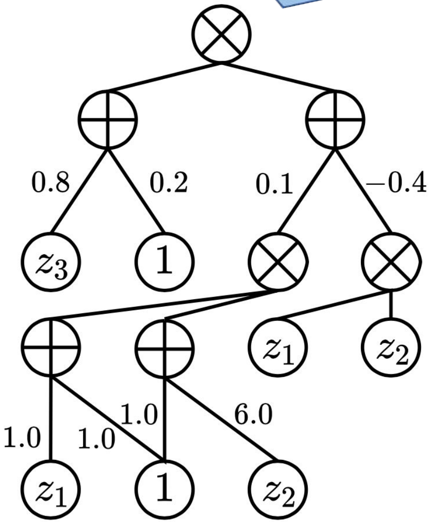
Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# PGCs Support Tractable Likelihoods/Marginals

Purely symbolic

$$z_i = \begin{cases} t, & X_i = 1 \\ 0, & X_i = 0 \\ 1, & otherwise \end{cases}$$

$\Pr(X_1 = 1, X_2 = 0, \dots) = ?$



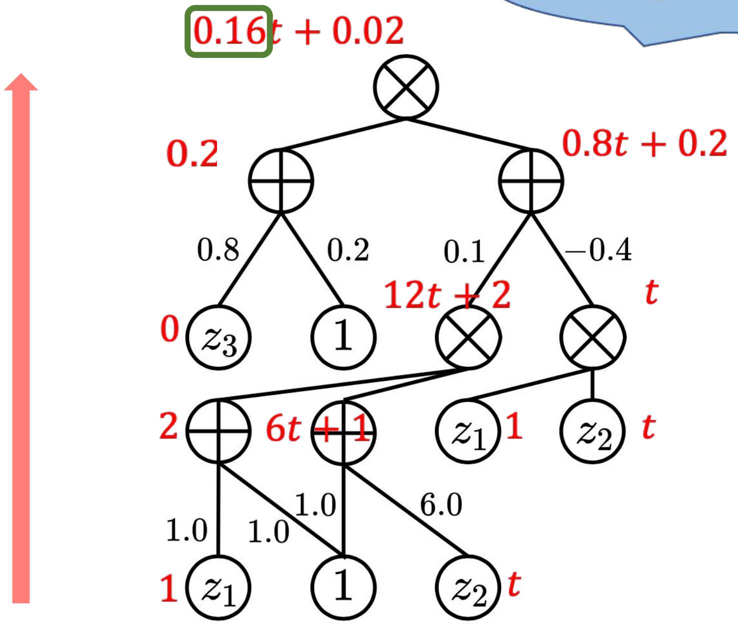
$p(t) = \alpha_k t^k + \cdots + \alpha_1 t$

