Deep Tractable Probabilistic Models




SAHIL SIDHEEKH SAURABH MATHUR ATHRESH KARANAM SRIRAAM NATARAJAN

THE UNIVERSITY OF TEXAS AT DALLAS

Generative models can synthesize data

Karras, Tero, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks." ICCV (2019).

See also: https://thispersondoesnotexist.com

S

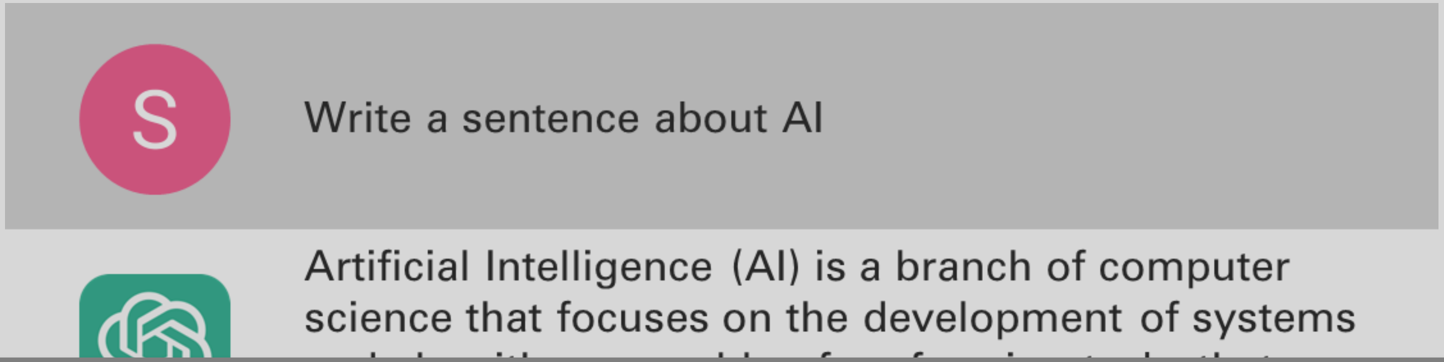
Write a sentence about AI

Artificial Intelligence (AI) is a branch of computer science that focuses on the development of systems and algorithms capable of performing tasks that typically require human intelligence, such as learning, problem-solving, and pattern recognition.

Generative models can synthesize data

GPT 3.5 by OpenAI

See also: chat.openai.com



But can they reason?

Generative models can synthesize data

GPT 3.5 by OpenAI

See also: chat.openai.com

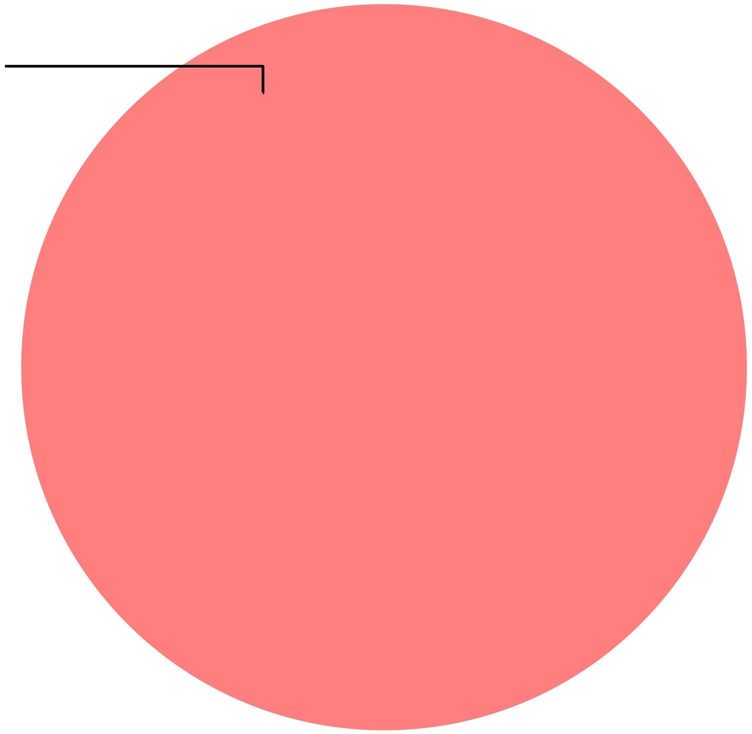
“[…] the model attaining high accuracy only on in-distribution test examples has not learned to reason. In fact, the model has learned to use statistical features in logical reasoning problems to make predictions rather than to emulate the correct reasoning function.”

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck "On the Paradox of Learning to Reason from Data. " IJCAI (2023).

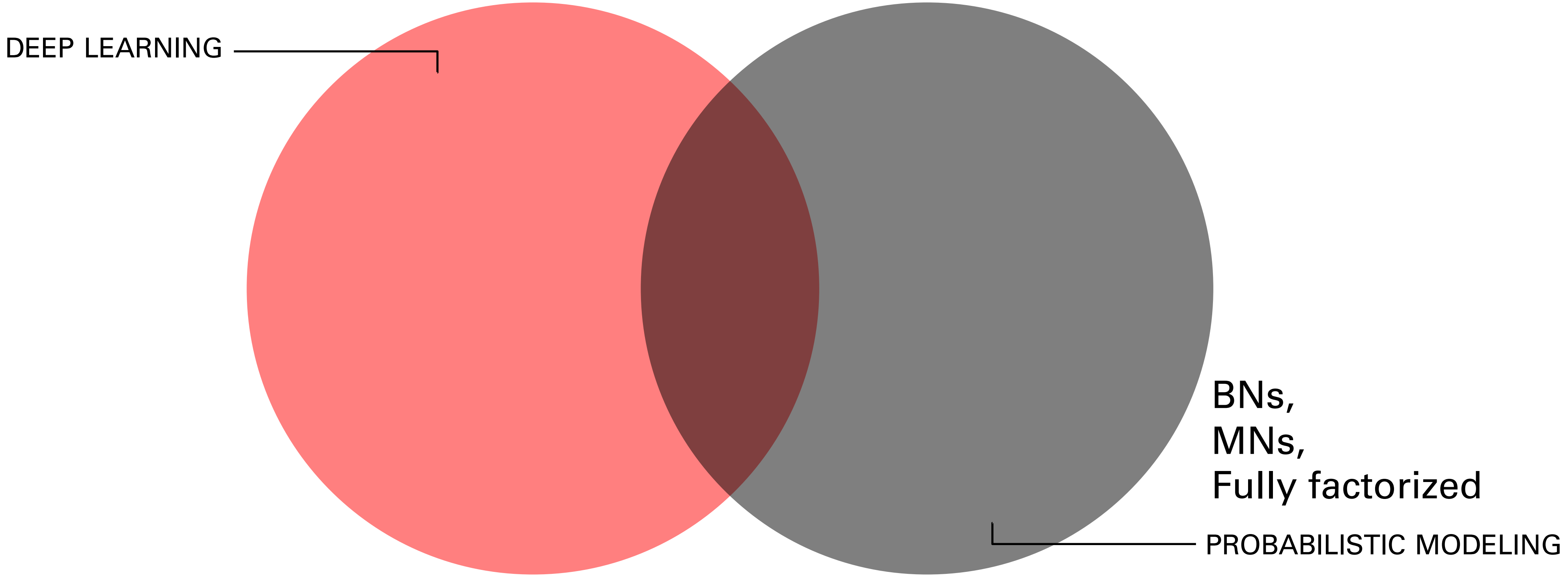
Deep Tractable Probabilistic Models

DEEP LEARNING

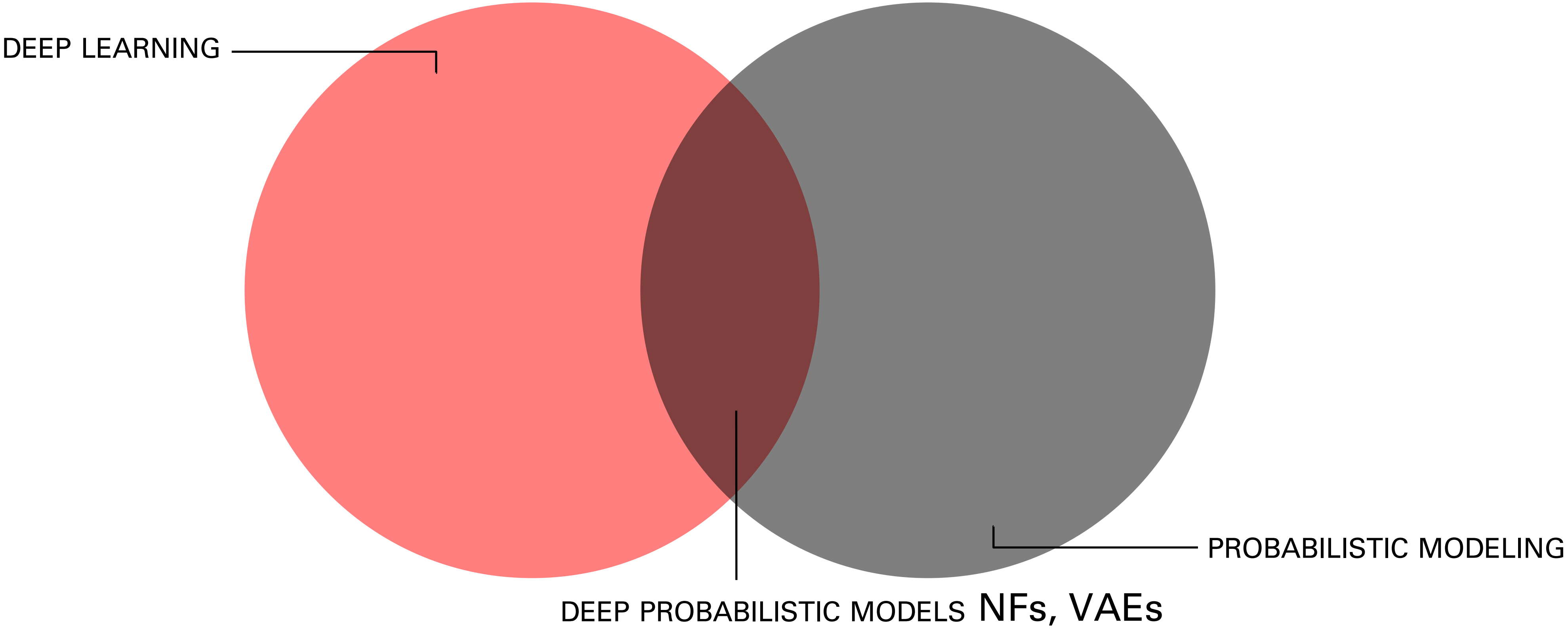
LLMs,
GANs,
VAEs



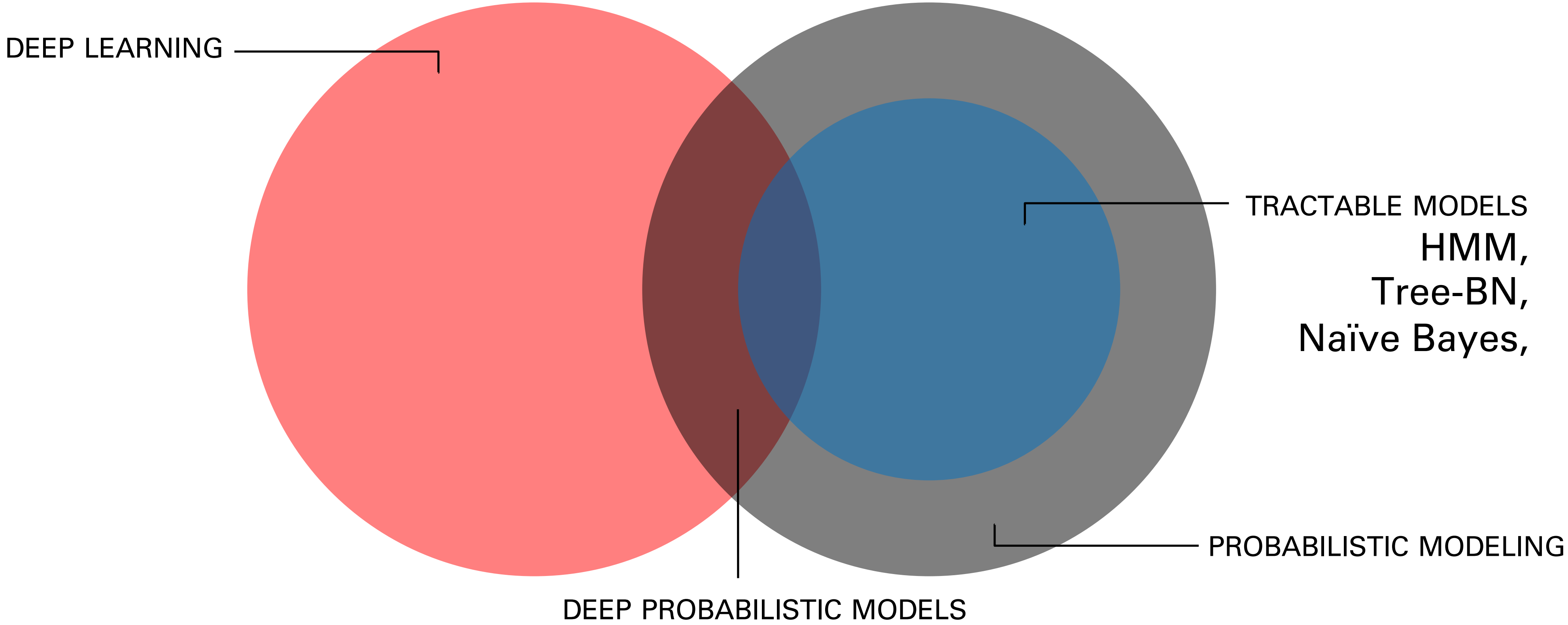
Deep Tractable Probabilistic Models



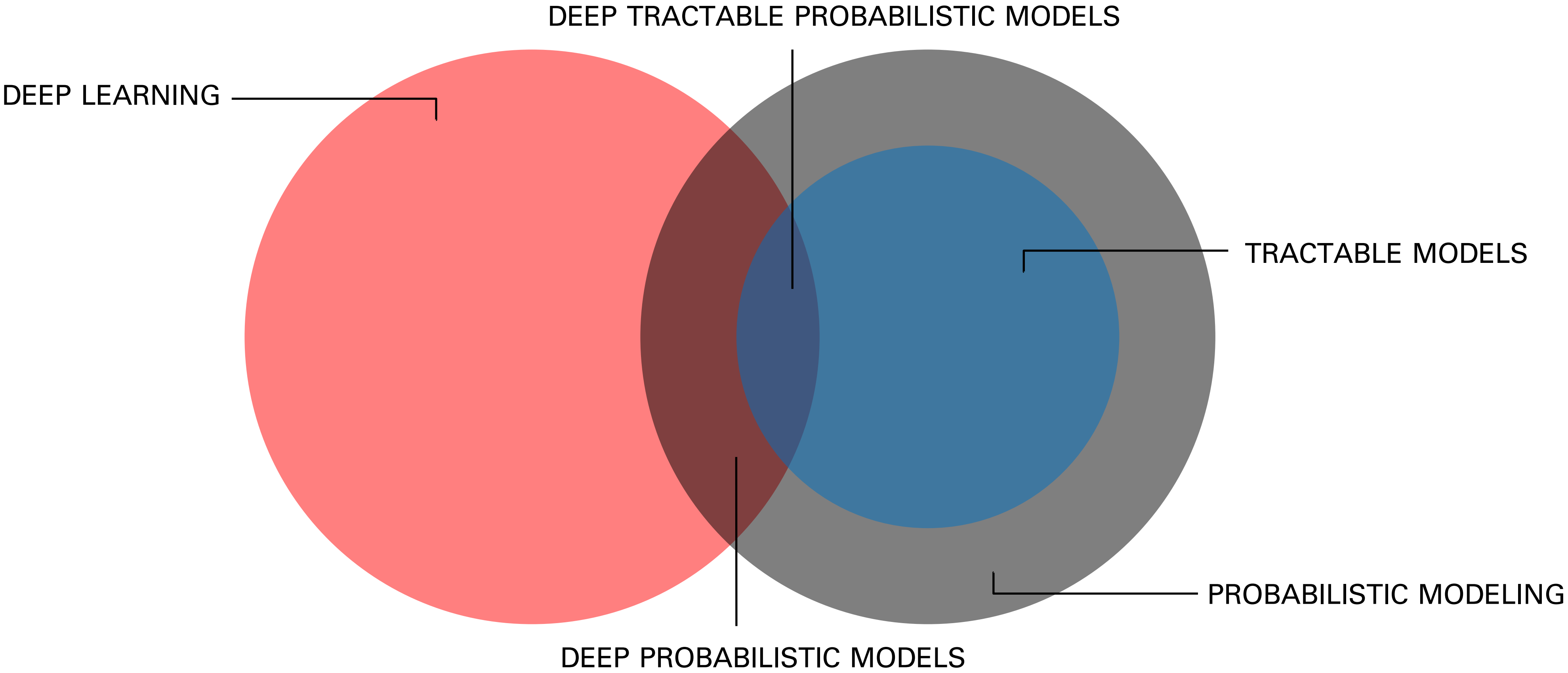
Deep Tractable Probabilistic Models



Deep Tractable Probabilistic Models



Deep Tractable Probabilistic Models



Key Question

How can we build generative models that can:

1. Fit complex data distributions, and
2. Efficiently reason under uncertainty

Why Probabilistic models?

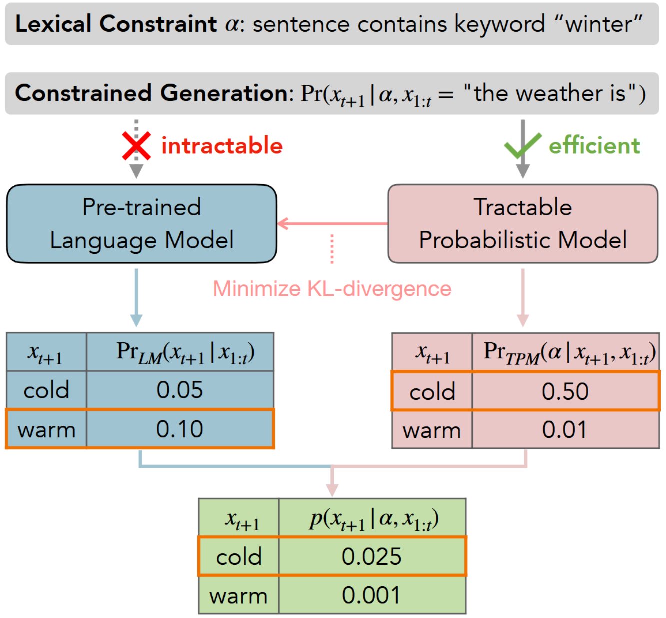
- **Reasoning under uncertainty.** Dealing with missing data
- **Robustness.** Models know what they don't know
- **Flexibility.** Interoperability via language of probability

Why Tractable Probabilistic Models?

- **Consistency.** Approximations might be internally inconsistent
- **Efficiency.** Transform CPU computation to GPU-accelerated computation
- **Explainability & Interpretability.** Can explain decisions using queries

DTPMs in recent literature

Constrained-text generation



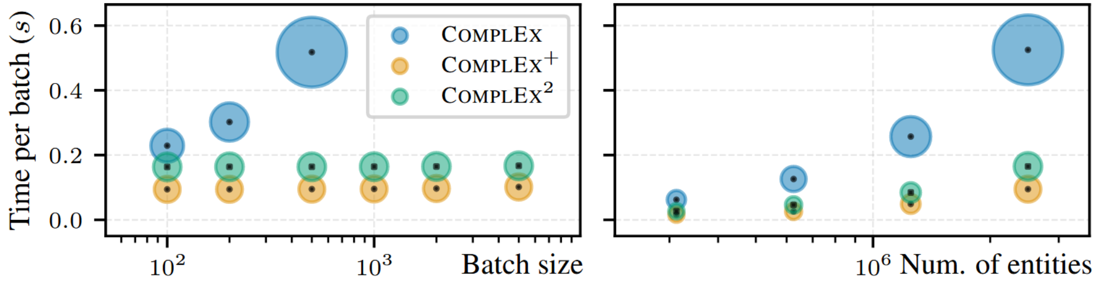
Zhang, Honghua, Meihua Dang, Nanyun Peng, and Guy Van den Broeck. "Tractable control for autoregressive language generation. " ICML (2023).

Structured-output prediction

Ahmed, Kareem, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, and Antonio Vergari. "Semantic probabilistic layers for neuro-symbolic learning." NeurIPS (2022).

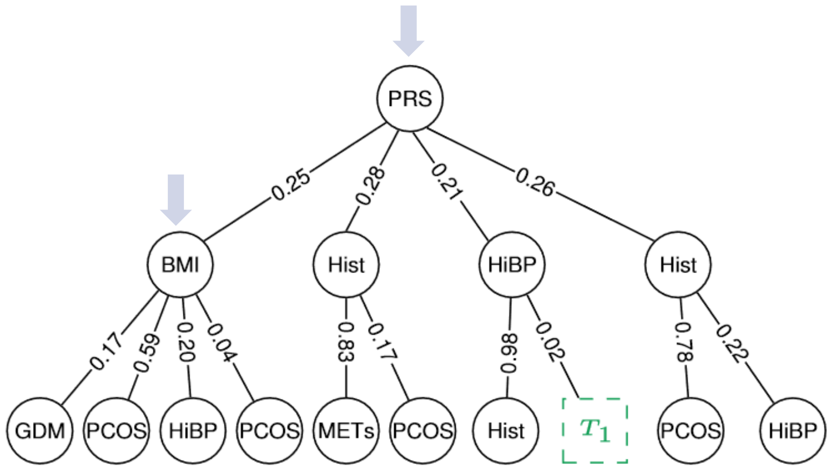
DTPMs in recent literature

Scaling Knowledge-graph embeddings



Loconte, Lorenzo, Nicola Di Mauro, Robert Peharz, and Antonio Vergari. "How to Turn Your Knowledge Graph Embeddings into Generative Models." NeurIPS (2023).

Knowledge-intensive learning



Mathur, Saurabh, Vibhav Gogate, and Sriraam Natarajan. "Knowledge intensive learning of cutset networks.“ UAI (2023).

Key Idea

DTPMs can answer complex queries over high dimensional data exactly by leveraging advances in deep learning.

Tutorial Outline

1. **Tractable Probabilistic Inference**
2. **Probabilistic Circuits (PCs)**

Demo 1

Short Q/A Break

3. **Deep parameterizations of PCs**
4. **Use cases of PCs**
5. **Summary**

Demo 2

Q/A

Tutorial Outline

1. **Tractable Probabilistic Inference**
2. **Probabilistic Circuits (PCs)**

Demo 1

Short Q/A Break

3. **Deep parameterizations of PCs**
4. **Use cases of PCs**
5. **Summary**

Demo 2

Q/A

Adapted from some amazing tutorials on probabilistic circuits:

- Probabilistic Circuits: Representations, Inference, Learning and Theory. Antonio Vergari, YooJung Choi, Robert Peharz, Guy Van den Broeck (IJCAI-PRICAI 2021, ECML-PKDD 2020, ECAI 2020)
- Probabilistic Circuits: Representations, Inference, Learning and Applications. Antonio Vergari, YooJung Choi, Robert Peharz, Guy Van den Broeck (AAAI 2020)
- Tractable Probabilistic Models. Antonio Vergari, Nicola Di Mauro, Guy Van den Broeck (ICLP 2019, UAI 2019)

Tractable Probabilistic Inference

DTPMs

Tractable Probabilistic Inference

1. **Probabilistic Inference & types of queries**
2. What is Tractable Probabilistic Inference?
3. Are popular generative models tractable?

Motivating Example
Mitigating the risk of APOs

15% of live births in India are Pre-term.

Shinjini Bhatnagar et al. A Pregnancy Cohort to Study Multidimensional Correlates of Preterm Birth in India: Study Design, Implementation, and Baseline Characteristics of the Participants, *American Journal of Epidemiology (2019).*

Motivating Example
Mitigating the risk of APOs

15% of live births in India are Pre-term.