$\alpha_k$ gives the answer

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# Example

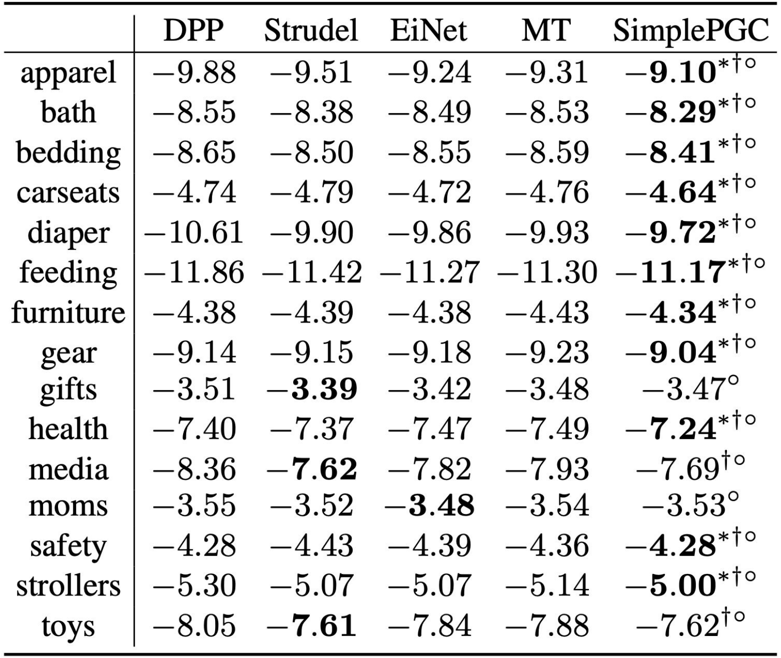
$\Pr(X_2 = 1, X_3 = 0) = ?$



| $X_1$ | $X_2$ | $X_3$ | $\Pr_\beta$ |
|---|---|---|---|
| 0 | 0 | 0 | 0.02 |
| 0 | 0 | 1 | 0.08 |
| 0 | 1 | 0 | 0.12 |
| 0 | 1 | 1 | 0.48 |
| 1 | 0 | 0 | 0.02 |
| 1 | 0 | 1 | 0.08 |
| 1 | 1 | 0 | 0.04 |
| 1 | 1 | 1 | 0.16 |

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# Experiment Results: Amazon Baby Registries



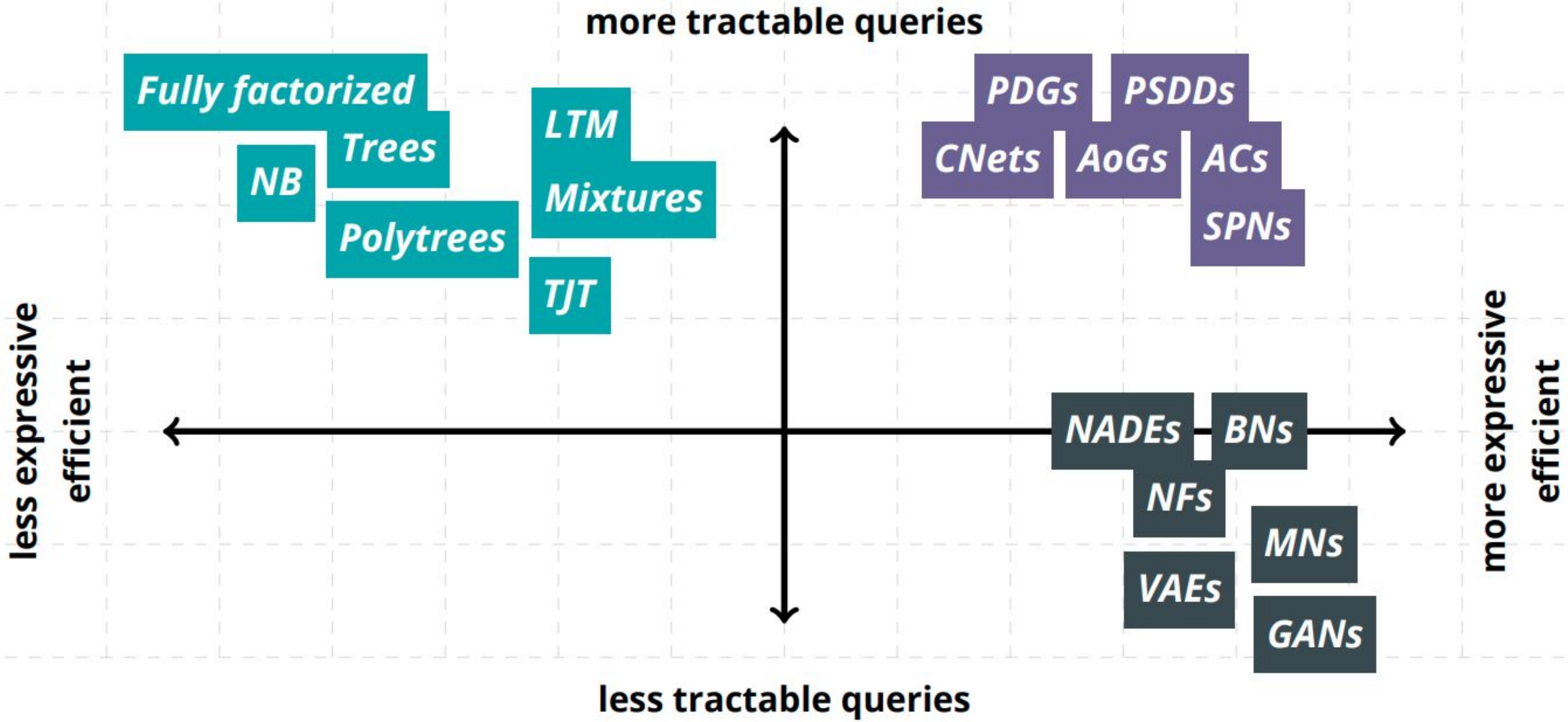
| | DPP | Strudel | EiNet | MT | SimplePGC |
|---|---|---|---|---|---|
| apparel | $-9.88$ | $-9.51$ | $-9.24$ | $-9.31$ | $\mathbf{-9.10}^{*\dagger\circ}$ |
| bath | $-8.55$ | $-8.38$ | $-8.49$ | $-8.53$ | $\mathbf{-8.29}^{*\dagger\circ}$ |
| bedding | $-8.65$ | $-8.50$ | $-8.55$ | $-8.59$ | $\mathbf{-8.41}^{*\dagger\circ}$ |
| carseats | $-4.74$ | $-4.79$ | $-4.72$ | $-4.76$ | $\mathbf{-4.64}^{*\dagger\circ}$ |