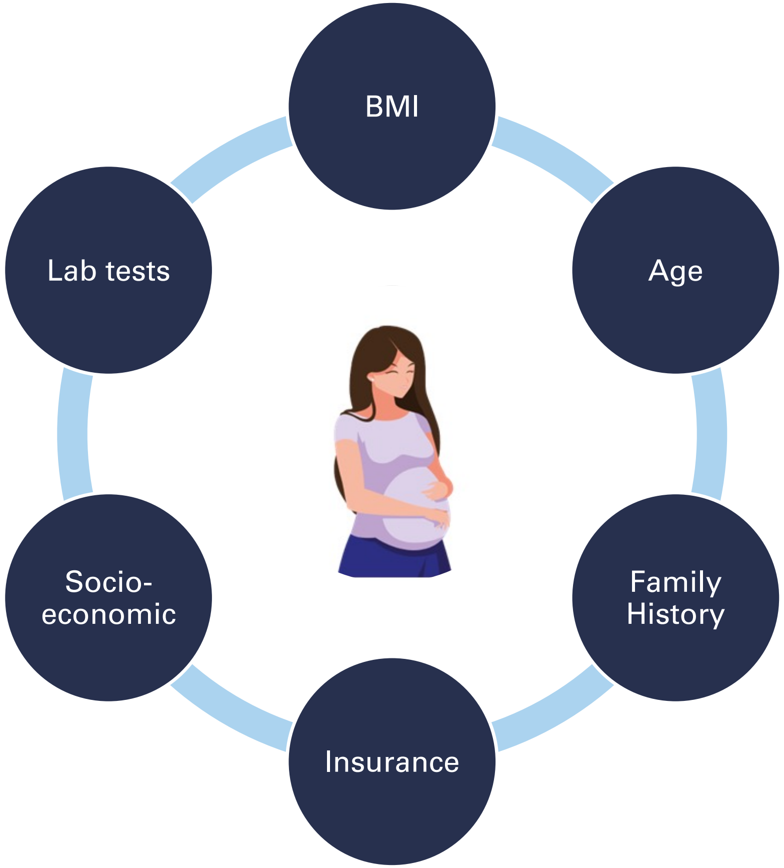
~20% pregnancies in US are affected by at least 1 Adverse Pregnancy Outcome

- Preeclampsia
- Gestational diabetes (GDM)
- Pre-term birth
- Fetal growth restriction
- Fetal demise

Shinjini Bhatnagar et al. A Pregnancy Cohort to Study Multidimensional Correlates of Preterm Birth in India: Study Design, Implementation, and Baseline Characteristics of the Participants, *American Journal of Epidemiology (2019).*

David M Haas et al. "A description of the methods of the nulliparous pregnancy outcomes study: monitoring mothers-to-be (numom2b). American journal of obstetrics and gynecology, 2015.

Motivating Example
Mitigating the risk of APOs

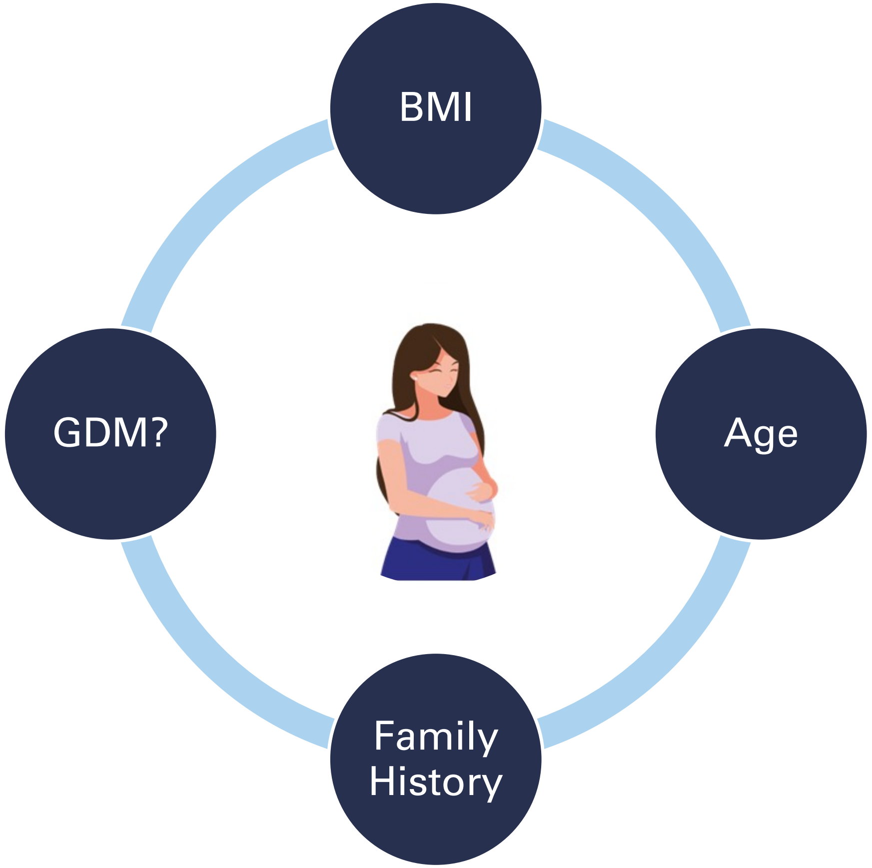


~20% pregnancies in US are affected by at least 1 Adverse Pregnancy Outcome

- Preeclampsia
- Gestational diabetes (GDM)
- Pre-term birth
- Fetal growth restriction
- Fetal demise

David M Haas et al. "A description of the methods of the nulliparous pregnancy outcomes study: monitoring mothers-to-be (numom2b). American journal of obstetrics and gynecology, 2015.

Set up



Consider a probabilistic model M,

Representing distribution over 4 Boolean random vars.

$\boldsymbol{P}_M(\text{GDM}, \text{Age}, \text{BMI}, \text{Hist})$ is a Probability Mass Function

$\boldsymbol{P}_M(\mathrm{GDM}, \mathrm{Age}, \mathrm{BMI}, \mathrm{Hist})$ is a PMF

- Each probability $0 \leq p_i \leq 1$
- All probabilities total up to 1

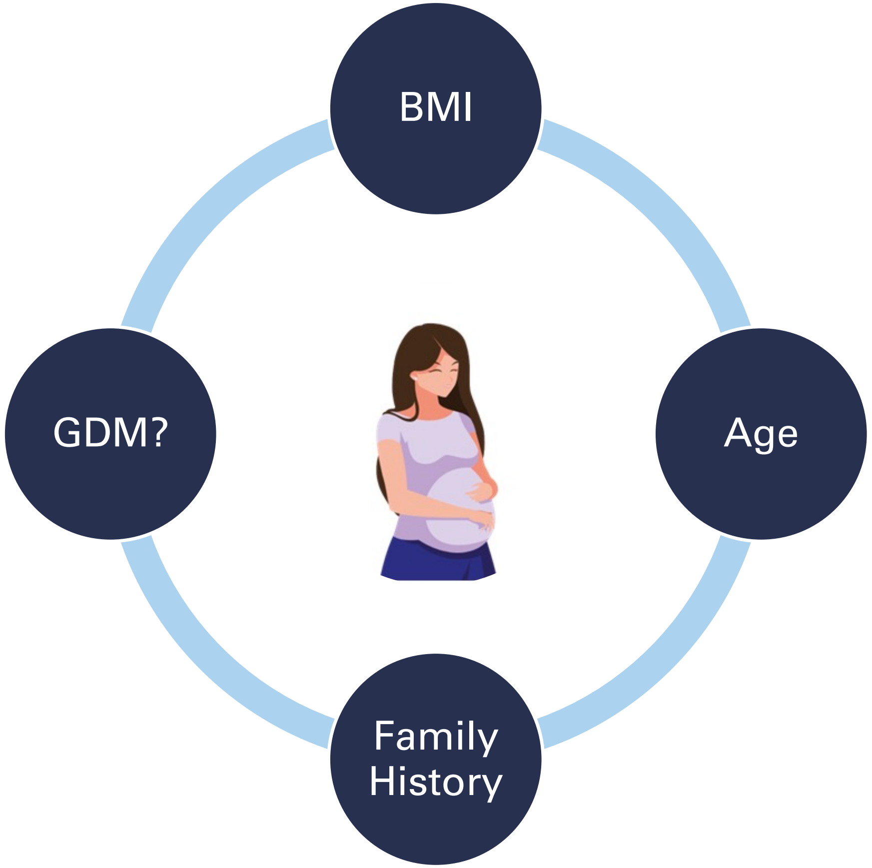
$$\sum_{\boldsymbol{x} \epsilon \{0,1\}} \boldsymbol{P}_M(\mathrm{GDM} = x_1, \mathrm{Age} = x_2, \mathrm{BMI} = x_3, \mathrm{Hist} = x_4) = 1$$

PROBABILITY

ASSIGNMENT

GDM	Age	BMI	Hist	$P_M(\mathrm{X})$
0	0	0	0	p_1
0	0	0	1	p_2
		⋮		⋮
1	1	1	1	p_{16}

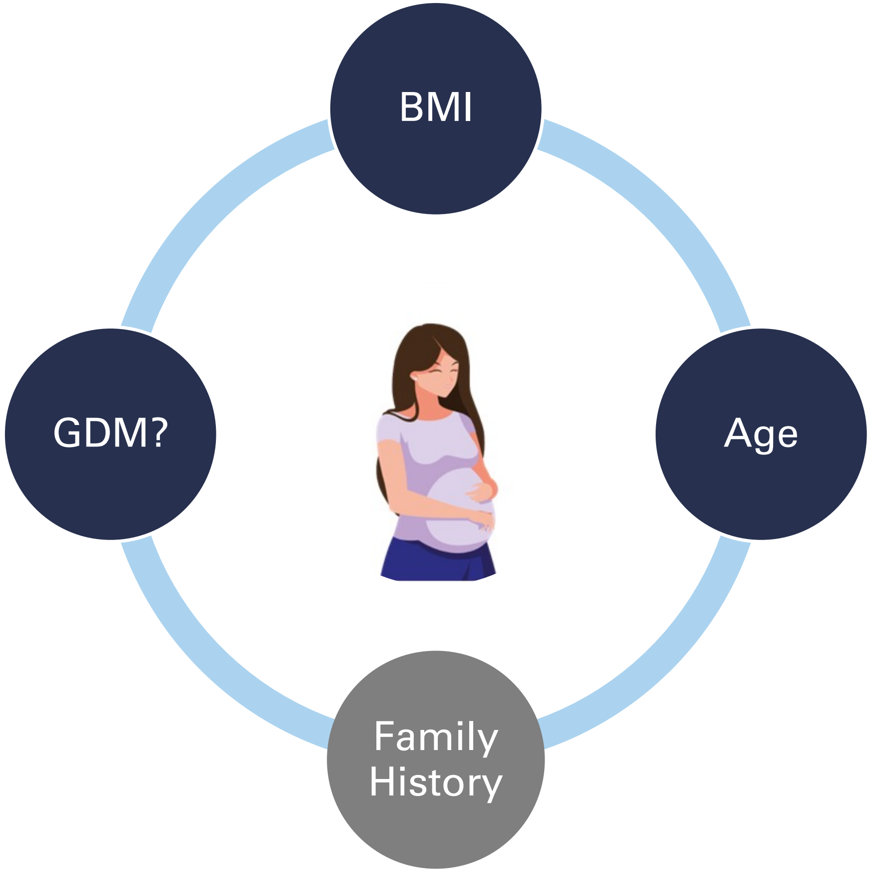
Types of Inference Queries
1. Evidential Inference (EVI)



Q1: What is the likelihood of a pregnant woman being over the age of 35, having high BMI, a family history of diabetes and gestational diabetes?

$$\boldsymbol{P}_M(\mathrm{GDM}=1, \mathrm{Age}=1, \mathrm{BMI}=1, \mathrm{Hist}=1)$$

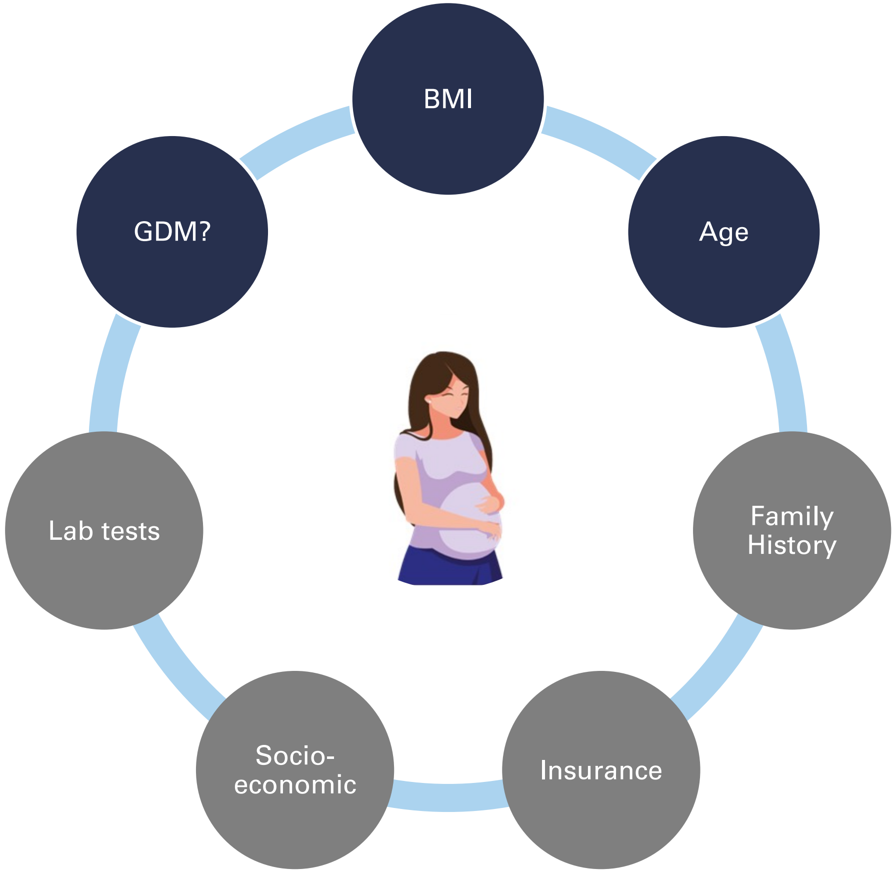
Types of Inference Queries
2. Marginal Inference (MAR)



Q2: What is the likelihood of a pregnant woman being over the age of 35, having high BMI, ~~a family history of diabetes and~~ gestational diabetes?

$$\sum_{x' \epsilon \{0,1\}} \boldsymbol{P}_M(\text{GDM} = 1, \text{Age} = 1, \text{BMI} = 1, \text{Hist} = x')$$

Types of Inference Queries
2. Marginal Inference (MAR)

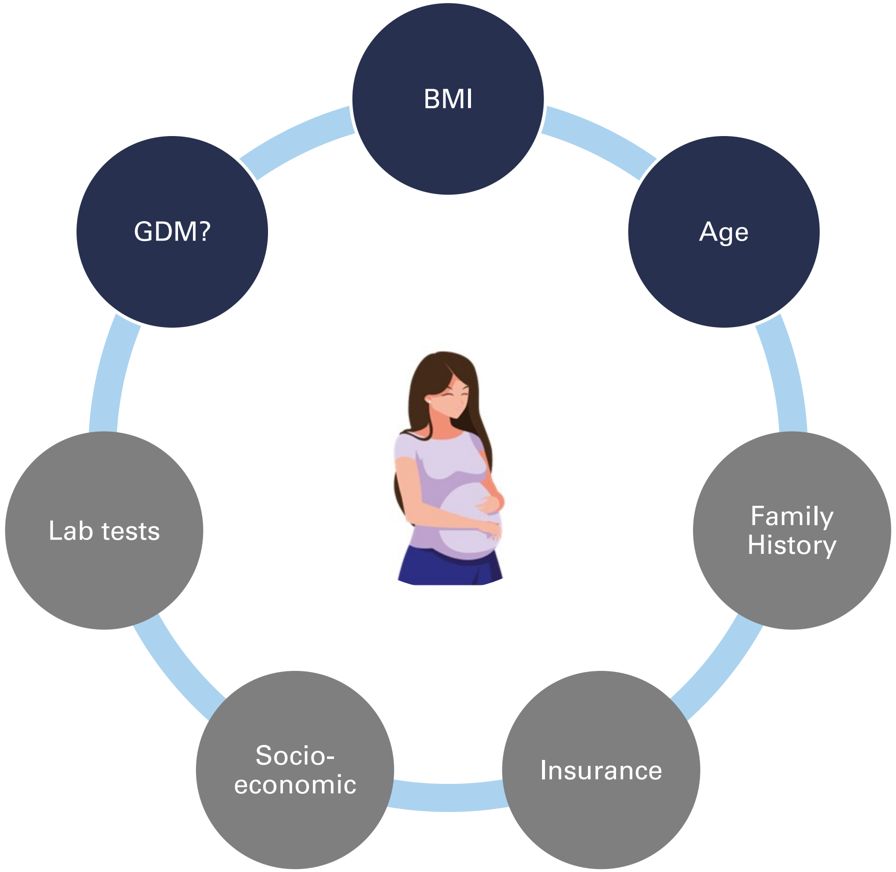


Q2: What is the likelihood of a pregnant woman being over the age of 35, having high BMI, ~~a family history of diabetes and~~ gestational diabetes?

$$\sum_{\boldsymbol{x'}} \boldsymbol{P}_M(\text{GDM} = 1, \text{Age} = 1, \text{BMI} = 1, \boldsymbol{X'} = \boldsymbol{x'})$$

All values of every other variable!

Types of Inference Queries
2. Marginal Inference (MAR)

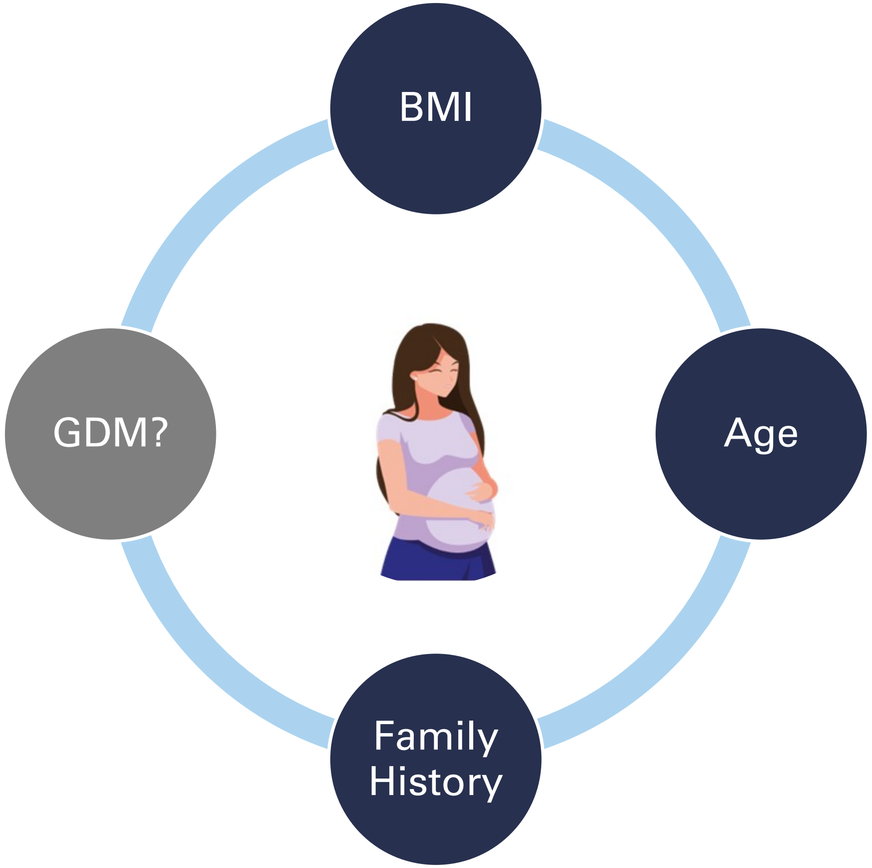


Q2: What is the likelihood of a pregnant woman being over the age of 35, having high BMI, ~~a family history of diabetes and~~ gestational diabetes?

$$\sum_{\boldsymbol{x}'} \boldsymbol{P}_M(\mathrm{GDM} = 1, \mathrm{Age} = 1, \mathrm{BMI} = 1, \boldsymbol{X}' = \boldsymbol{x}')$$

All values of every other variable!

Types of Inference Queries
3. Conditional Inference (CON)

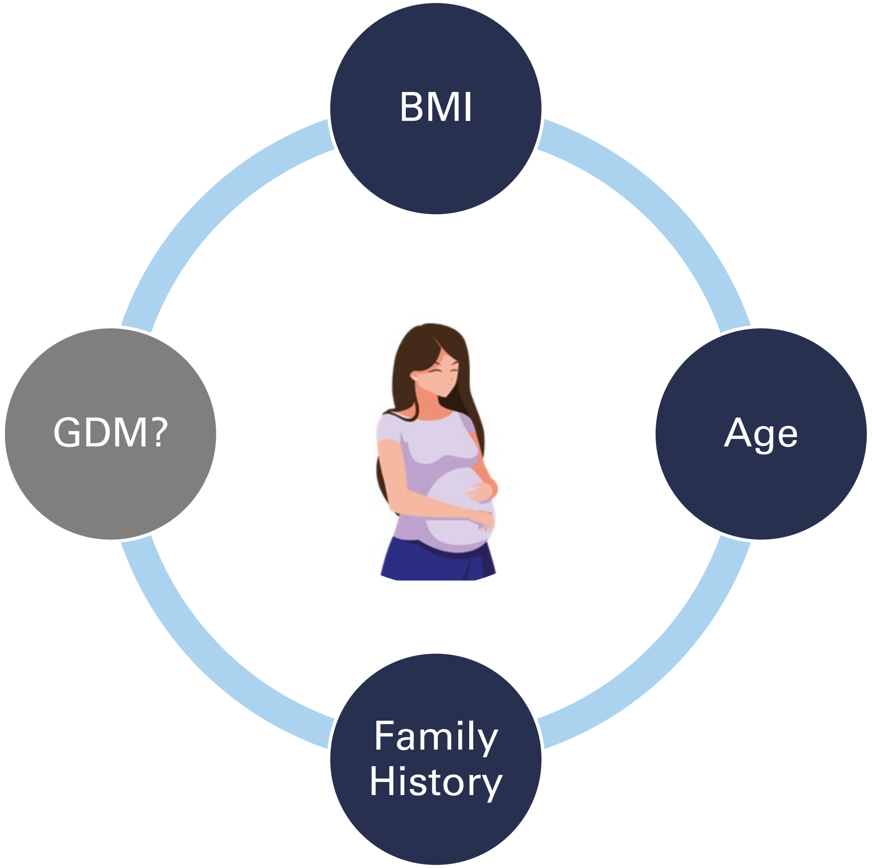


Q3: What is the likelihood of a pregnant woman having gestational diabetes given that she is over the age of 35, has high BMI, and a family history of diabetes?

$$\boldsymbol{P}_M(\mathrm{GDM} = 1 \mid \mathrm{Age} = 1, \mathrm{BMI} = 1, \mathrm{Hist} = 1) =$$

$$\frac{\boldsymbol{P}_M(\mathrm{GDM} = 1, \mathrm{Age} = 1, \mathrm{BMI} = 1, \mathrm{Hist} = 1)}{\sum_{x \epsilon \{0,1\}} \boldsymbol{P}_M(\mathrm{GDM} = x, \mathrm{Age} = 1, \mathrm{BMI} = 1, \mathrm{Hist} = 1)}$$

Types of Inference Queries
4. Maximum a posteriori Inference (MAP)

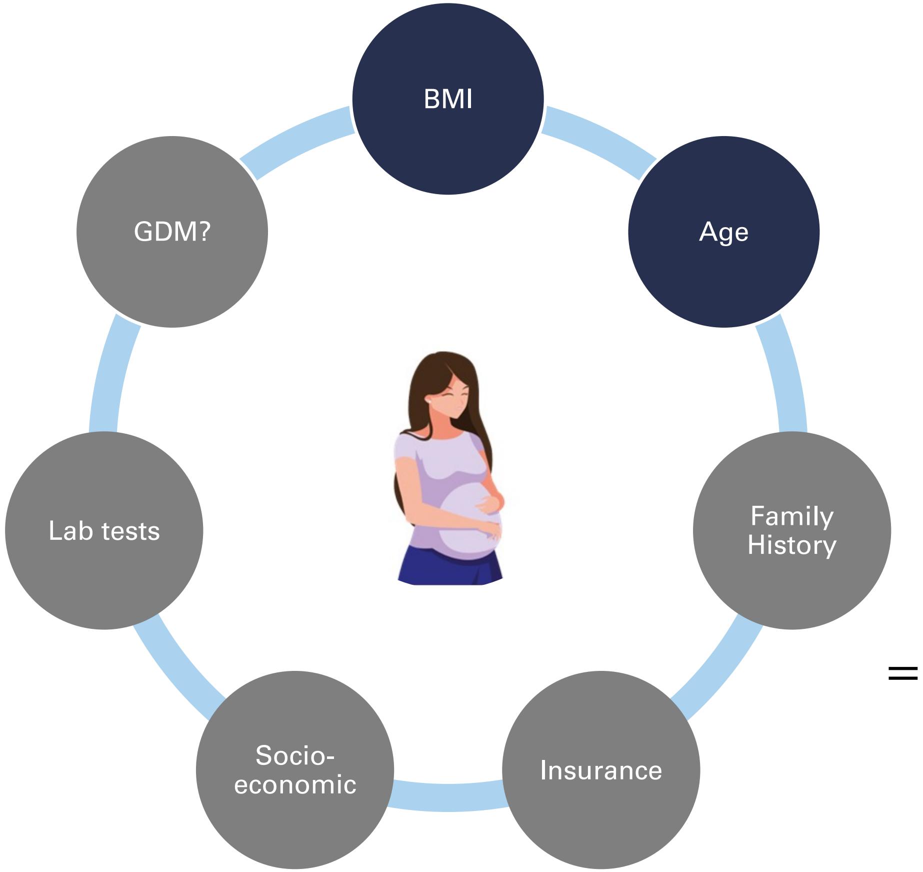


Q4: A pregnant woman is over the age of 35, has high BMI, and a family history of diabetes. Is she likely to develop Gestational Diabetes?

$$\arg\max_{x\epsilon\{0,1\}} \boldsymbol{P}_M(\mathrm{GDM} = x \mid \mathrm{Age} = 1, \mathrm{BMI} = 1, \mathrm{Hist} = 1)$$

$$= \arg\max_{x\epsilon\{0,1\}} \boldsymbol{P}_M(\mathrm{GDM} = x, \mathrm{Age} = 1, \mathrm{BMI} = 1, \mathrm{Hist} = 1)$$

Types of Inference Queries
5. Marginal MAP (MMAP)



Q5: A pregnant woman is over the age of 35, has high BMI, and a family history of diabetes. Is she likely to develop Gestational Diabetes?

$$\underset{x \epsilon \{0,1\}}{\arg\max}\ \boldsymbol{P}_M(\mathrm{GDM}=x, \mathrm{Age}=1, \mathrm{BMI}=1, \mathrm{Hist}=1)$$

$$= \underset{x \epsilon \{0,1\}}{\arg\max}\ \sum_{\boldsymbol{x}'} \boldsymbol{P}_M(\mathrm{GDM}=x, \mathrm{Age}=1, \mathrm{BMI}=1, \mathrm{Hist}=1, \boldsymbol{X}'=\boldsymbol{x}')$$

All values of every other variable!

Types of Inference Queries

Question	Query	Expression
How likely is a data point?	Full evidence (EVI)	$\boldsymbol{P}_M(\boldsymbol{X} = x)$
How likely is this partial data point?	Marginal (MAR)	$\boldsymbol{P}_M(\boldsymbol{X}_E = x_E)$
What is the most likely assignment given all remaining values?	Maximum a posteriori (MAP)	$\arg\max_{x_{-E}} \boldsymbol{P}_M(\boldsymbol{X}_{-E} = x_{-E} \mid \boldsymbol{X}_E = x_E)$
What is the most likely assignment given some values?	Marginal MAP (MMAP)	$\arg\max_{x_Q} \boldsymbol{P}_M(\boldsymbol{X}_Q = x_Q \mid \boldsymbol{X}_E = x_E)$

DTPMs

Tractability

1. Probabilistic Inference & types of queries
2. **What is Tractable Probabilistic Inference?**
3. Are popular generative models tractable?

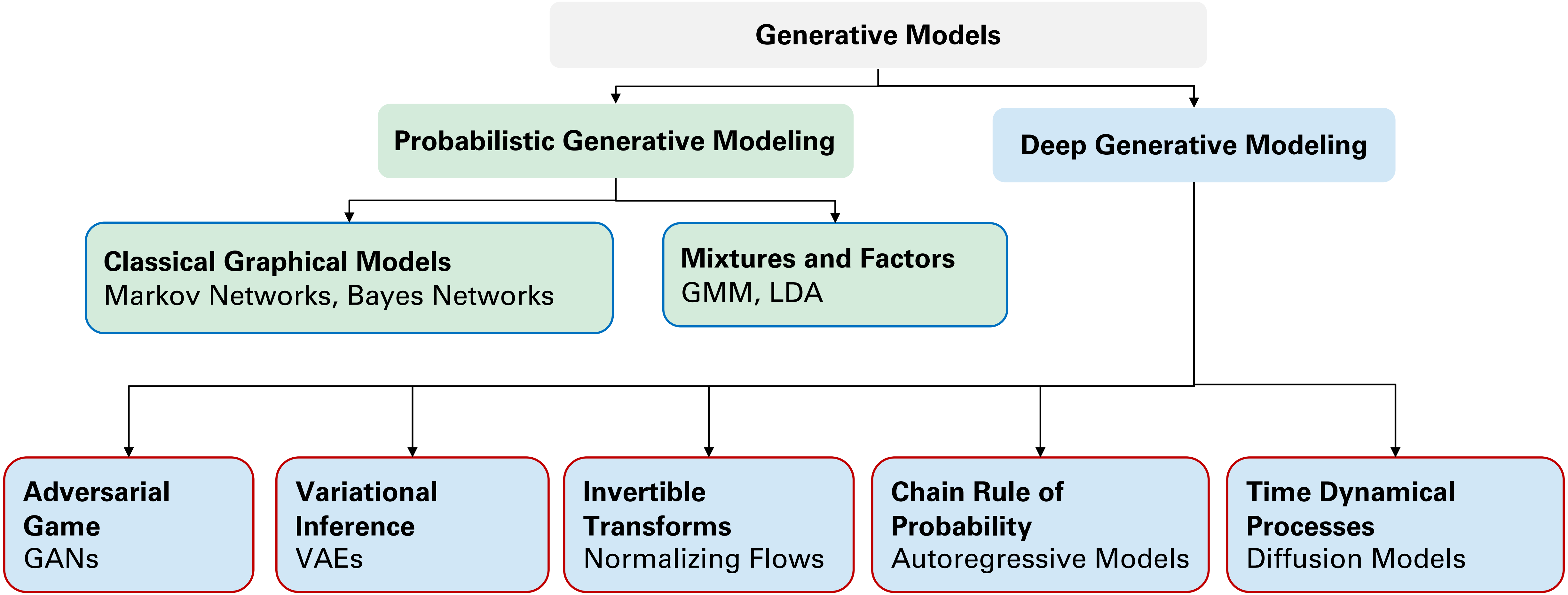
What is Tractable Probabilistic Inference?

A class of queries Q is tractable on a family of probabilistic models M *if and only if* for any query $q \in Q$ and model $m \in M$, $q(m)$ is exactly computable in $O(poly|m|)$ time.

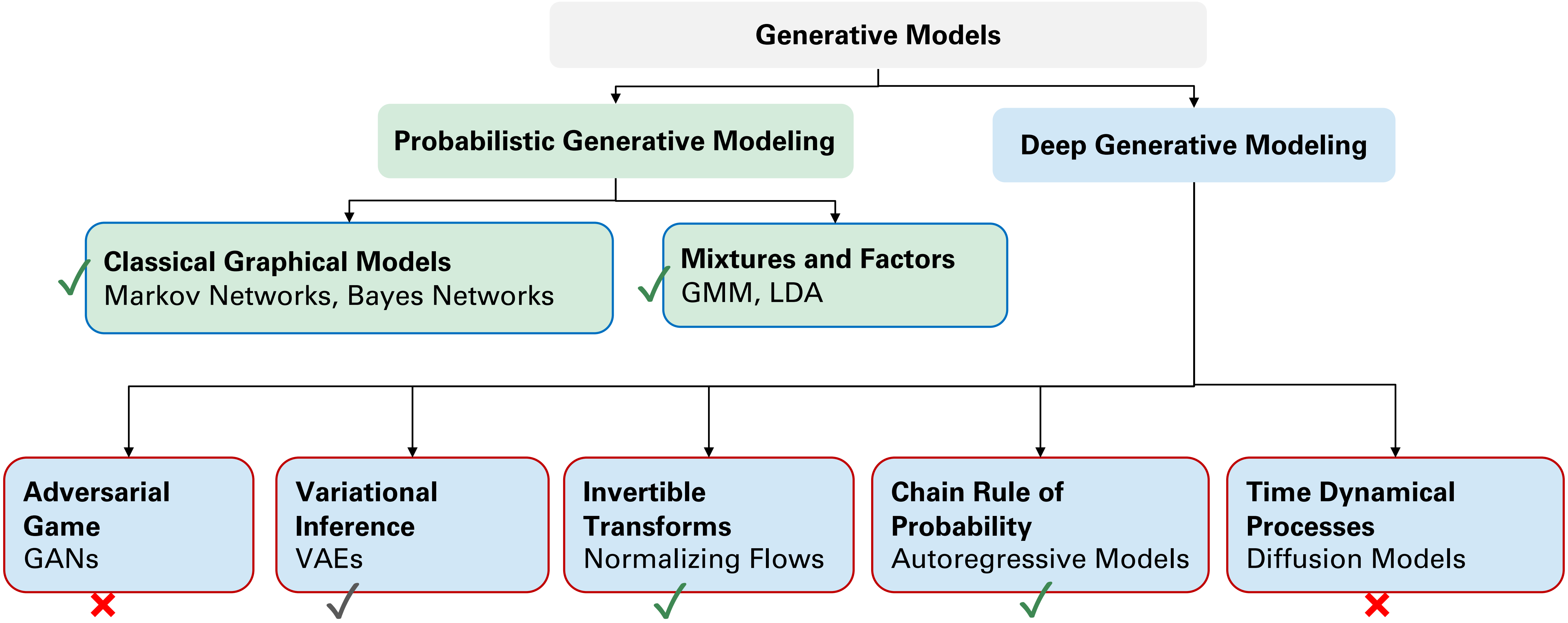
DTPMs

Tractability

1. Probabilistic Inference & types of queries
2. What is Tractable Probabilistic Inference?
3. **Are popular generative models tractable?**

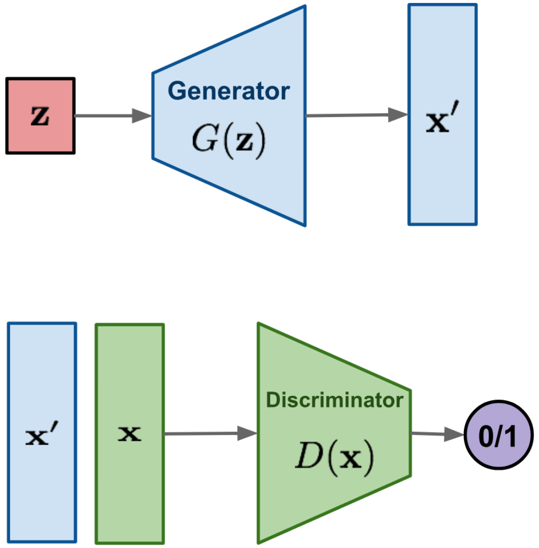

Generative Models
Probabilistic Generative Modeling
Deep Generative Modeling
Classical Graphical Models
Markov Networks, Bayes Networks
Mixtures and Factors
GMM, LDA
Adversarial Game
GANs
Variational Inference
VAEs
Invertible Transforms
Normalizing Flows
Chain Rule of Probability
Autoregressive Models
Time Dynamical Processes
Diffusion Models

Is the Probability Mass Function explicitly defined?



Deep Generative Models
Generative Adversarial Networks

Type:	Adversarial Game-based DGM
Specific features:	1) Generator transforms noise into data 2) Discriminator detects synthetic data
Learning:	1) Backpropagation 2) Minimax loss, Wasserstein loss
Inference:	Sampling from generator
Limitation:	1) Failure to converge 2) Mode collapse



Bond-Taylor, Sam, et al. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE transactions on pattern analysis and machine intelligence* (2021).

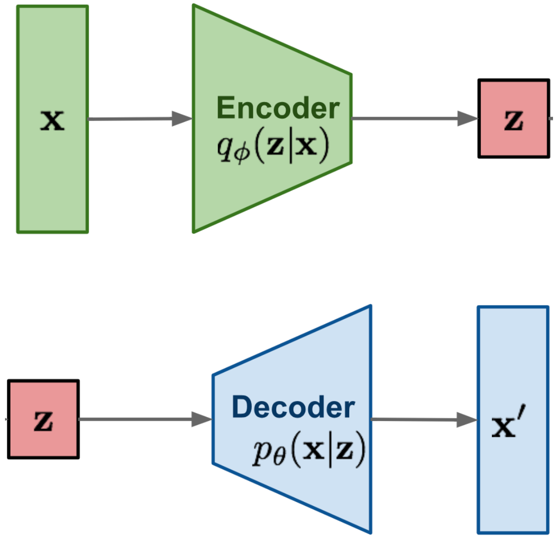
Weng, Lilian. "Flow-Based Deep Generative Models." Lil'Log, 13 Oct. 2018, lilianweng.github.io/posts/2018-10-13-flow-models/.

Deep Generative Models
Generative Adversarial Networks

	EVI	MAR	MAP	MMAP
GAN	✗	✗	✗	✗

Deep Generative Models
Variational Auto-Encoders

Type:	Approximate inference-based DGM
Specific features:	1) Encoder: data $\mapsto$ latent space 2) Decoder: latent $\mapsto$ data space
Learning:	1) Backpropagation 2) Expectation lower bound objective
Inference:	Sampling from Decoder
Limitation:	Only access to lower bound of PDF



Bond-Taylor, Sam, et al. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE transactions on pattern analysis and machine intelligence* (2021).

Weng, Lilian. "Flow-Based Deep Generative Models." Lil'Log, 13 Oct. 2018, lilianweng.github.io/posts/2018-10-13-flow-models/.

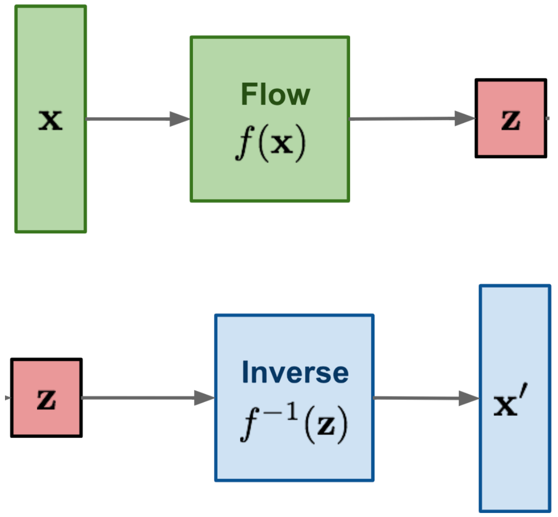
Deep Generative Models

Variational Auto-Encoders

	EVI	MAR	MAP	MMAP
GAN	✗	✗	✗	✗
VAE	✓	✗	✗	✗

Deep Generative Models
Normalizing Flows

Type:	Invertible transform-based DGM
Specific features:	1) Maps base ↦ data distribution 2) Each transform is invertible
Learning:	1) Backpropagation 2) Maximum likelihood
Inference:	1) PMF 2) Sampling from inverse flow
Limitation:	No explicit factorization of PDF



Bond-Taylor, Sam, et al. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE transactions on pattern analysis and machine intelligence* (2021).

Weng, Lilian. "Flow-Based Deep Generative Models." Lil'Log, 13 Oct. 2018, lilianweng.github.io/posts/2018-10-13-flow-models/.

Deep Generative Models
Normalizing Flows

	EVI	MAR	MAP	MMAP
GANs	✗	✗	✗	✗
VAEs	✓	✗	✗	✗
Normalizing Flows	✓	✗	✗	✗

45

Probabilistic Graphical Models
Markov Networks

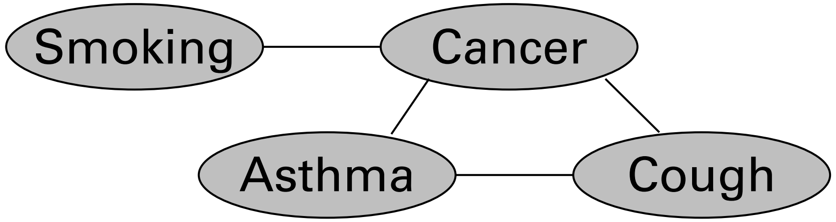
Type: Undirected PGM

Specific features:
1) Undirected graph of random vars.
2) Edge represents correlation
3) Factors defined over cliques

Learning: Maximum (pseudo-)likelihood

Inference: EVI is tractable if Z is tractable

Limitation:
1) Z is intractable in general
2) Exact inference is #P-hard in general



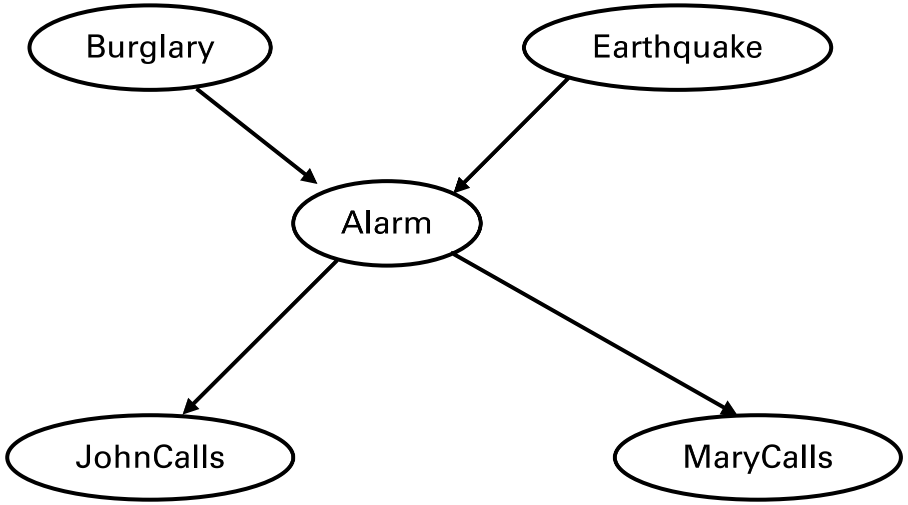
$$P(x) = \frac{1}{Z}\prod_{c}\Phi_c(x_c)$$

$$Z = \sum_{x}\prod_{c}\Phi_c(x_c)$$

Koller, Daphne, and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Probabilistic Graphical Models
Bayesian Networks

Type:	Directed PGM
Specific features:	1) DAG of random vars. 2) Directed edge represents influence 3) Local conditional distributions
Learning:	Maximum likelihood
Inference:	Only EVI is tractable. Practical exact inference via compilation & cutset conditioning
Limitation:	Exact inference is #P-hard in general

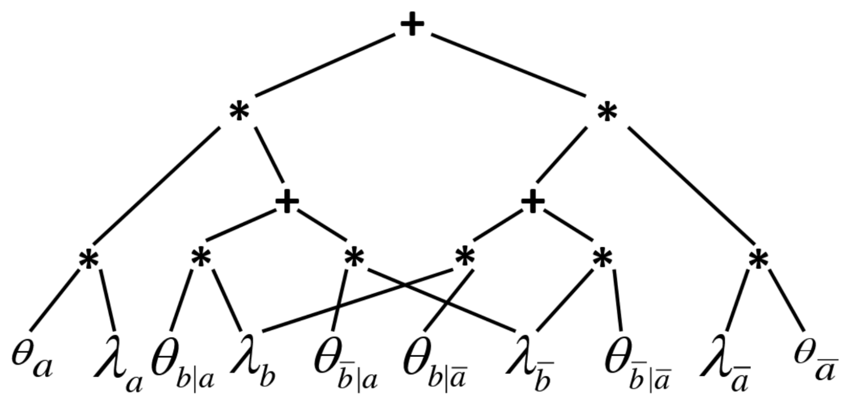


$$\boldsymbol{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \boldsymbol{P}(X_i \mid \mathbf{Pa}_{X_i})$$

Koller, Daphne, and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

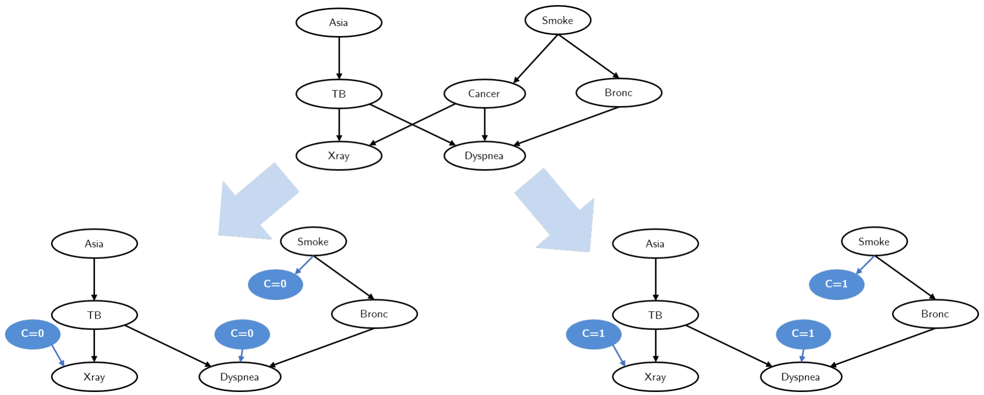
Exact inference in BNs

BY COMPILATION



Darwiche, Adnan. "A differential approach to inference in Bayesian networks." JACM (2003)

BY CUTSET CONDITIONING



Pearl, Judea. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann (1988).