| diaper | $-10.61$ | $-9.90$ | $-9.86$ | $-9.93$ | $\mathbf{-9.72}^{*\dagger\circ}$ |
| feeding | $-11.86$ | $-11.42$ | $-11.27$ | $-11.30$ | $\mathbf{-11.17}^{*\dagger\circ}$ |
| furniture | $-4.38$ | $-4.39$ | $-4.38$ | $-4.43$ | $\mathbf{-4.34}^{*\dagger\circ}$ |
| gear | $-9.14$ | $-9.15$ | $-9.18$ | $-9.23$ | $\mathbf{-9.04}^{*\dagger\circ}$ |
| gifts | $-3.51$ | $\mathbf{-3.39}$ | $-3.42$ | $-3.48$ | $-3.47^{\circ}$ |
| health | $-7.40$ | $-7.37$ | $-7.47$ | $-7.49$ | $\mathbf{-7.24}^{*\dagger\circ}$ |
| media | $-8.36$ | $\mathbf{-7.62}$ | $-7.82$ | $-7.93$ | $-7.69^{\dagger\circ}$ |
| moms | $-3.55$ | $-3.52$ | $\mathbf{-3.48}$ | $-3.54$ | $-3.53^{\circ}$ |
| safety | $-4.28$ | $-4.43$ | $-4.39$ | $-4.36$ | $\mathbf{-4.28}^{*\dagger\circ}$ |
| strollers | $-5.30$ | $-5.07$ | $-5.07$ | $-5.14$ | $\mathbf{-5.00}^{*\dagger\circ}$ |
| toys | $-8.05$ | $\mathbf{-7.61}$ | $-7.84$ | $-7.88$ | $-7.62^{\dagger\circ}$ |