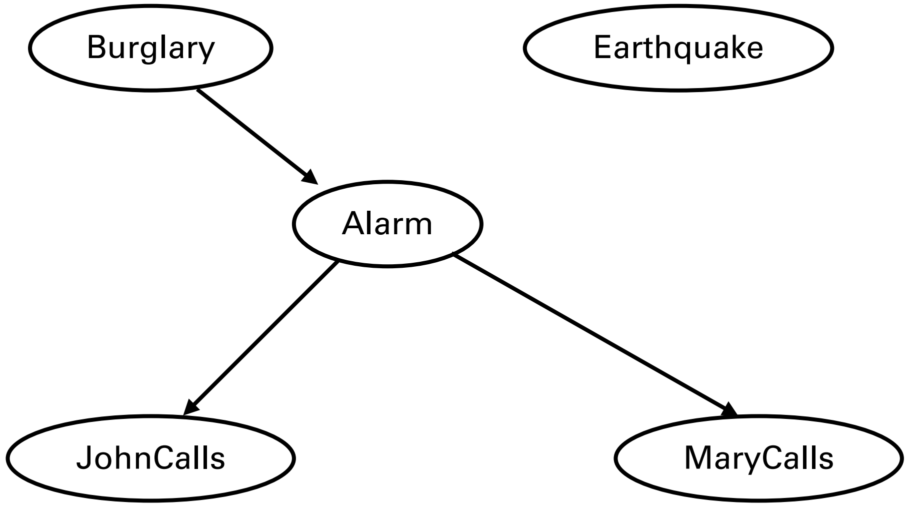
Dechter, Rina, and Robert Mateescu. "AND/OR search spaces for graphical models." *Artificial intelligence* (2007)

Probabilistic Graphical Models
Bayesian & Markov Networks

	EVI	MAR	MAP	MMAP
BNs & MNs*	✓	✗	✗	✗

Probabilistic Graphical Models
Tree Bayesian Networks

Type:	Directed PGM
Specific features:	BN with bounded number of parents
Learning:	Maximum likelihood Structure learning by chow-liu algorithm
Inference:	Tractable for EVI, MAR, CON
Limitation:	Limited representational power



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

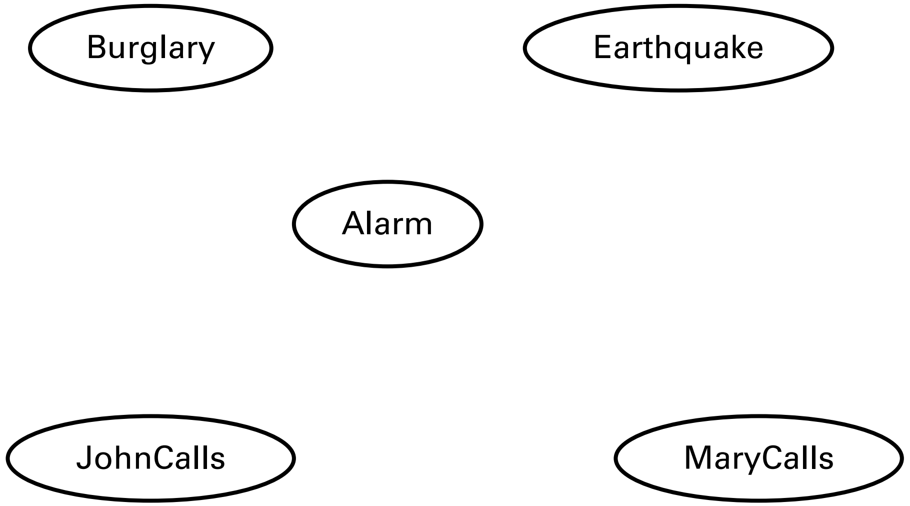
Koller, Daphne, and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Probabilistic Graphical Models
Tree Bayesian Networks

	EVI	MAR	MAP	MMAP
BNs & MNs*	✓	✗	✗	✗
Tree BNs/MNs	✓	✓	✗	✗

Probabilistic Graphical Models
Fully Factorized Models

Type:	PGM
Specific features:	BN/MN with no edges
Learning:	Maximum likelihood
Inference:	Tractable for EVI, MAR, CON, MAP, MMAP
Limitation:	Highly limited representational power Can't model any correlations



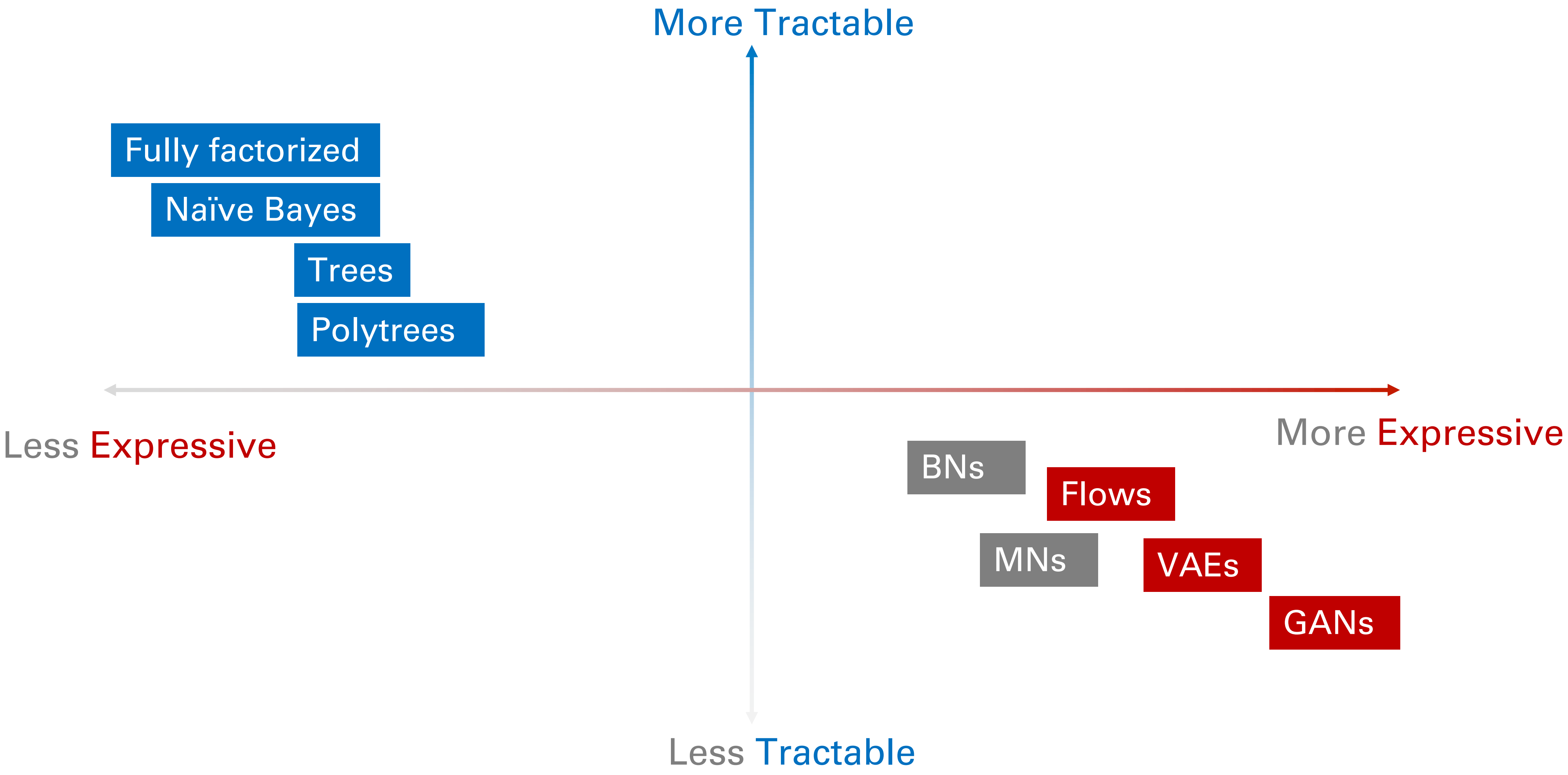
$$\boldsymbol{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \boldsymbol{P}(X_i)$$

Koller, Daphne, and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

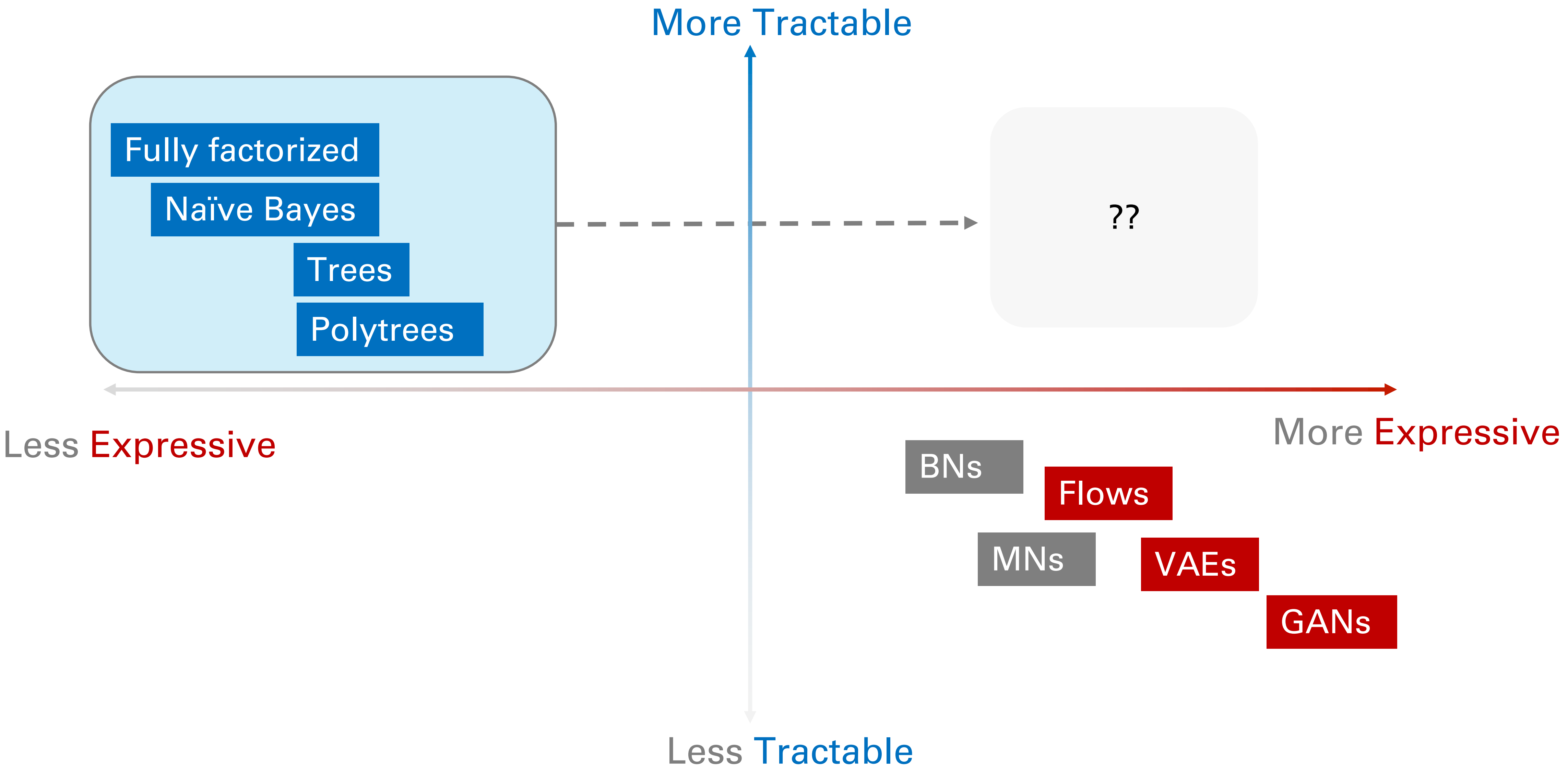
Probabilistic Graphical Models
Fully Factorized Models

	EVI	MAR	MAP	MMAP
BNs & MNs*	✓	✗	✗	✗
Tree BNs/MNs	✓	✓	✗	✗
Fully factorized	✓	✓	✓	✓

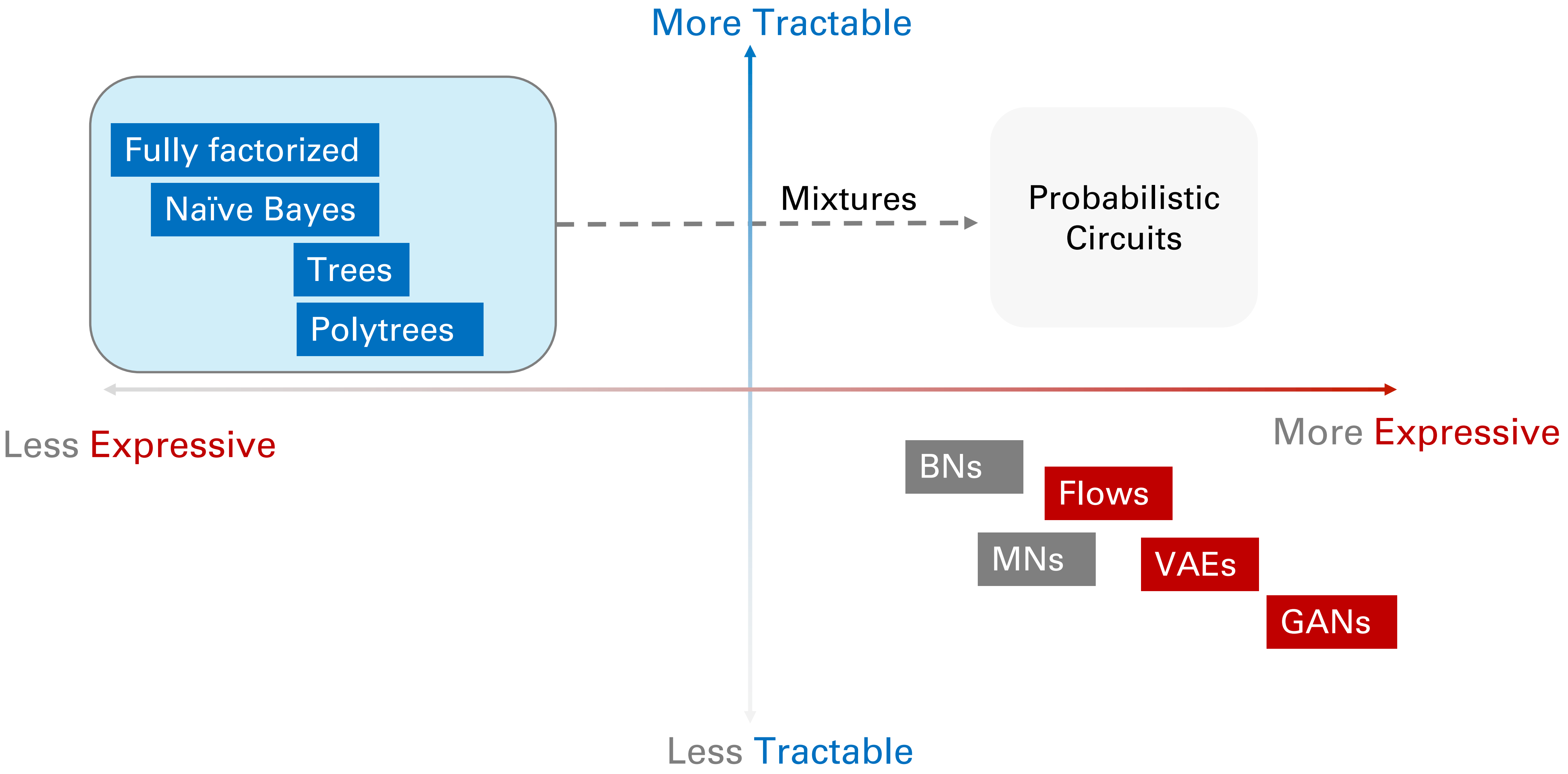
Structure Enables Tractability But Affects Expressivity



How to Improve Expressivity of Tractable Models?



How to Improve Expressivity of Tractable Models?



Probabilistic Circuits

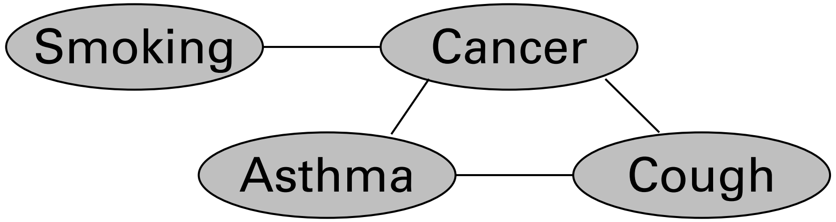
ACHIEVING THE BEST OF BOTH

DTPMs

Probabilistic Circuits

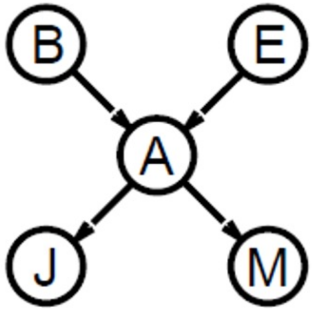
1. **Representing distributions using PCs**
2. Tractable Inference on PCs
3. Learning PCs
4. Expressive Deep Parameterizations

Revisiting Probabilistic Models

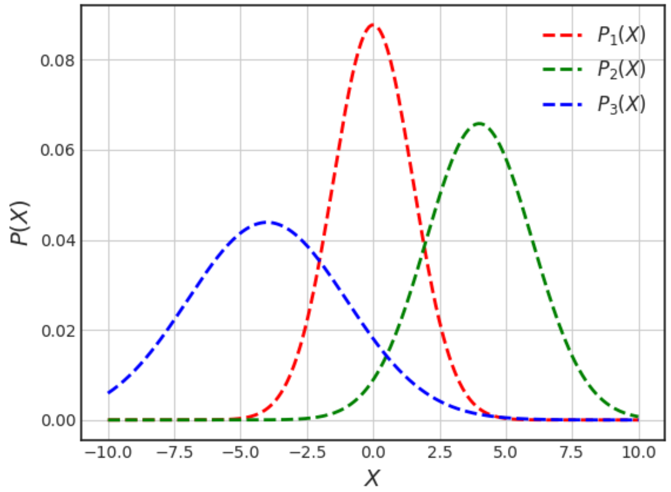


$$P(x) = \frac{1}{Z}\prod_c \Phi_c(x_c)$$

$$Z = \sum_x \prod_c \Phi_c(x_c)$$



$$\boldsymbol{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \boldsymbol{P}(X_i \mid \mathbf{Pa}_{X_i})$$

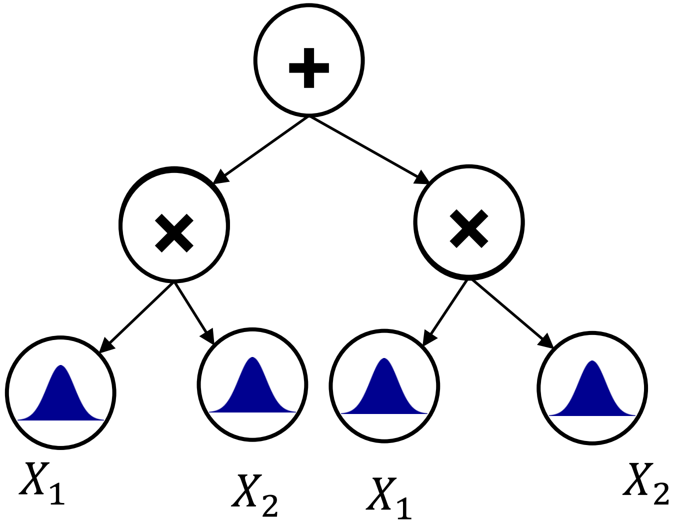


$$\boldsymbol{P}(X) = \sum_{\boldsymbol{i}} \boldsymbol{w_i P_i(X)}$$

Inference typically involves computing sums and products!

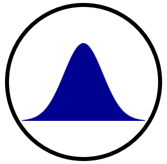
Probabilistic Circuits
Modeling Distributions as Computational Graphs

1. A probabilistic circuit C over variables $\boldsymbol{X}$ is a computational graph encoding a probability distribution $P(\boldsymbol{X})$
2. Comprises 3 types of Nodes
   - Leaf Nodes represent distributions
   - Internal nodes – Sums and Products
3. A language to define mixtures of simpler distributions

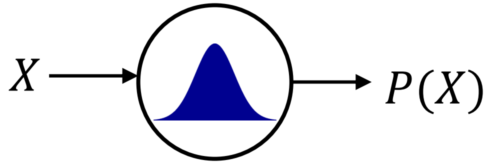


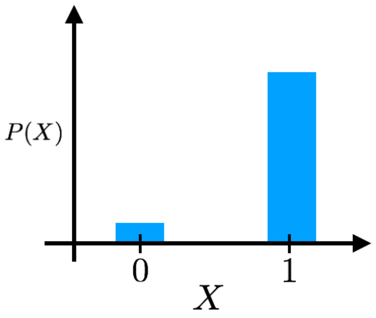
PC Building Blocks - Leaf Nodes

Simple Tractable Distributions

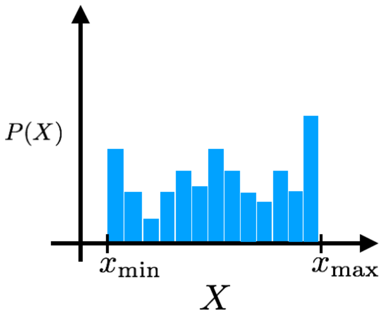


- Leaf node encodes a simple univariate tractable distribution
- Outputs the probability density or mass
- E.g., Gaussian, Categorical, Poisson, Tree-structured BN

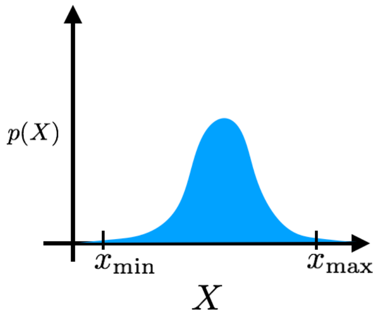




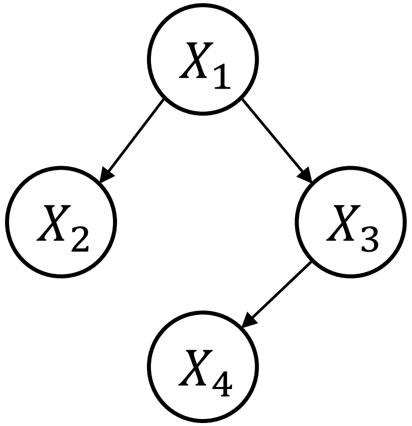
Bernoulli Distribution



Categorical Distribution

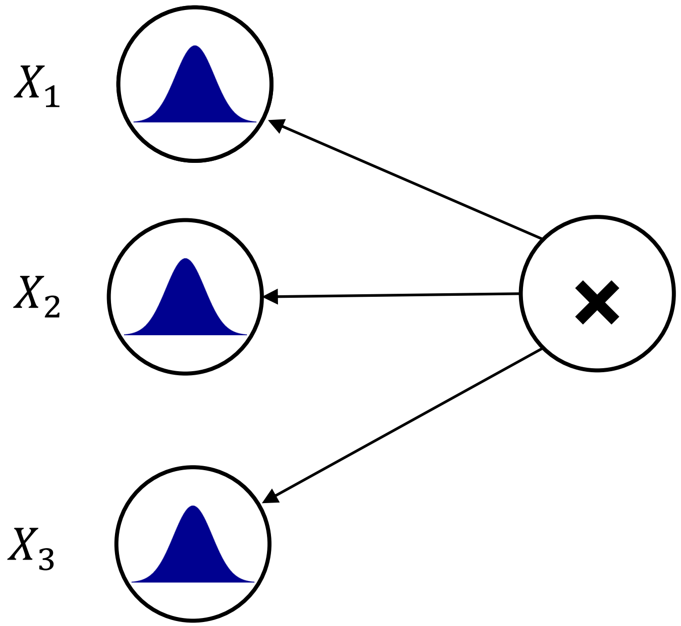


Gaussian Density



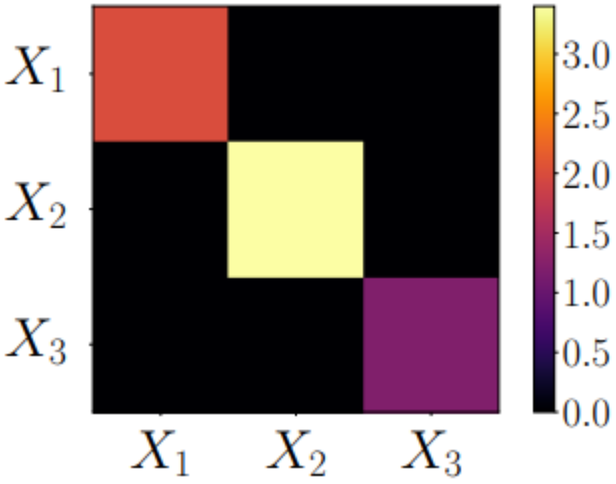
PC Building Blocks - Product Nodes Represent Factorizations

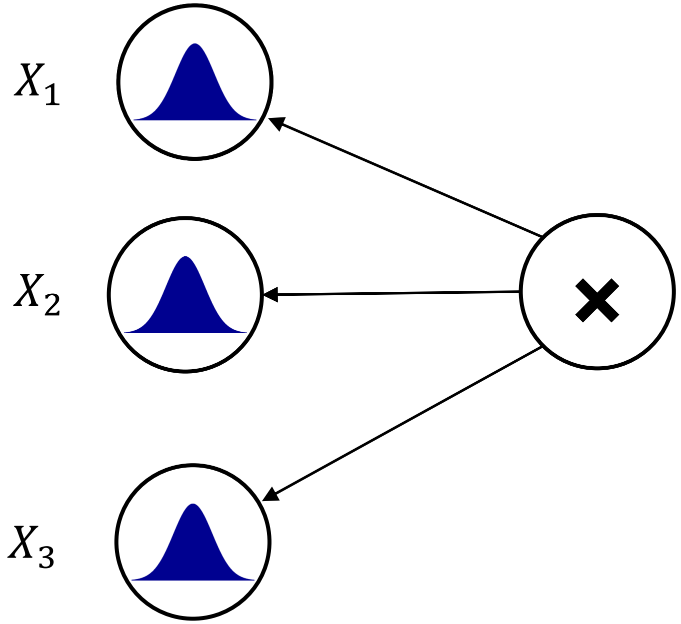
- Takes product of input distributions
  - $\times(X_1, X_2, X_3) = P(X_1)P(X_2)P(X_3)$
- Encodes independence
- Enables Tractability