SimplePGC achieves SOTA result on 11/15 datasets

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

# Conclusion

1. What are tractable probabilistic circuits?
2. Are these models any good?
3. How far can we push tractable inference?
4. What is their expressive power?



more tractable queries

Fully factorized

Trees

NB

Polytrees

LTM

Mixtures

TJT

PDGs

PSDDs

CNets

AoGs

ACs

SPNs

less expressive
efficient

more expressive
efficient

NADEs

BNs

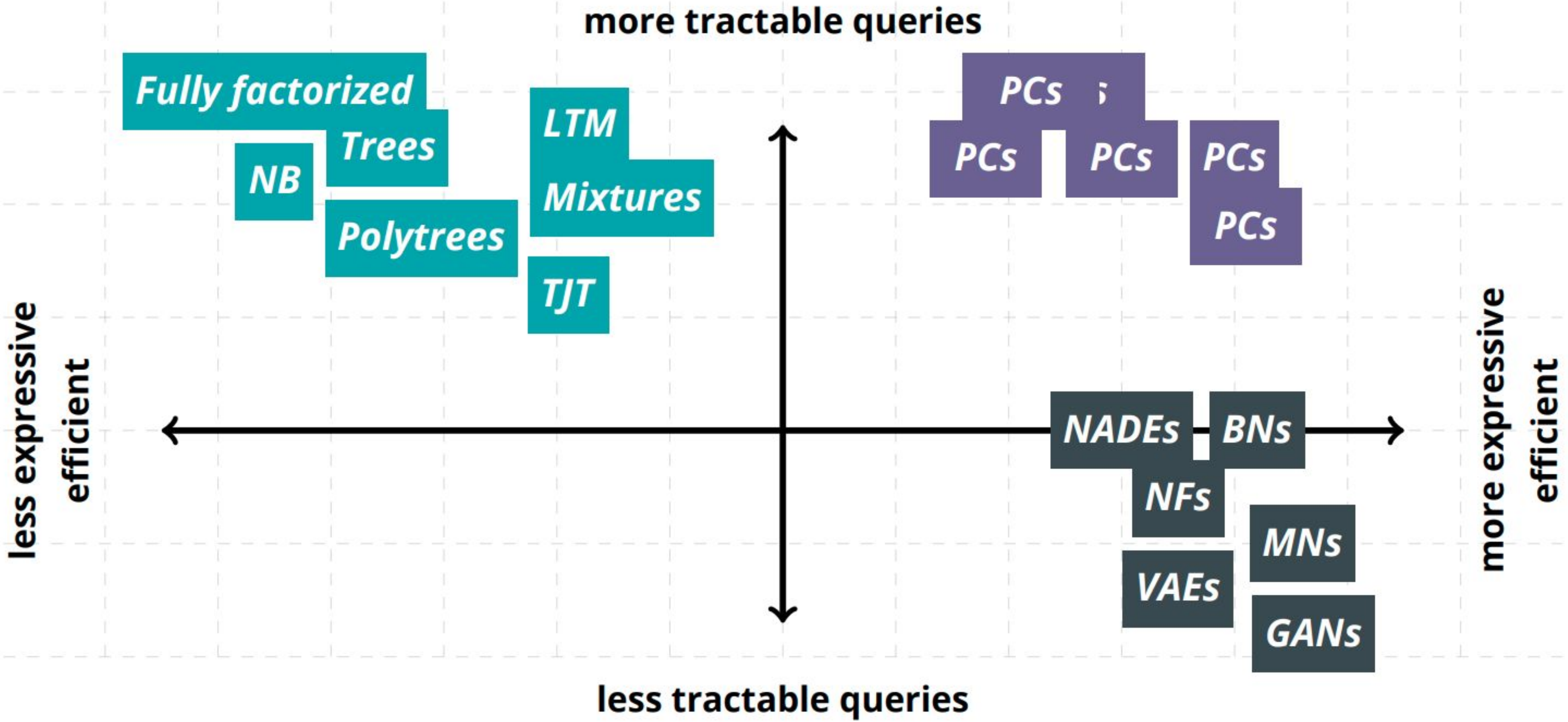
NFs

MNs

VAEs

GANs

less tractable queries



more tractable queries

Fully factorized

Trees

NB

Polytrees

LTM

Mixtures

TJT

PCs

PCs

PCs

PCs

PCs

less expressive efficient

more expressive efficient

NADEs

BNs

NFs

MNs

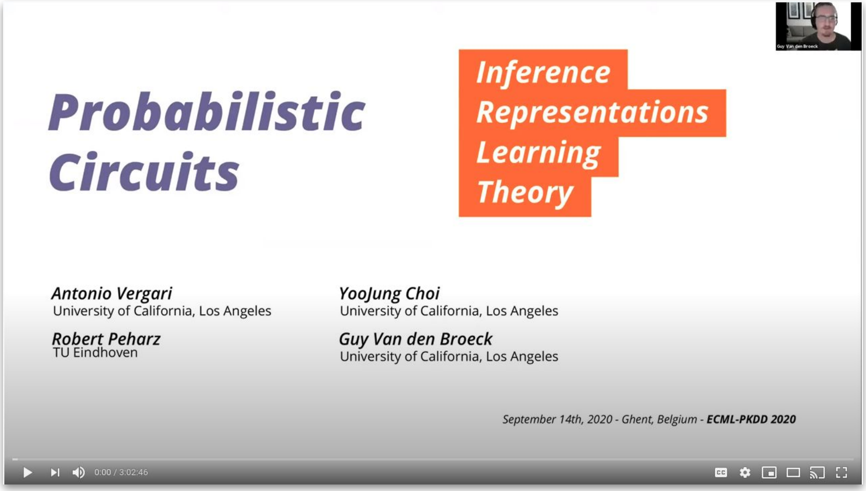
VAEs

GANs

less tractable queries

# *Learn more about probabilistic circuits?*

## Tutorial (3h)



**Probabilistic Circuits**

*Inference*
*Representations*
*Learning*
*Theory*