PC Building Blocks - Product Nodes Represent Factorizations

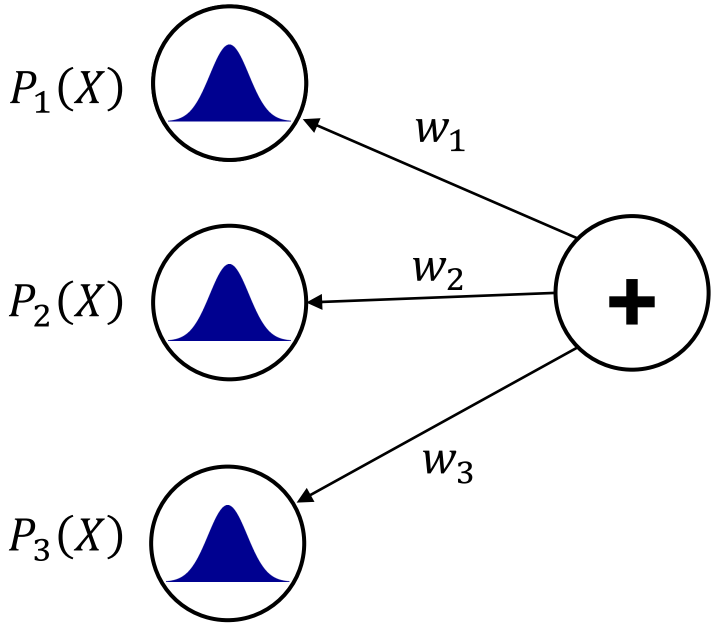
- Takes product of input distributions
  - $\times(X_1, X_2, X_3) = P(X_1)P(X_2)P(X_3)$
- Encodes independence
- Enables Tractability





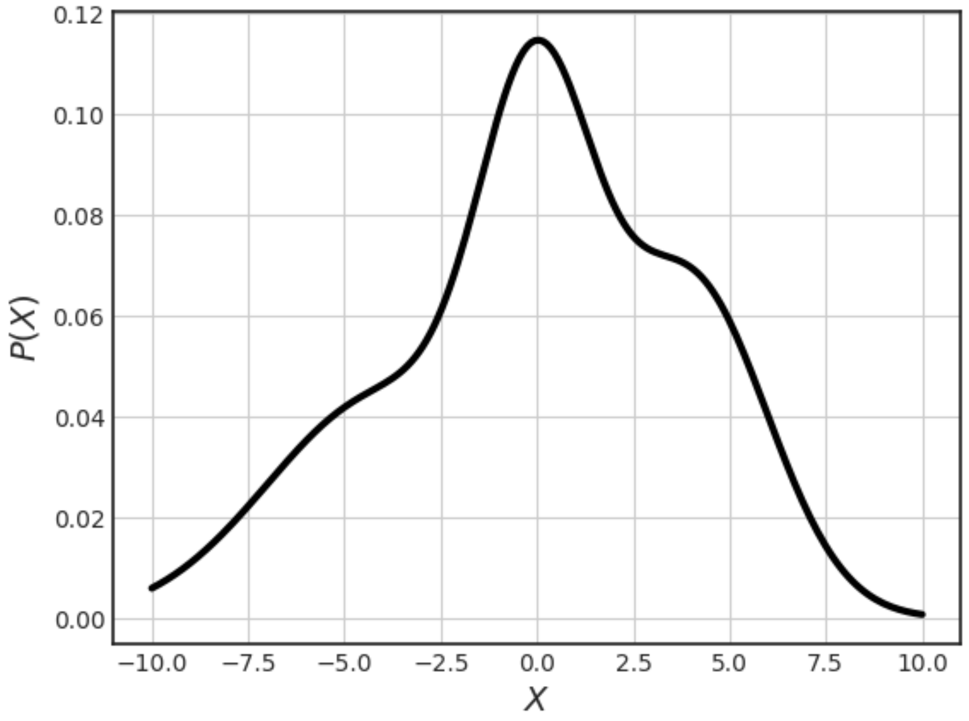
PC Building Blocks - Sum Nodes Represent Mixtures

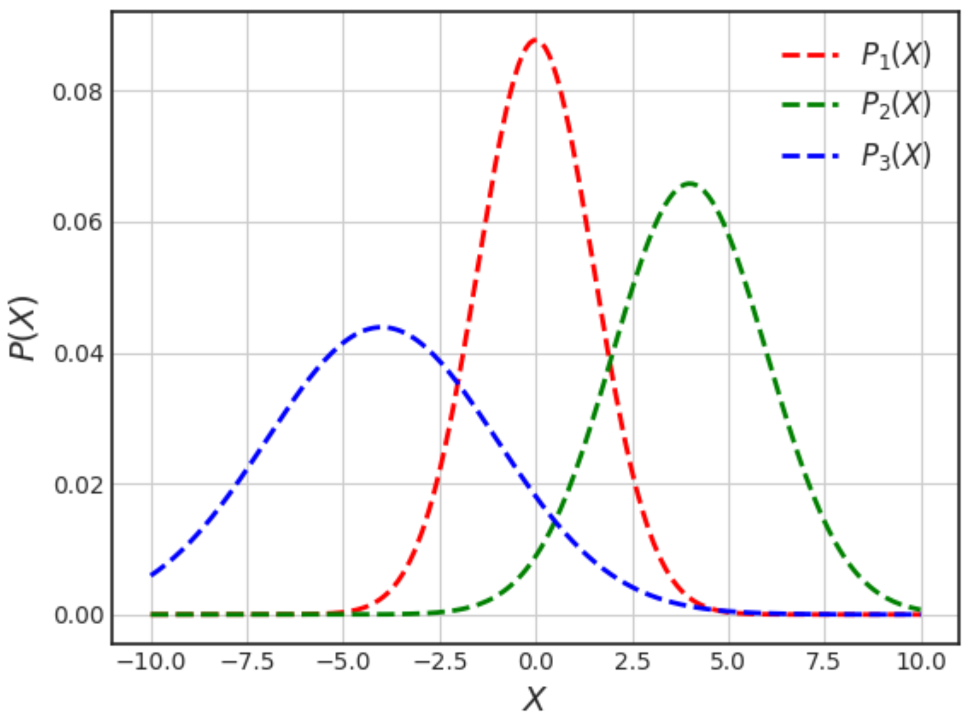
- Performs convex sum of inputs
- Add expressivity
  - Equivalent to mixtures



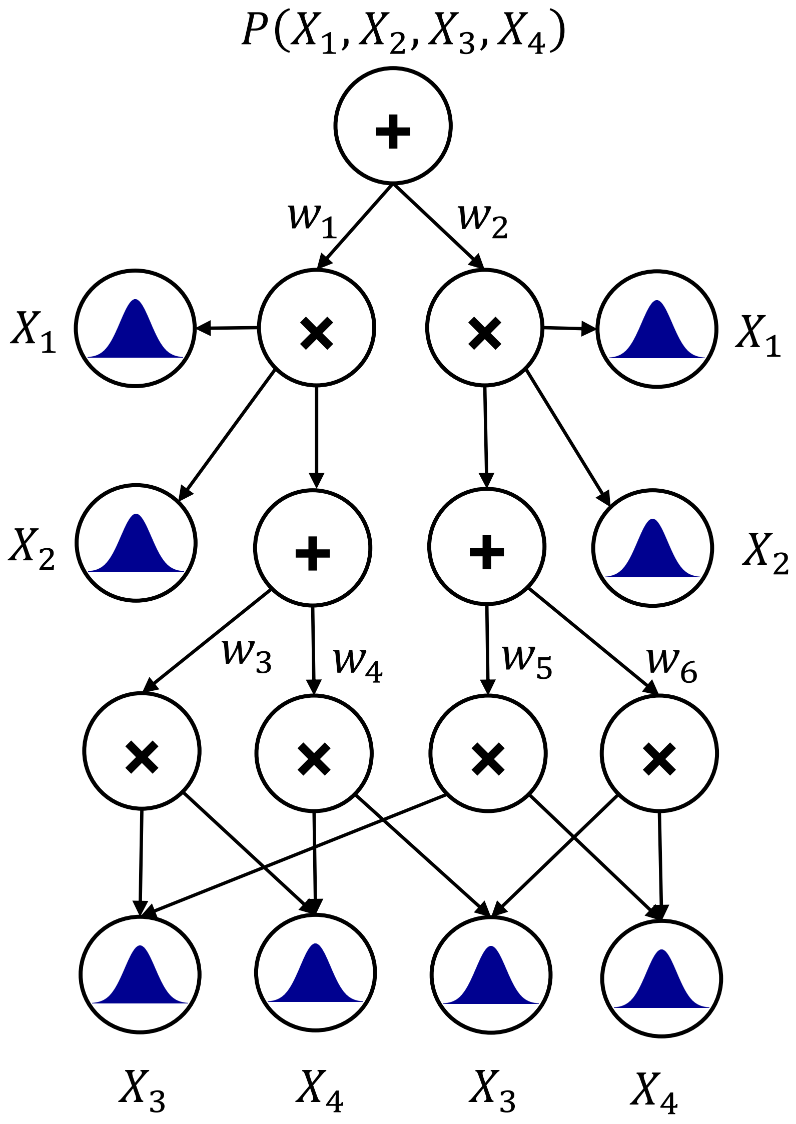
$$+(\boldsymbol{X}) = \boldsymbol{w_1 P_1}(\boldsymbol{X}) + \boldsymbol{w_2 P_2}(\boldsymbol{X}) + \boldsymbol{w_3 P_3}(\boldsymbol{X})$$

Mixtures Improve Expressivity





$$P(X) = w_1 P_1(X) + w_2 P_2(X) + w_3 P_3(X)$$
$$P_i(X) = N(\mu_i, \sigma_i)$$



Probabilistic Circuits

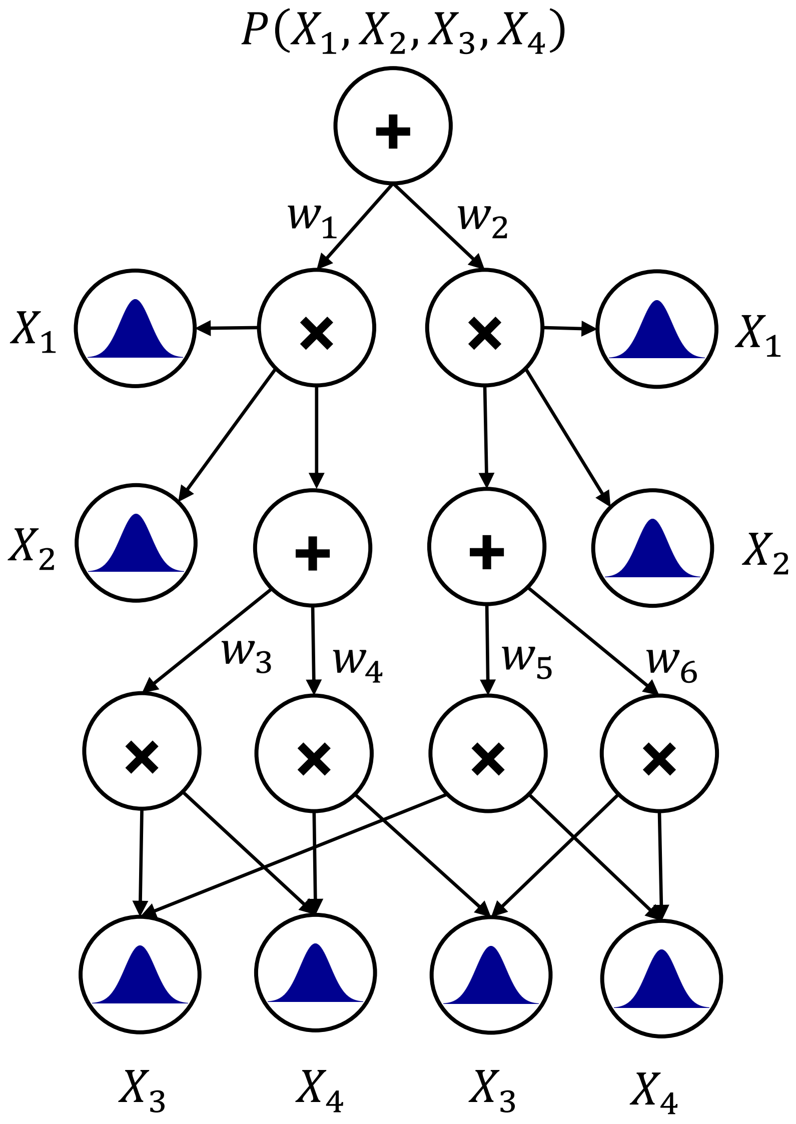
Stack together as a computational graph!

Evaluated bottom up
Joint density given by output of root node

DTPMs

Probabilistic Circuits

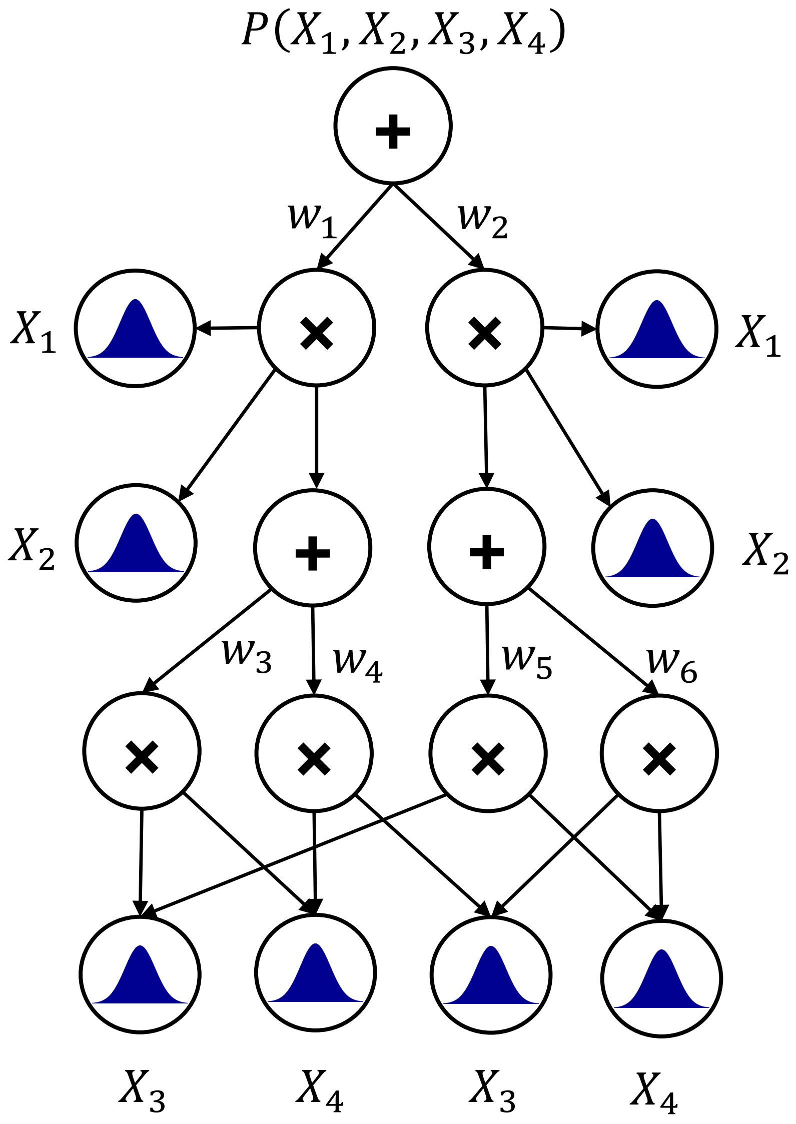
1. Representing distributions using PCs
2. **Tractable Inference on PCs**
3. Learning PCs
4. Expressive Deep Parameterizations



Probabilistic Circuits

Stack together as a computational graph!

Just stack nodes arbitrarily?



Probabilistic Circuits

Stack together as a computational graph!

Just stack nodes arbitrarily?

No! You need some structure!

Tractability Requires Imposing Structure

(1) Smoothness. Scope of each child of sum node is identical

(2) Decomposability. Scope of each child of product node is disjoint

(3) Determinism. At most one child of sum node is non-zero

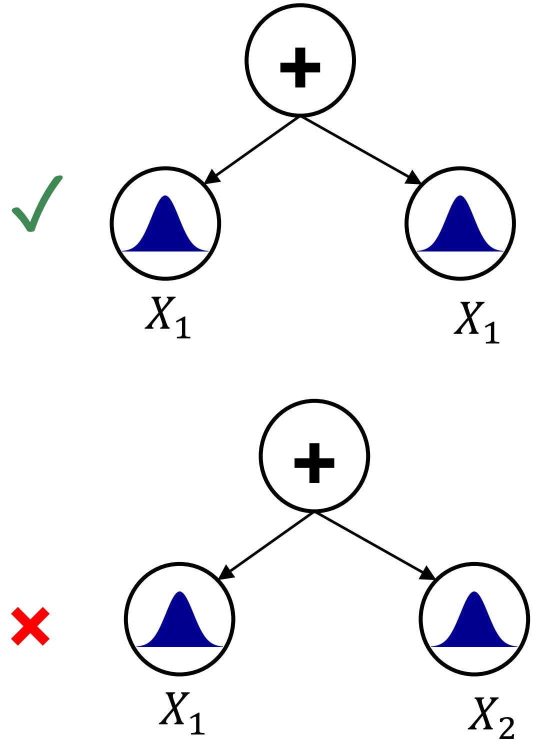
(4) ..

Rahman, Tahrima, Prasanna Kothalkar, and Vibhav Gogate. "Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees." ECML PKDD 2014,

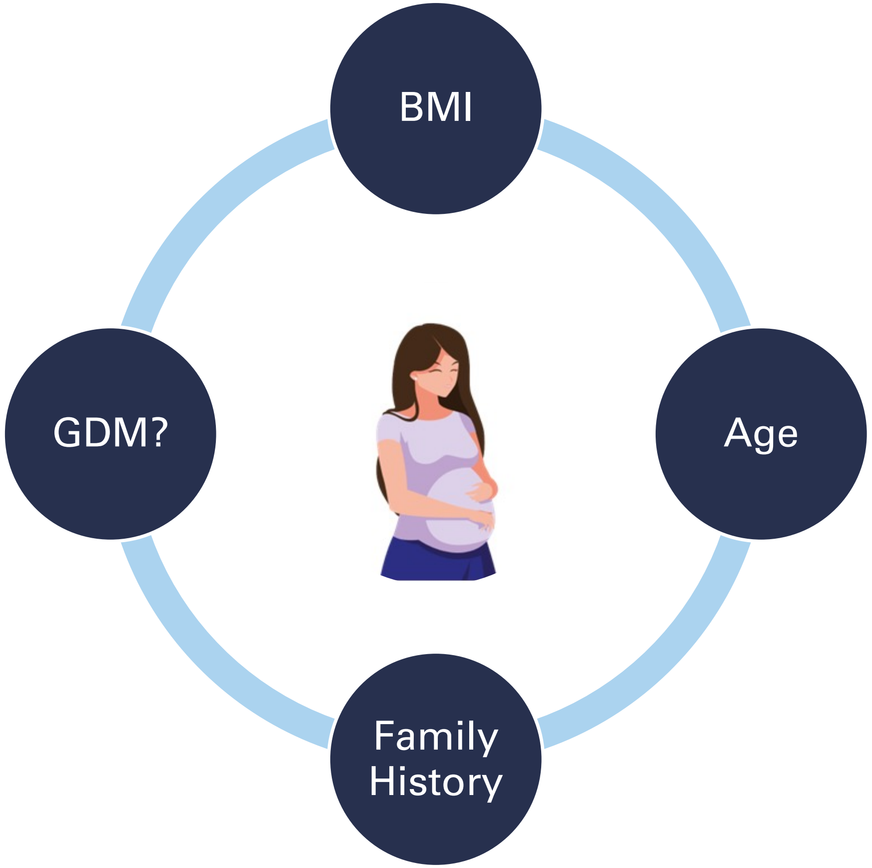
Choi, YooJung, Vergari, Antonio, and Van den Broeck, Guy. “Probabilistic circuits: A unifying framework for tractable probabilistic modeling.” (2020).