*Antonio Vergari*
University of California, Los Angeles

*YooJung Choi*
University of California, Los Angeles

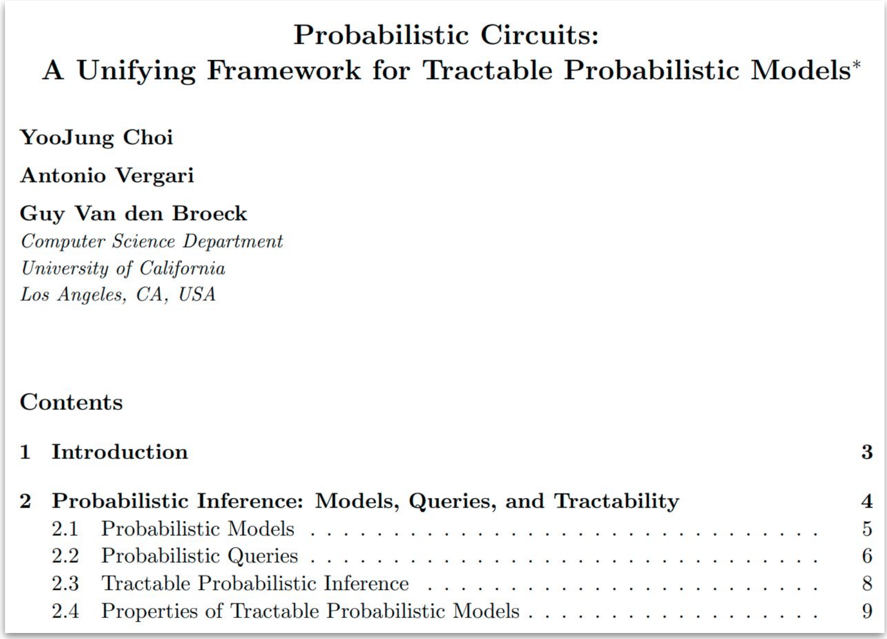
*Robert Peharz*
TU Eindhoven

*Guy Van den Broeck*
University of California, Los Angeles

*September 14th, 2020 - Ghent, Belgium - **ECML-PKDD 2020***

https://youtu.be/2RAG5-L9R70

## Overview Paper (80p)



Probabilistic Circuits:
A Unifying Framework for Tractable Probabilistic Models*

**YooJung Choi**
**Antonio Vergari**
**Guy Van den Broeck**
*Computer Science Department*
*University of California*
*Los Angeles, CA, USA*

**Contents**

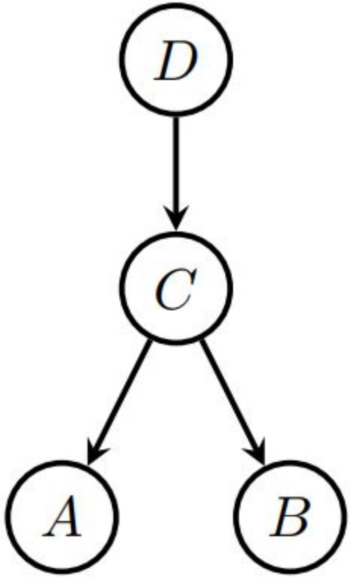
**1 Introduction** **3**

**2 Probabilistic Inference: Models, Queries, and Tractability** **4**
2.1 Probabilistic Models . . . . . . . . . . 5
2.2 Probabilistic Queries . . . . . . . . . . 6
2.3 Tractable Probabilistic Inference . . . . . . . . . . 8
2.4 Properties of Tractable Probabilistic Models . . . . . . . . . . 9

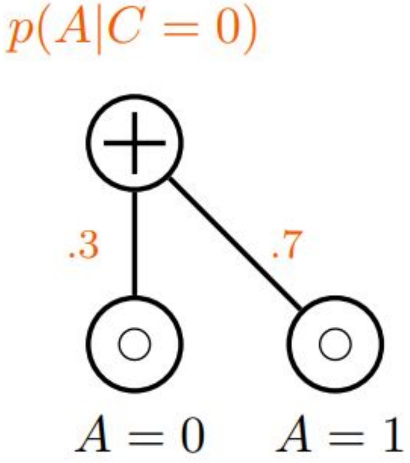
http://starai.cs.ucla.edu/papers/ProbCirc20.pdf

# From BN trees to circuits

*via compilation*

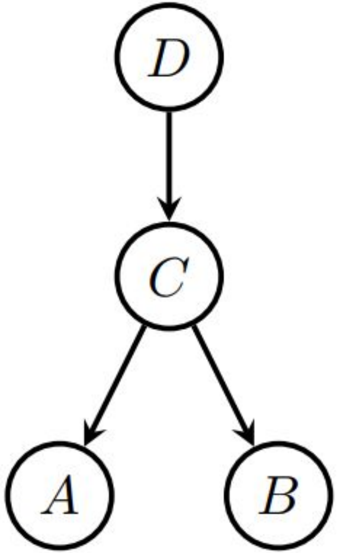


...compile a leaf CPT

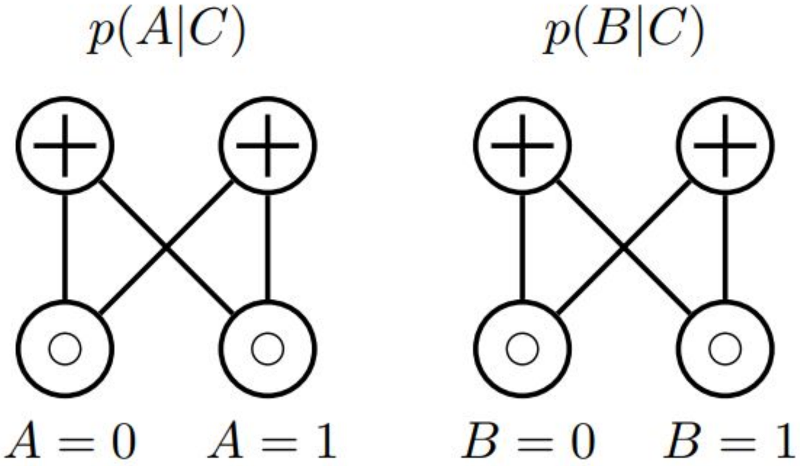


# From BN trees to circuits

*via compilation*

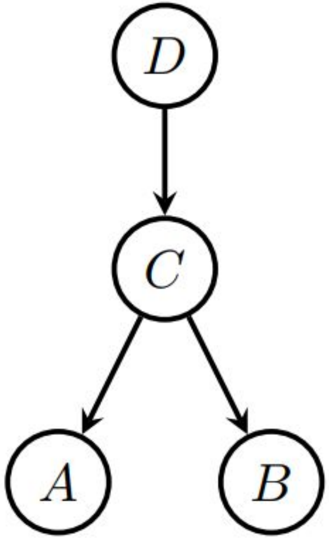


...compile a leaf CPT...for all leaves...