Structural Properties
(1) Smoothness

- A sum node is smooth if all its children have the same scope
- A PC is smooth if all its sum nodes are smooth
- Ensures valid mixtures
- Tractable EVI by bottom-up evaluation



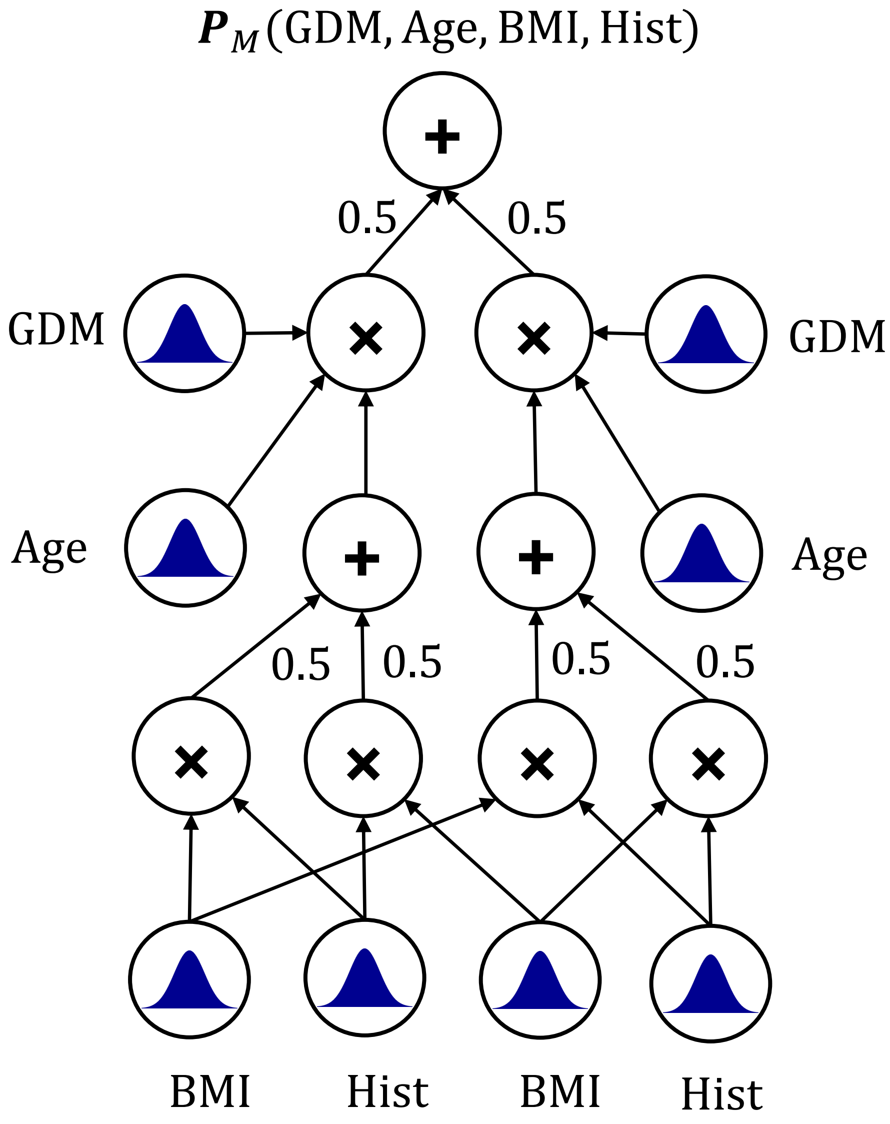
Tractable Inference Using PCs
Evidential Inference (EVI)



Q1: What is the likelihood of observing a 35 year old pregnant woman with gestational diabetes having a BMI of 25, and a family history of diabetes?

$$\boldsymbol{P}_M(\text{GDM} = 1, \text{Age} = 35, \text{BMI} = 25, \text{Hist} = 1)$$

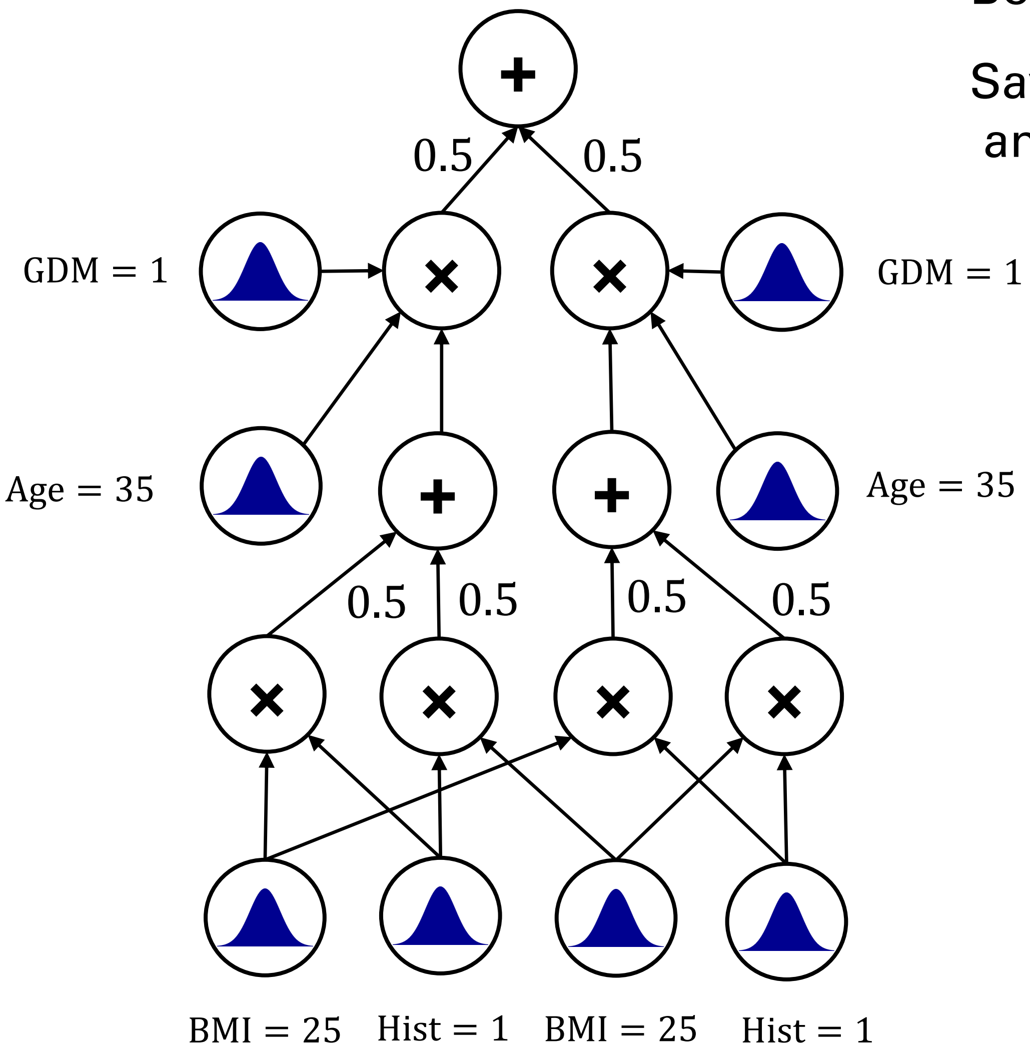
Tractability for EVI: Evaluate the PC!



Bottom-up evaluation gives tractable EVI if smooth!

Say, given $\mathrm{GDM} = 1, \mathrm{Age} = 35,\ \mathrm{BMI} = 25, \mathrm{Hist} = 1$
and uniform sum weights $w_i = 0.5$

Tractability for EVI: Evaluate the PC!

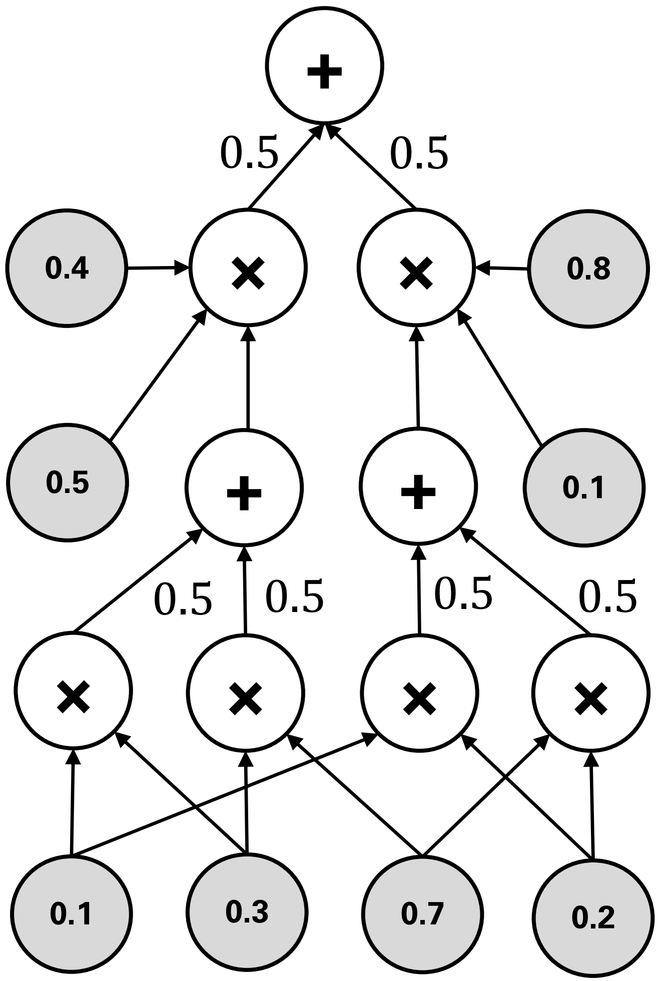


Bottom-up evaluation gives tractable EVI if smooth!

Say, given $GDM = 1, Age = 35,\ BMI = 25, Hist = 1$ and uniform sum weights $w_i = 0.5$

1. Plugin the values of variables to the leaves to get the corresponding PDF/PMF

Tractability for EVI: Evaluate the PC!

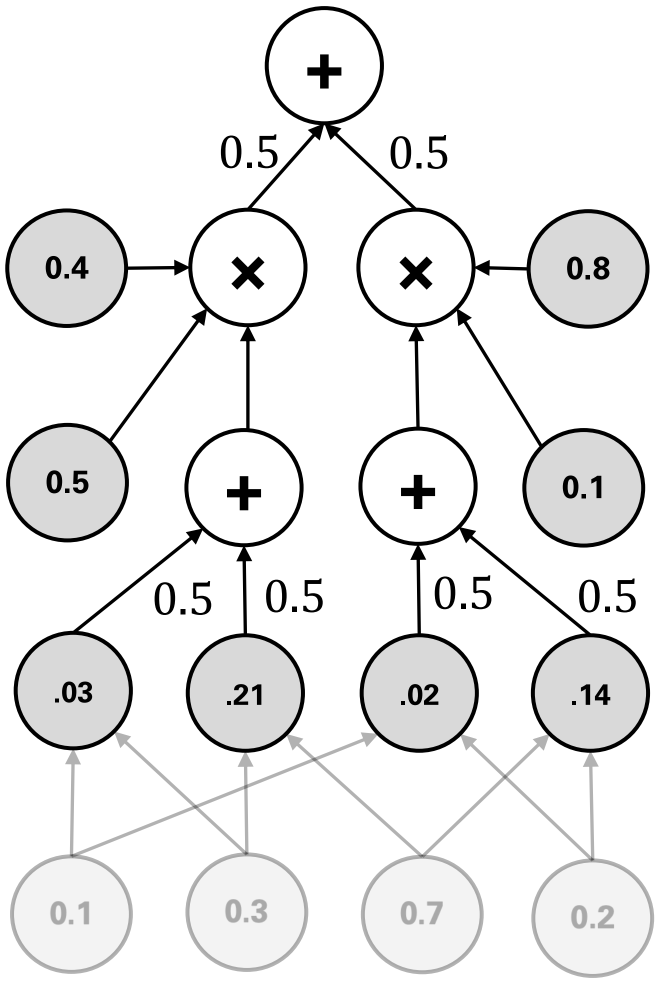


Bottom-up evaluation gives tractable EVI if smooth!

Say, given $GDM = 1, Age = 35,\ BMI = 25, Hist = 1$ and uniform sum weights $w_i = 0.5$

1. Plugin the values of variables to the leaves to get the corresponding PDF/PMF

Tractability for EVI: Evaluate the PC!



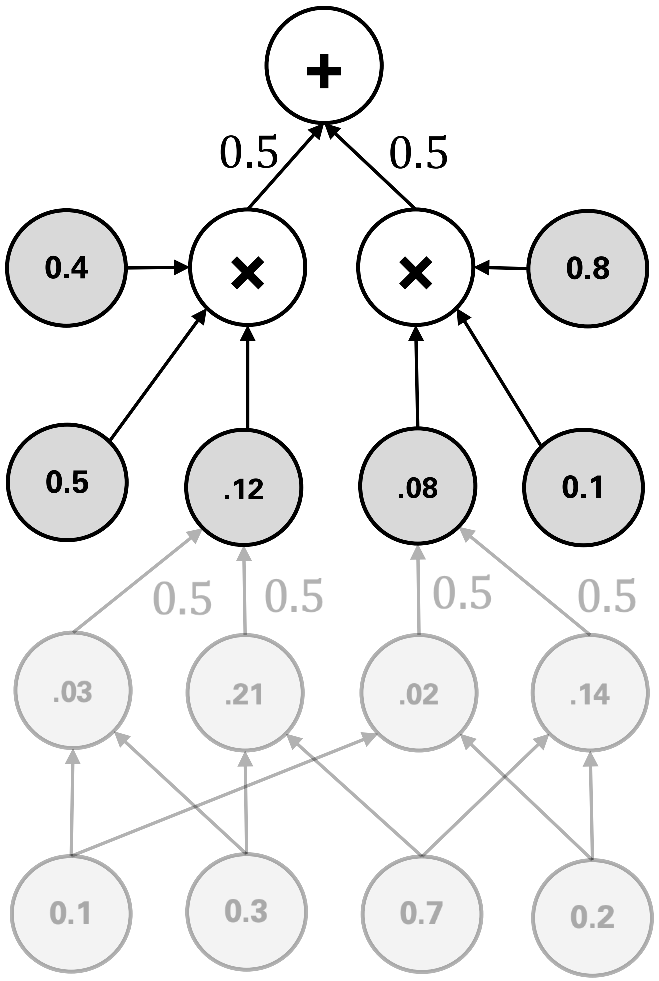
Bottom-up evaluation gives tractable EVI if smooth!

Say, given $GDM = 1, Age = 35, \ BMI = 25, Hist = 1$ and uniform sum weights $w_i = 0.5$

1. Plugin the values of variables to the leaves to get the corresponding PDF/PMF

2. Propagate the values upwards by performing sums and products represented by the computational graph

Tractability for EVI: Evaluate the PC!

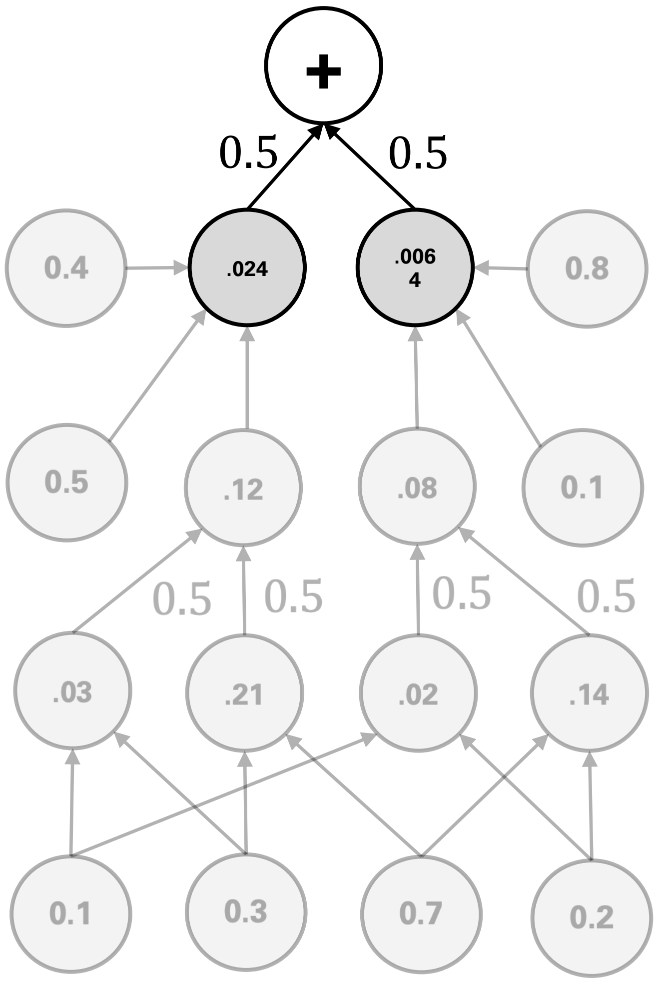


Bottom-up evaluation gives tractable EVI if smooth!

Say, given $GDM = 1, Age = 35,\ BMI = 25, Hist = 1$ and uniform sum weights $w_i = 0.5$

1. Plugin the values of variables to the leaves to get the corresponding PDF/PMF
2. Propagate the values upwards by performing sums and products represented by the computational graph
3. Recurse till you reach the root

Tractability for EVI: Evaluate the PC!



Bottom-up evaluation gives tractable EVI if smooth!

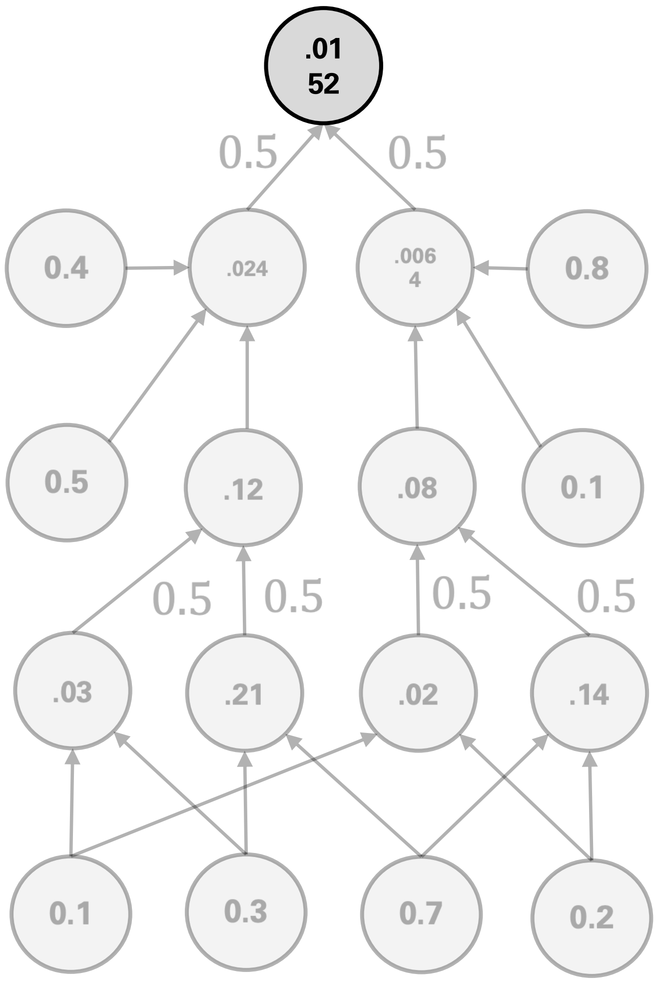
Say, given $GDM = 1, Age = 35,\ BMI = 25, Hist = 1$ and uniform sum weights $w_i = 0.5$

1. Plugin the values of variables to the leaves to get the corresponding PDF/PMF

2. Propagate the values upwards by performing sums and products represented by the computational graph

3. Recurse till you reach the root

Tractability for EVI: Evaluate the PC!



Bottom-up evaluation gives tractable EVI if smooth!

Say, given $GDM = 1, Age = 35, \ BMI = 25, Hist = 1$
and uniform sum weights $w_i = 0.5$

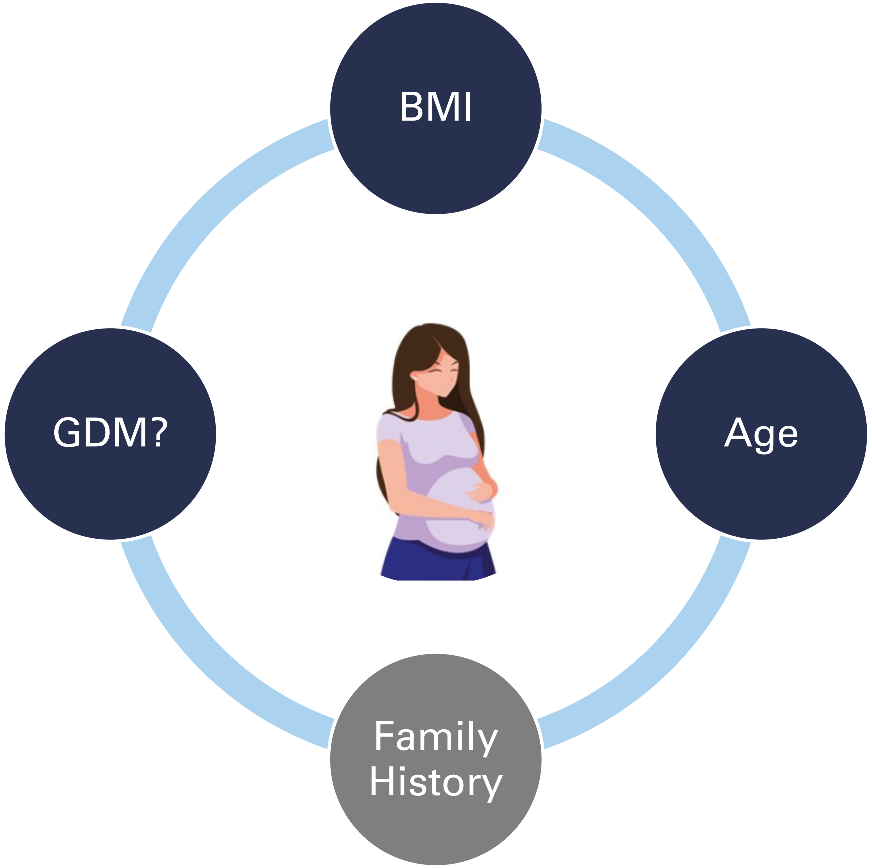
1. Plugin the values of variables to the leaves to get the corresponding PDF/PMF

2. Propagate the values upwards by performing sums and products represented by the computational graph

3. Recurse till you reach the root

$$\boldsymbol{P}_M(\mathrm{GDM} = 1, \mathrm{Age} = 35, \mathrm{BMI} = 25, \mathrm{Hist} = 1) = 0.0152$$

Tractable Inference Using PCs
Marginal Inference (MAR)



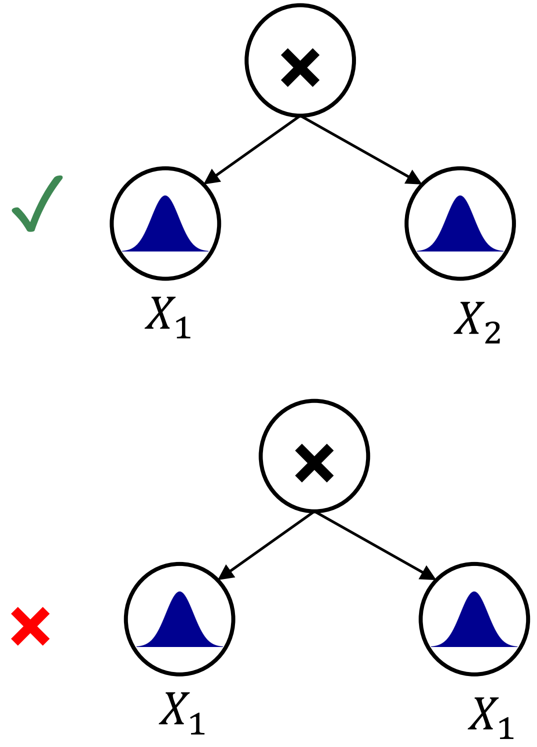
Q1: What is the likelihood of observing a 35 year old pregnant woman with gestational diabetes having a BMI of 25 ~~and a family history of diabetes~~?

$$\boldsymbol{P}_M(\mathrm{GDM}=1, \mathrm{Age}=35, \mathrm{BMI}=25)$$

$$= \sum_{x' \in \{0,1\}} \boldsymbol{P}_M(\mathrm{GDM}=1, \mathrm{Age}=35, \mathrm{BMI}=25, \mathrm{Hist}=x')$$

Structural Properties
(2) Decomposability

- A product node is decomposable if the scopes of its children are disjoint
- A PC is decomposable if all its product nodes are decomposable

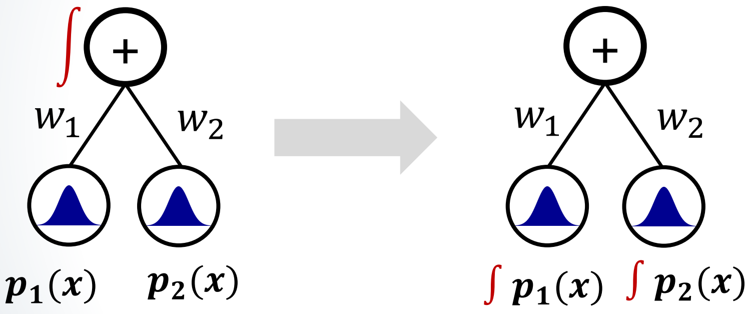


Tractability for MAR
Smooth and Decomposable PC

Computing MAR involves integrating (or summing) out the values of a variable

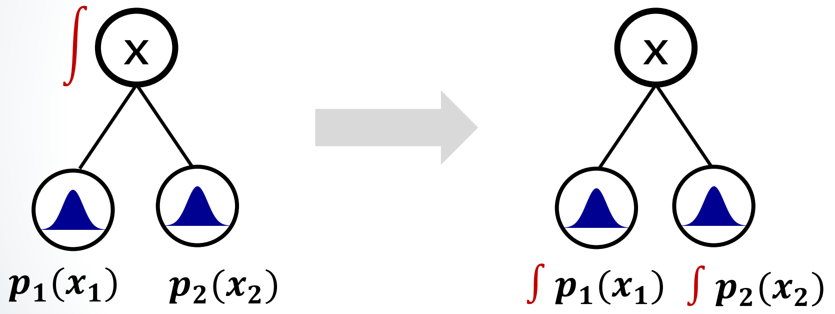
Smooth Sum Node

$$\int p(\boldsymbol{x})\, d\boldsymbol{x} = \int w_1\, p_1(\boldsymbol{x}) + w_2 p_2(\boldsymbol{x})\, d\boldsymbol{x}$$
$$= w_1 \int p_1(\boldsymbol{x}) d\boldsymbol{x} \;+\; w_2 \int p_2(\boldsymbol{x}) d\boldsymbol{x}$$



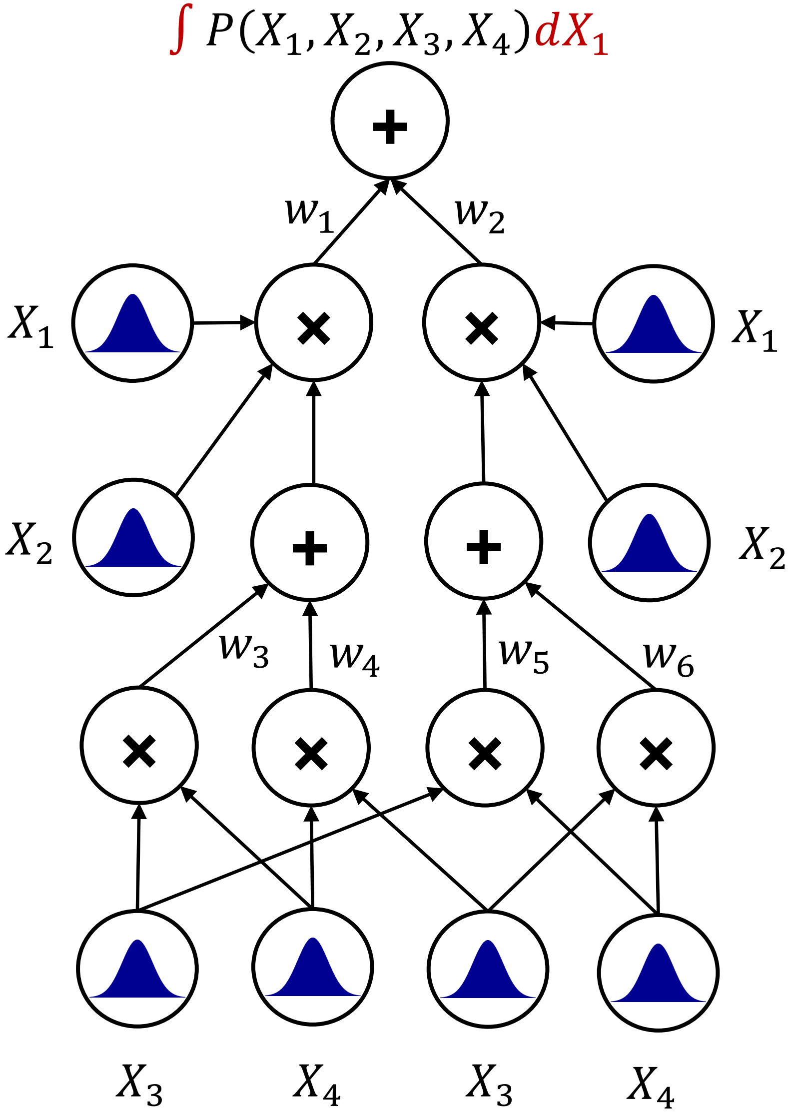
Decomposable Product Node

$$\int p(\boldsymbol{x_1}, \boldsymbol{x_2})\, d\boldsymbol{x_1} d\boldsymbol{x_2} = \int p_1(\boldsymbol{x_1}) p_2(\boldsymbol{x_2}) d\boldsymbol{x_1} d\boldsymbol{x_2}$$
$$= \left(\int p_1(\boldsymbol{x_1}) d\boldsymbol{x_1}\right) . \left(\int p_2(\boldsymbol{x_2})\; d\boldsymbol{x_2}\right)$$



Can push down integrals from a node to its children!

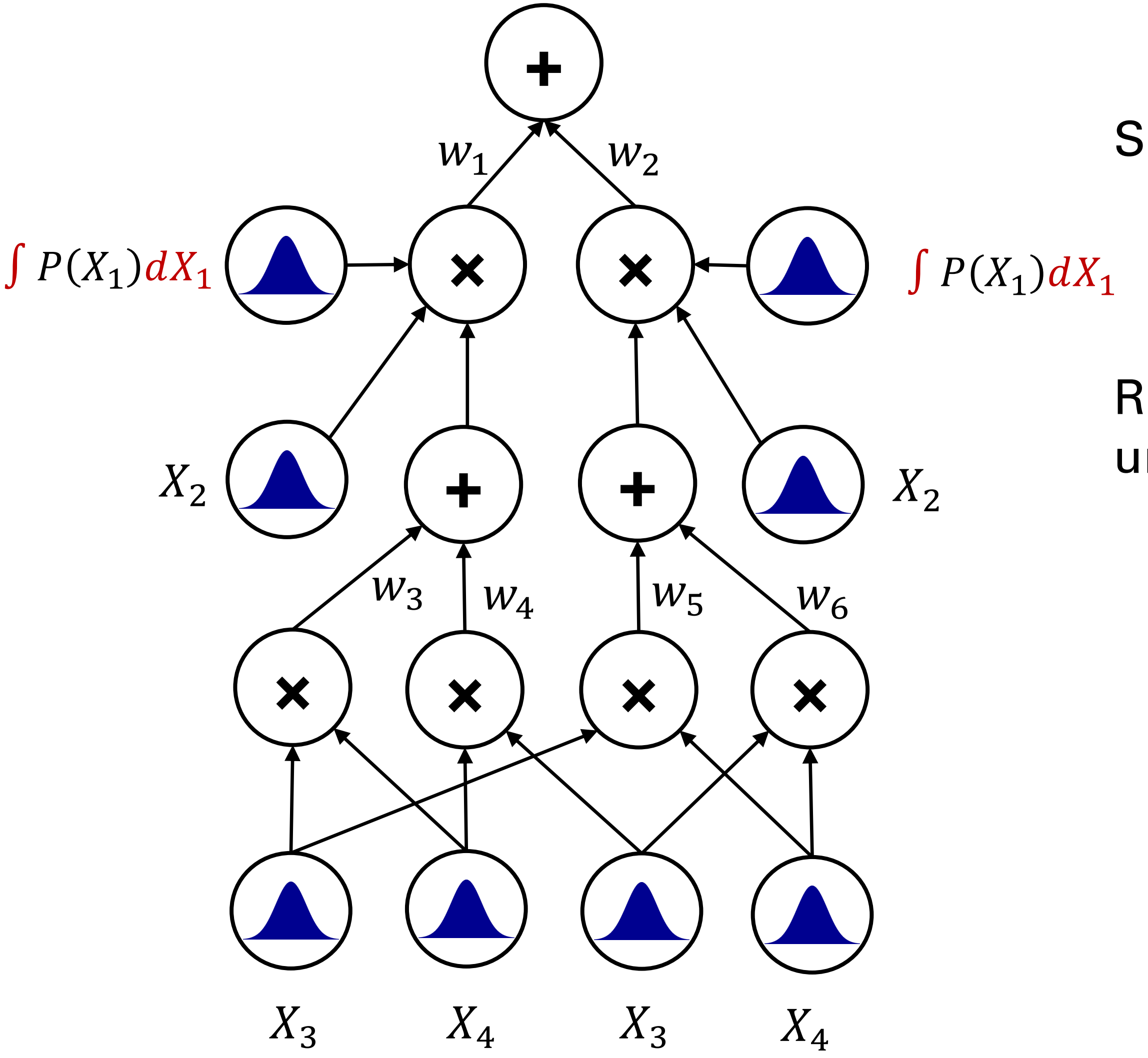
Tractability for MAR



Say, we wish to marginalize out X_1 i.e compute

$$P(X_2, X_3, X_4) = \int P(X_1, X_2, X_3, X_4) dX_1$$

Tractability for MAR

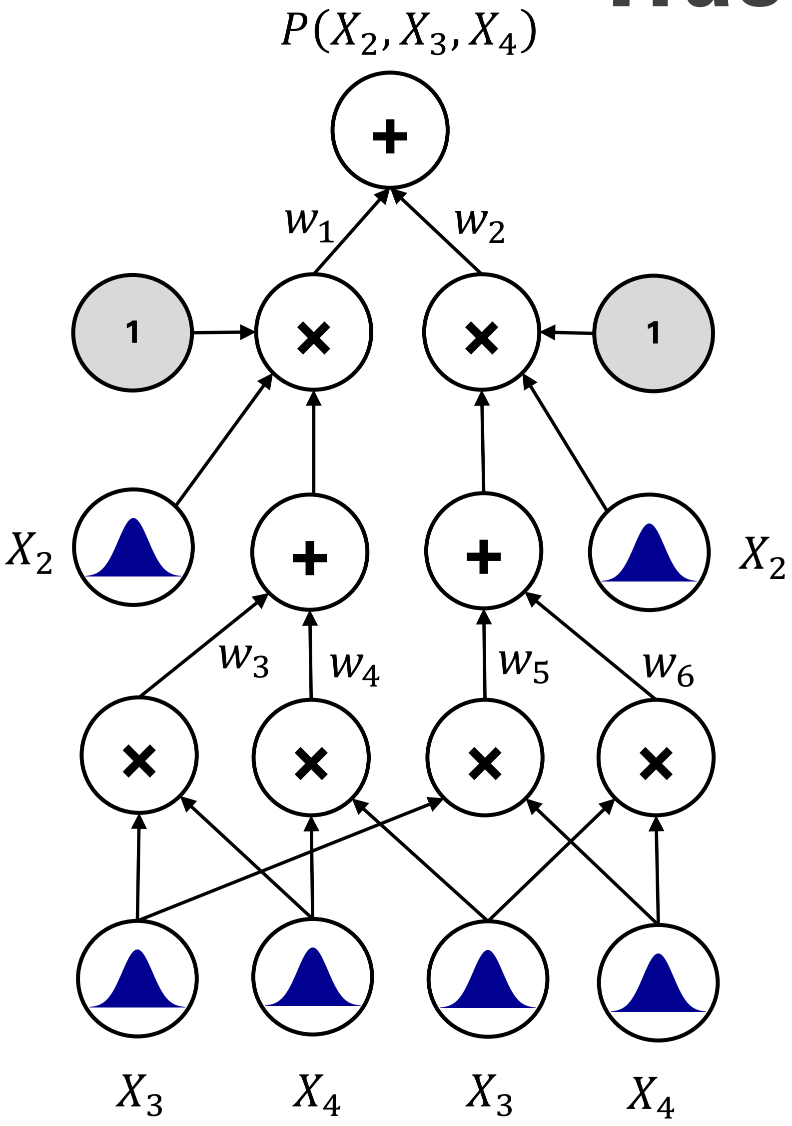


Say, we wish to marginalize out X_1 i.e compute

$$P(X_2, X_3, X_4) = \int P(X_1, X_2, X_3, X_4)dX_1$$

Reduces to marginalizing corresponding univariate leaf distributions!

Tractability for MAR



Say, we wish to marginalize out X_1 i.e compute

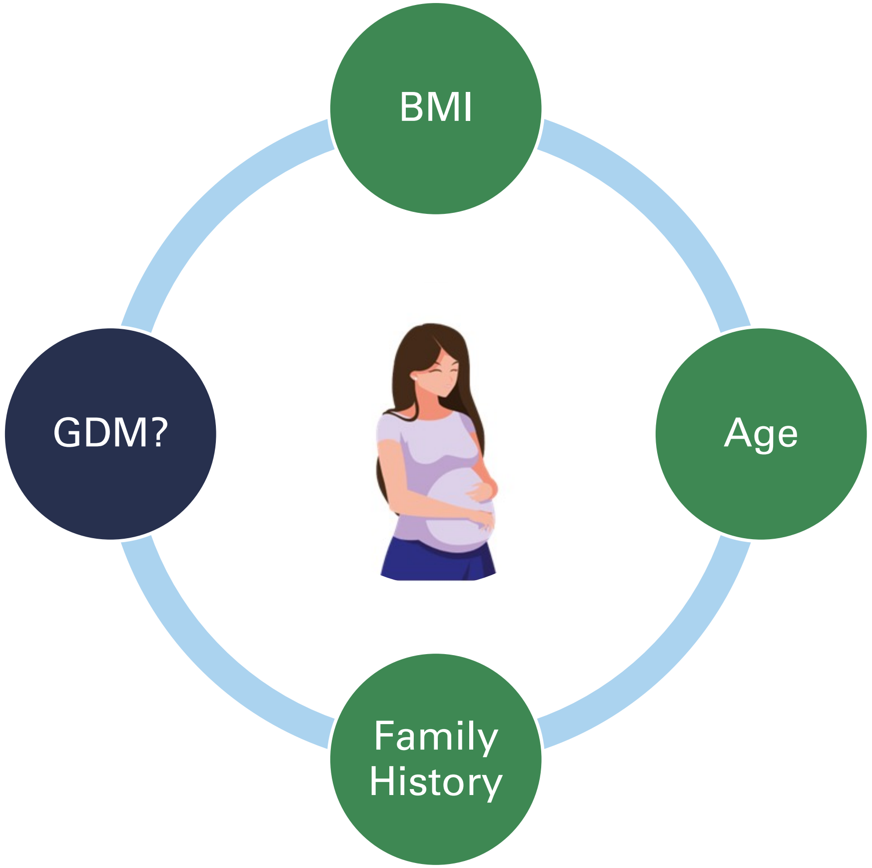
$$P(X_2, X_3, X_4) = \int P(X_1, X_2, X_3, X_4) dX_1$$

Reduces to marginalizing corresponding univariate leaf distributions!

Complete Marginalization:

- Set corresponding leaf nodes to 1
- Bottom up eval by setting other variables
- Linear time!

Tractable Inference Using PCs
Conditional Inference (CON)



Q3: What is the likelihood that a pregnant woman has gestational diabetes given that she is 35 years old, has BMI of 25, and a family history of diabetes?

$$\boldsymbol{P}_M(\mathrm{GDM}=1 \mid \mathrm{Age}=35, \mathrm{BMI}=25, \mathrm{Hist}=1)$$

$$= \frac{\boldsymbol{P}_M(\mathrm{GDM}=1, \mathrm{Age}=35, \mathrm{BMI}=25, \mathrm{Hist}=1)}{\sum_{x \epsilon \{0,1\}} \boldsymbol{P}_M(\mathrm{GDM}=x, \mathrm{Age}=35, \mathrm{BMI}=25, \mathrm{Hist}=1)}$$

Tractability for CON

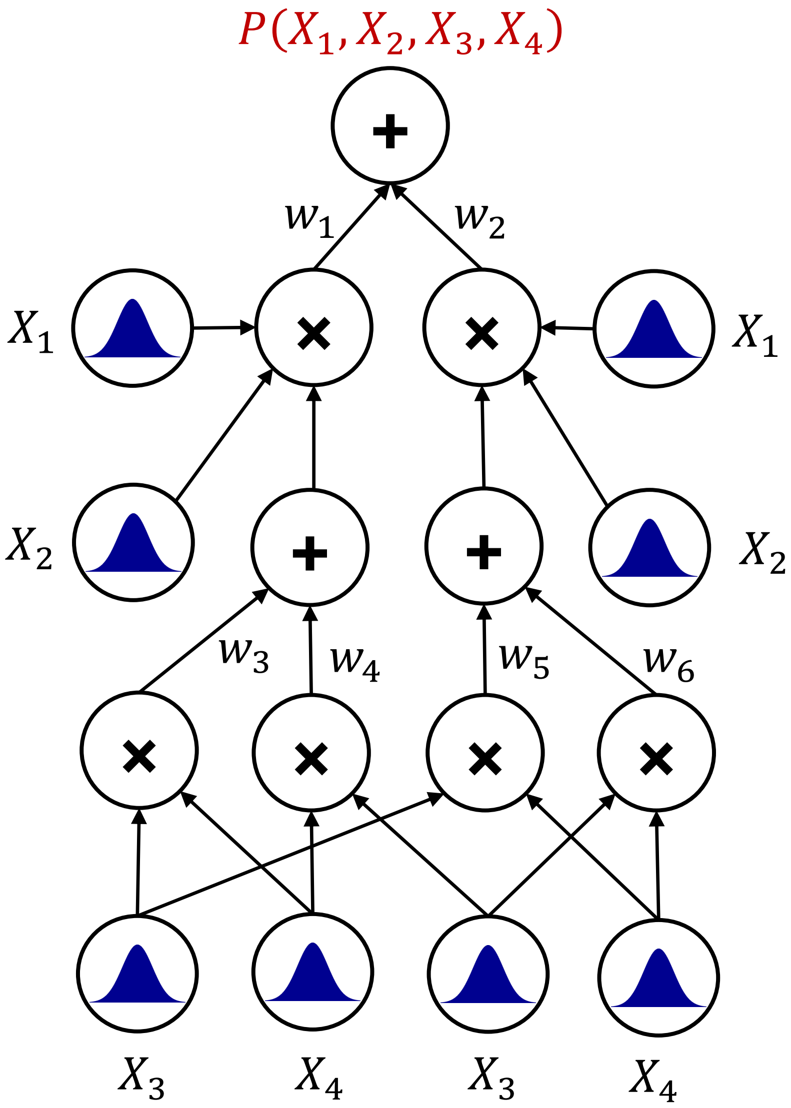
Say, we wish to marginalize out X_1
i.e compute $P(X_2, X_3, X_4 | X_1)$

$$= \frac{P(X_1, X_2, X_3, X_4)}{P(X_1)}$$

Tractability for CON

Say, we wish to marginalize out X_1
i.e compute $P(X_2, X_3, X_4 | X_1)$

$$= \frac{P(X_1, X_2, X_3, X_4)}{P(X_1)}$$

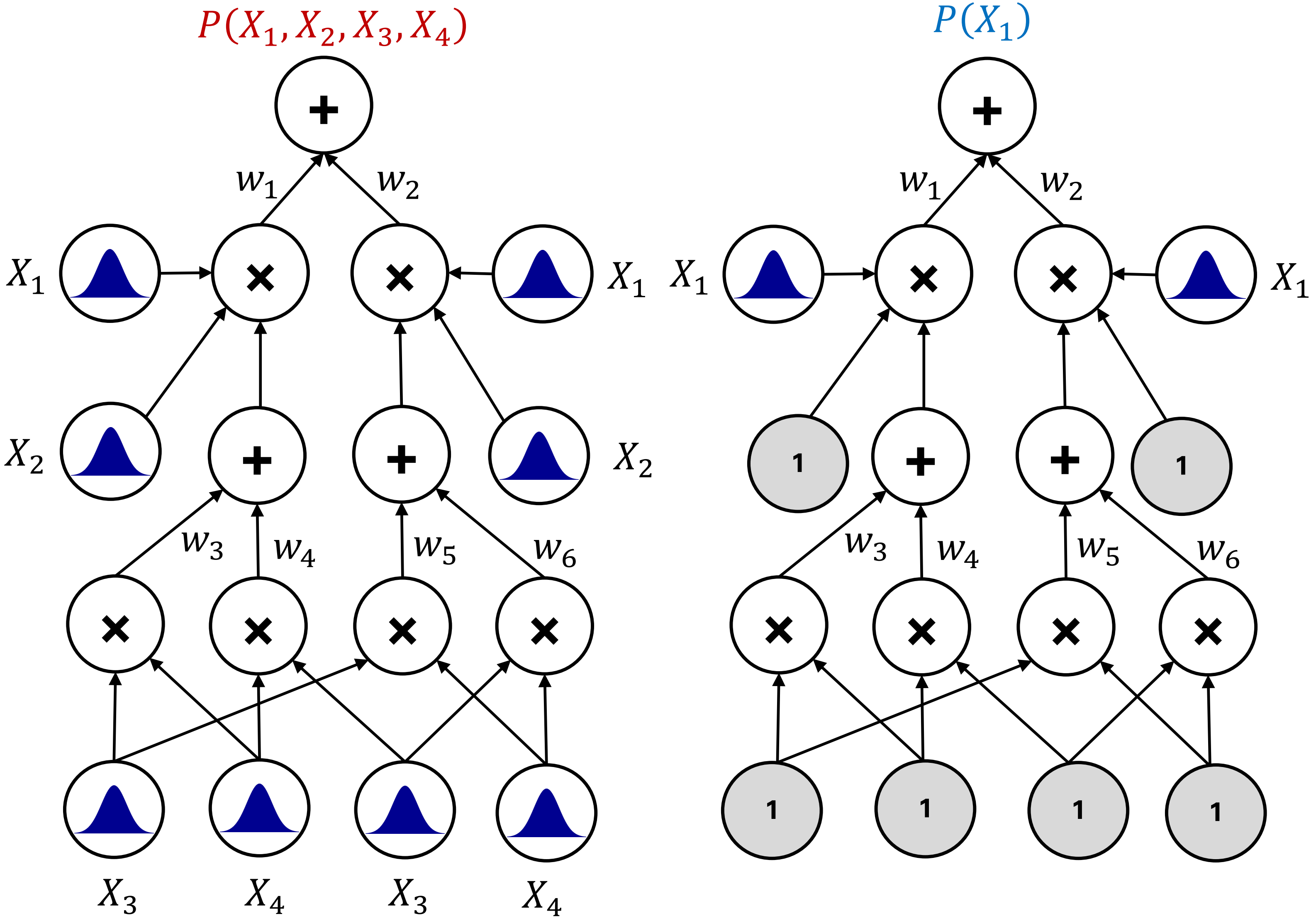


Tractability for CON

Say, we wish to marginalize out X_1
i.e compute $P(X_2, X_3, X_4|X_1)$

$$= \frac{P(X_1, X_2, X_3, X_4)}{P(X_1)}$$

$$= \frac{P(X_1, X_2, X_3, X_4)}{\int P(X_1, X_2, X_3, X_4) dX_2 dX_3 dX_4}$$



Tractability for CON

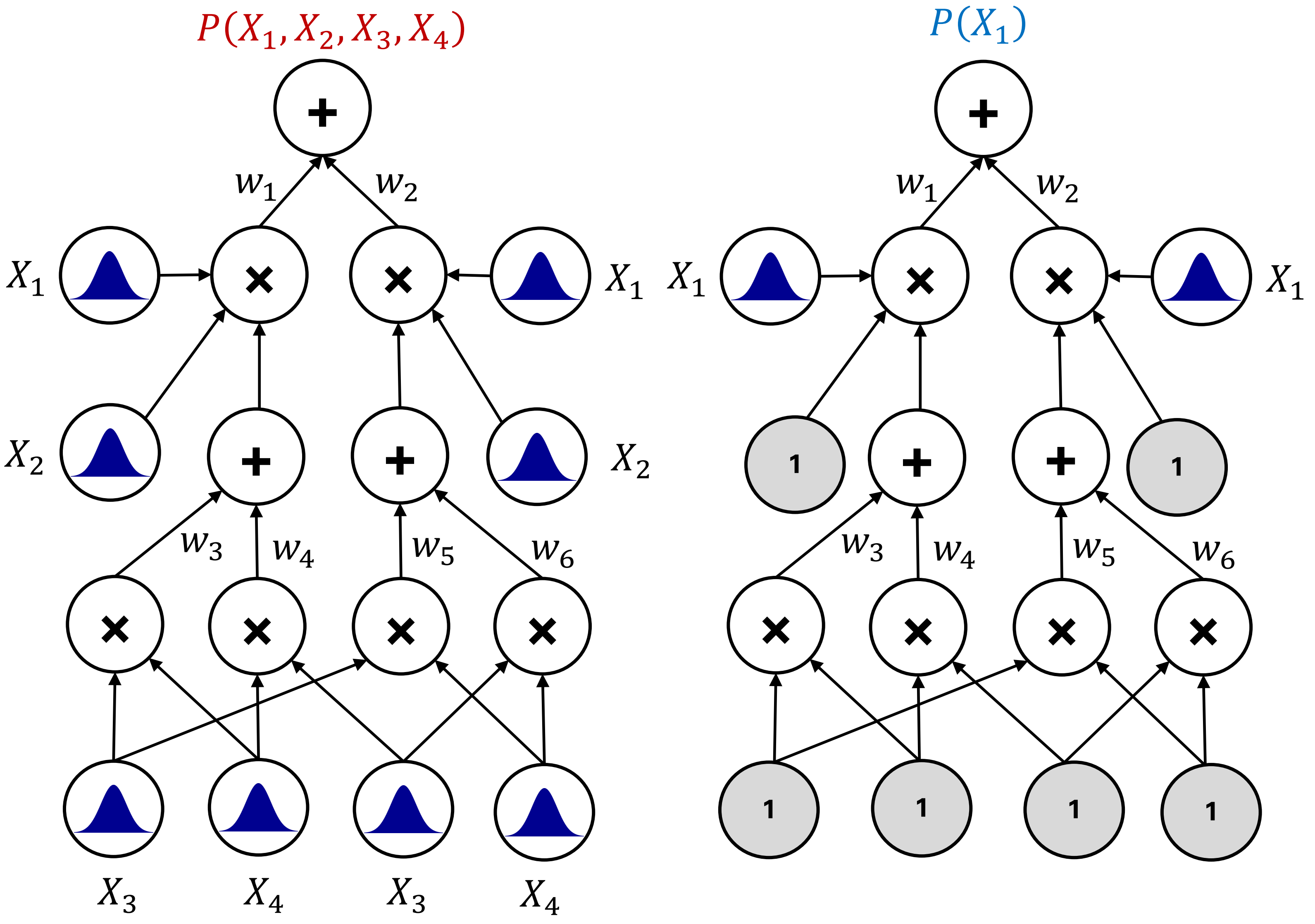
Say, we wish to marginalize out X_1
i.e compute $P(X_2, X_3, X_4 | X_1)$

$$= \frac{P(X_1, X_2, X_3, X_4)}{P(X_1)}$$

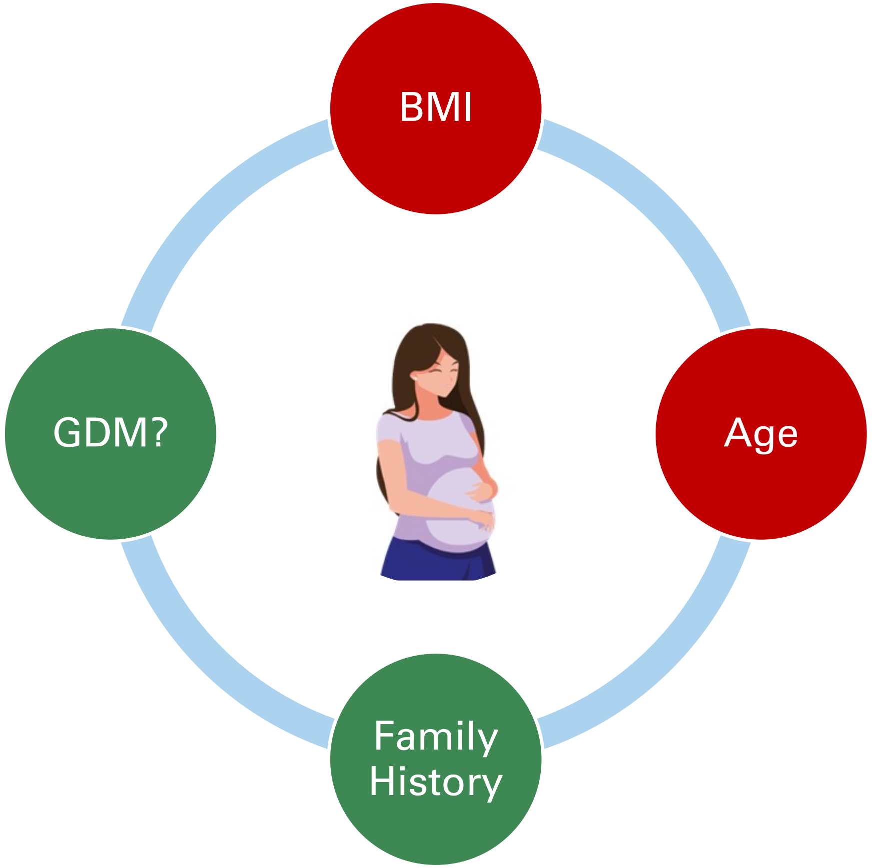
$$= \frac{P(X_1, X_2, X_3, X_4)}{\int P(X_1, X_2, X_3, X_4) dX_2 dX_3 dX_4}$$

Special case of EVI and MAR

- Tractable if MAR is tractable
- Two bottom-up evaluation!
- Linear time!