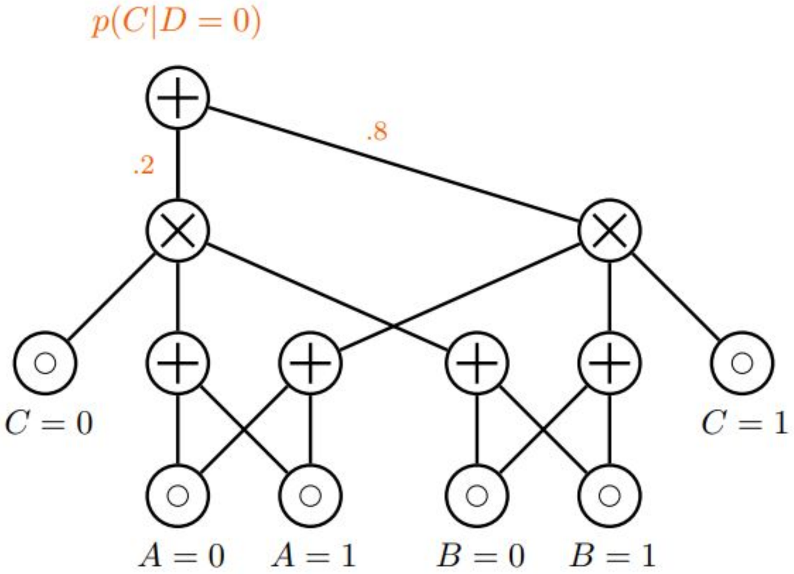


# From BN trees to circuits

*via compilation*

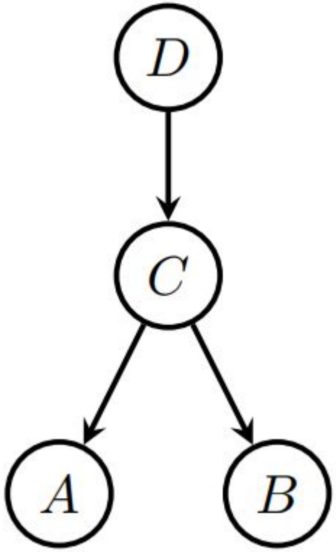


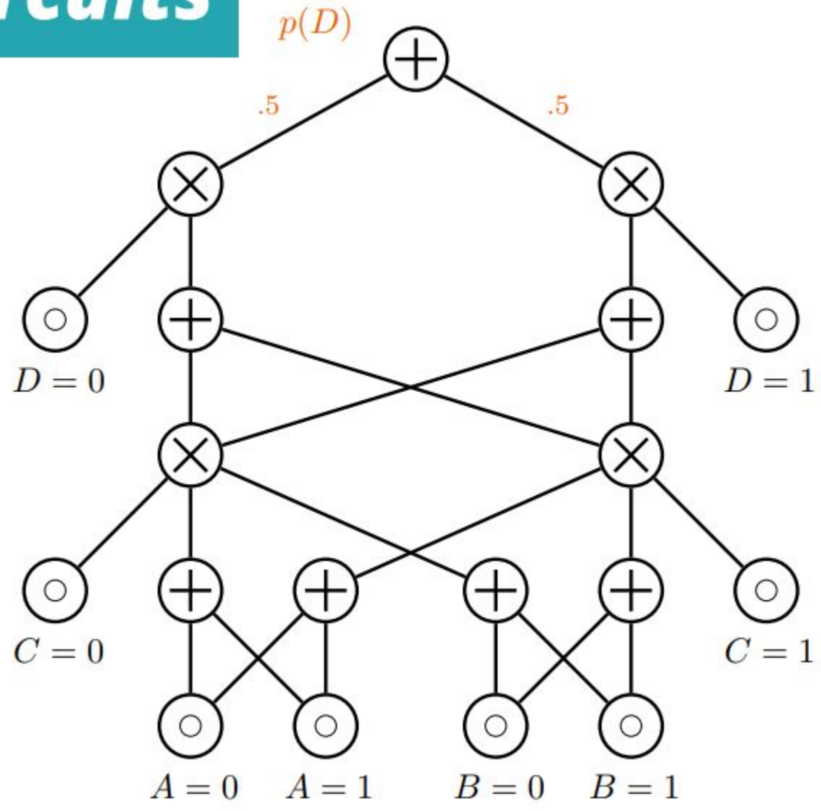
...and recurse over parents...



# From BN trees to circuits

*via compilation*





$p(D)$
.5
.5
$D = 0$
$D = 1$
$C = 0$
$C = 1$
$A = 0$
$A = 1$
$B = 0$
$B = 1$