Tractable Inference Using PCs
Maximum a posteriori Inference (MAP)



Q4: What age group and BMI of women having family history of diabetes is most likely to develop gestational diabetes?

$$\underset{age,bmi}{\arg\max}\ \boldsymbol{P}_M(\text{Age} = age, \text{BMI} = bmi | \text{GDM} = 1, \text{Hist} = 1)$$

$$= \underset{age,bmi}{\arg\max}\ \boldsymbol{P}_M(\text{Age} = age, \text{BMI} = bmi, \text{GDM} = 1, \text{Hist} = 1)$$

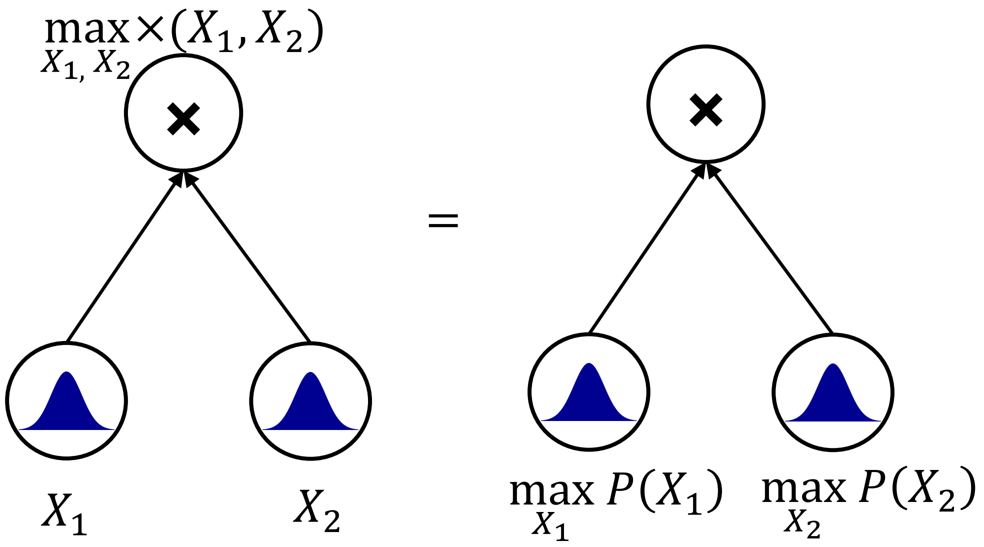
Involves finding maxima of distributions

Tractability for MAP

Suppose we wish to compute $\max_{X_1,X_2,X_3,X_4} P(X_1, X_2, X_3, X_4)$

If **decomposable**, max reduces to max over children of product nodes as

$$\max_{X_1, X_2} P(X_1)(P(X_2) = \max_{X_1} P(X_1) \max_{X_2} P(X_2)$$

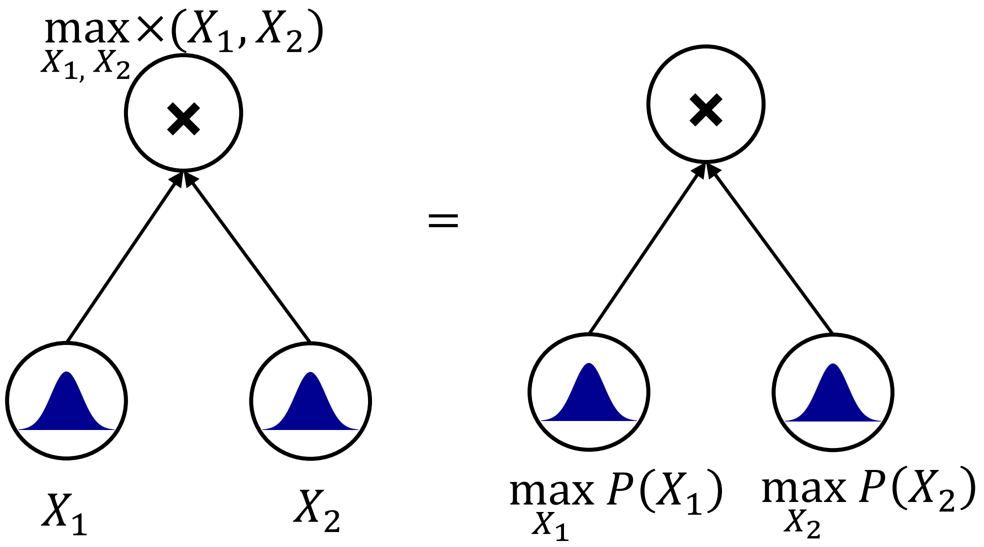


Tractability for MAP

Suppose we wish to compute $\max_{X_1, X_2, X_3, X_4} P(X_1, X_2, X_3, X_4)$

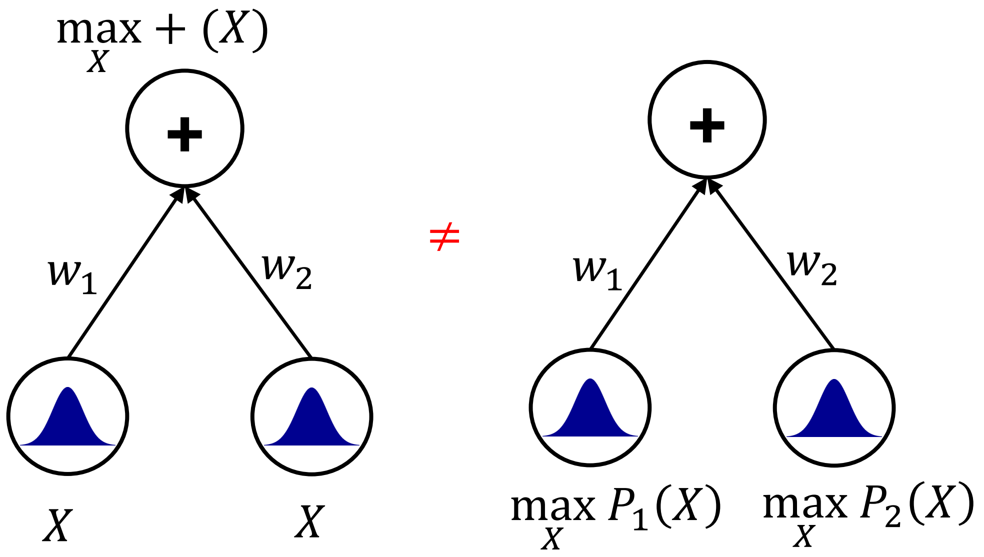
If **decomposable**, max reduces to max over children of product nodes as

$$\max_{X_1, X_2} P(X_1)(P(X_2) = \max_{X_1} P(X_1) \max_{X_2} P(X_2)$$



But cannot push down max over sum nodes!

$$\max_X w_1 P_1(X) + w_2 P_2(X) \neq w_1 \max_X P_1(X) + w_2 \max_X P_2(X)$$

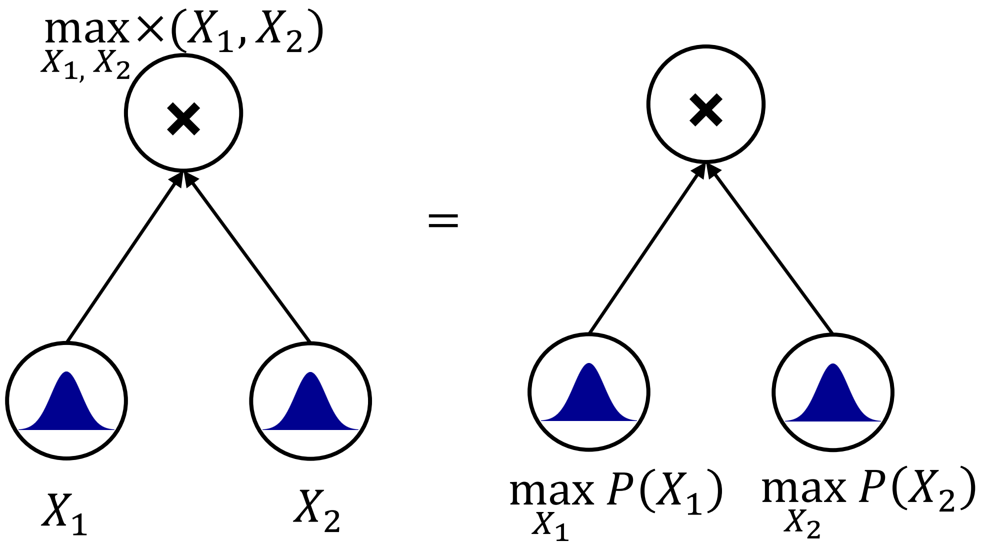


Tractability for MAP

Suppose we wish to compute $\max_{X_1,X_2,X_3,X_4} P(X_1, X_2, X_3, X_4)$

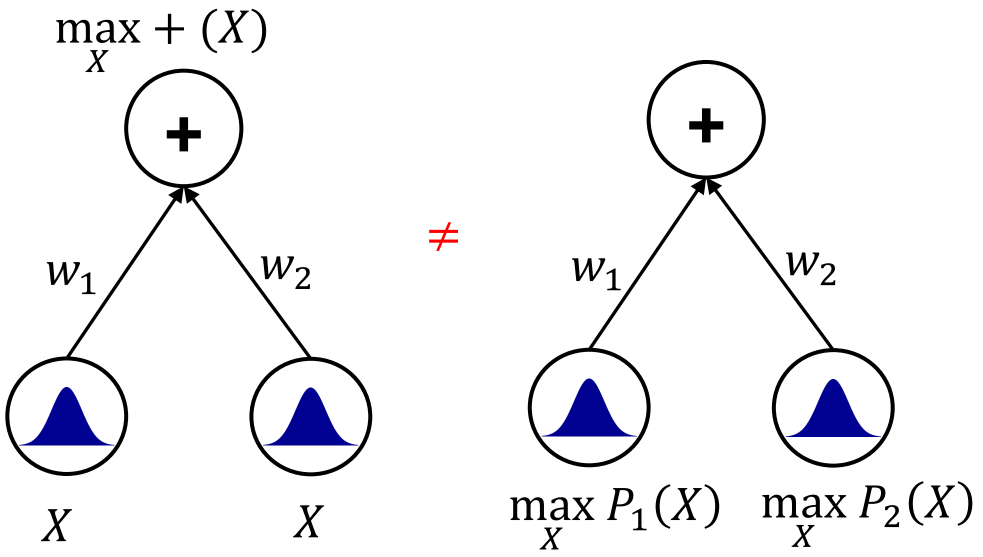
If **decomposable**, max reduces to max over children of product nodes as

$$\max_{X_1, X_2} P(X_1)(P(X_2) = \max_{X_1} P(X_1) \max_{X_2} P(X_2)$$



But cannot push down max over sum nodes!

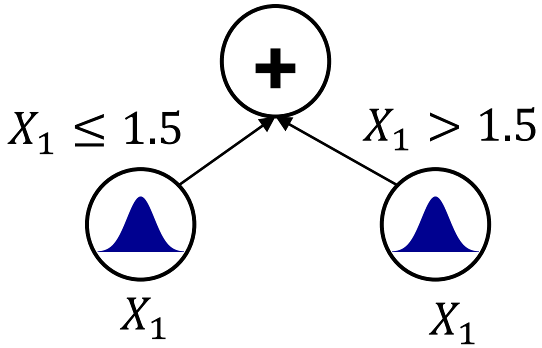
$$\max_X w_1 P_1(X) + w_2 P_2(X) \neq w_1 \max_X P_1(X) + w_2 \max_X P_2(X)$$

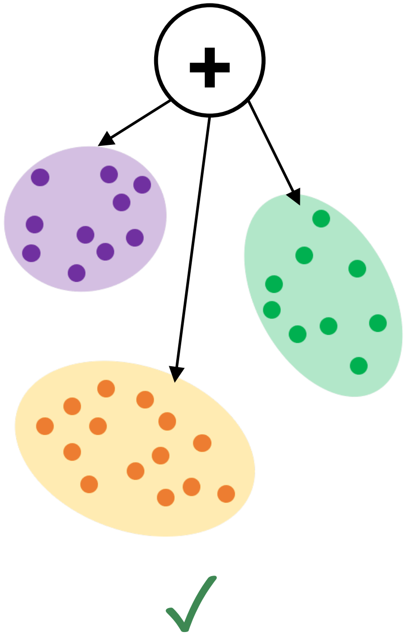


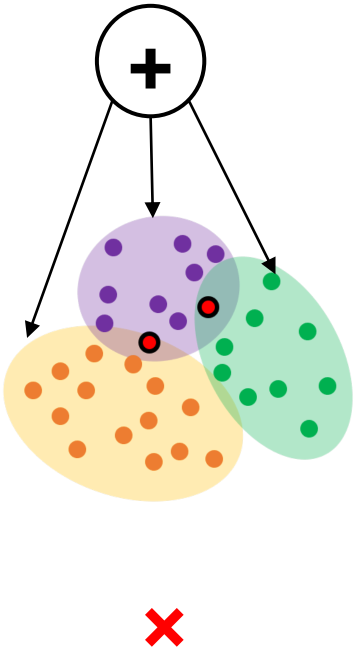
Need more structural properties on the sum nodes!

Structural Properties
(3) Determinism

- A sum node is deterministic if, for any fully-instantiated input, the output of at most one of its children is nonzero.
- A PC is deterministic if all its nodes are deterministic.
- Disjoint mixtures / hard partitions
- For e.g., Decision trees, OR-trees





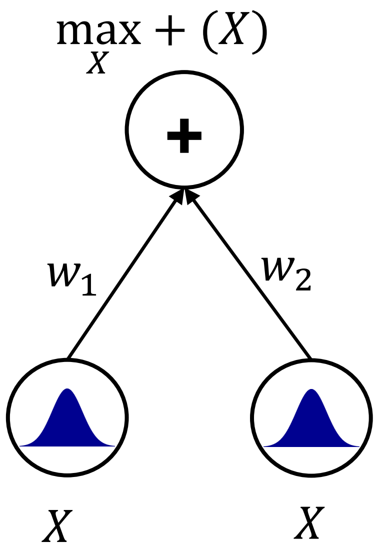


Tractability for MAP

If the sum node is deterministic

- Only one input to the sum node is non-zero
- Thus sum equals a weighted max of inputs
- We can replace + with max operator

$$\max_X \sum_i w_i P_i(X) = \max_X \max_i w_i P_i(X)$$

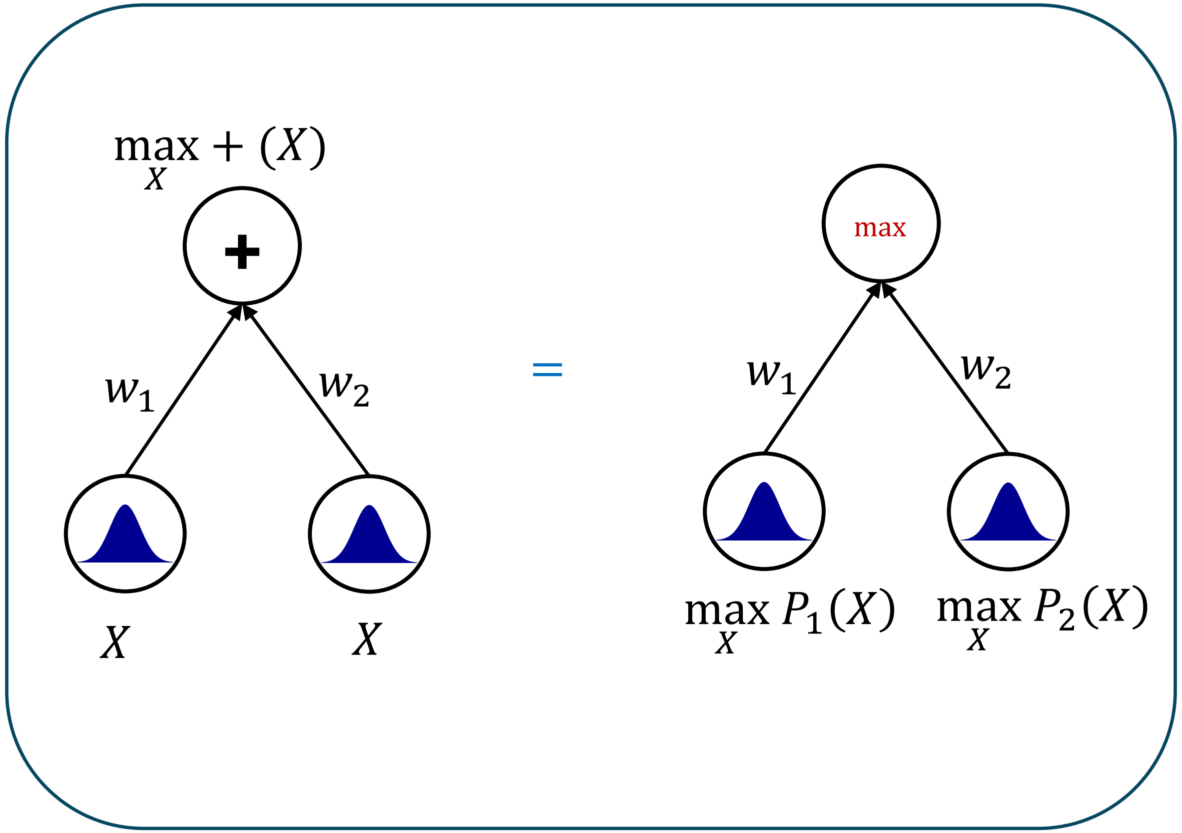


Tractability for MAP

If the sum node is deterministic

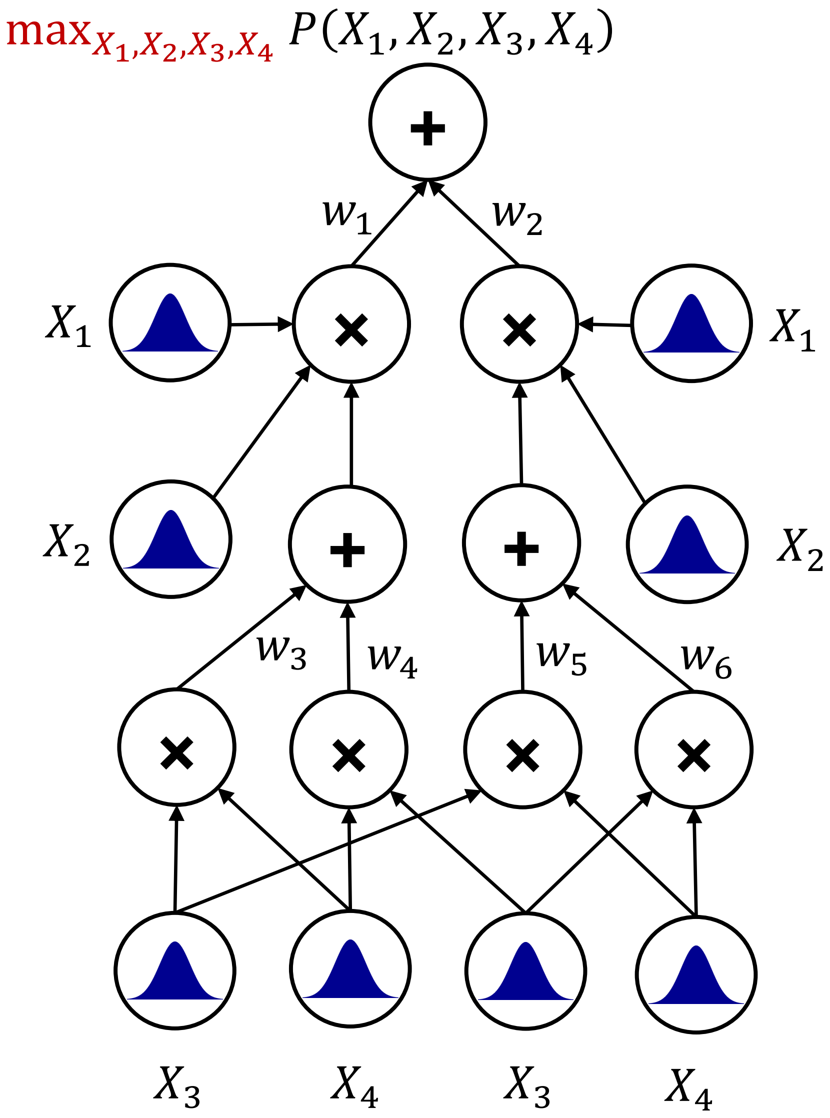
- Only one input to the sum node is non-zero
- Thus sum equals a weighted max of inputs
- We can replace + with max operator

$$\max_X \sum_i w_i P_i(X) = \max_X \max_i w_i P_i(X)$$
$$= \max_i \max_X w_i P_i(X)$$
$$= \max_i \, w_i \, \max_X P_i(X)$$



Can push down max from root to leaves!

Tractability for MAP



Suppose we wish to compute

$$\max_{X_1,X_2,X_3,X_4} P(X_1,X_2,X_3,X_4)$$

If **decomposable**, max reduces to max over children of product nodes as

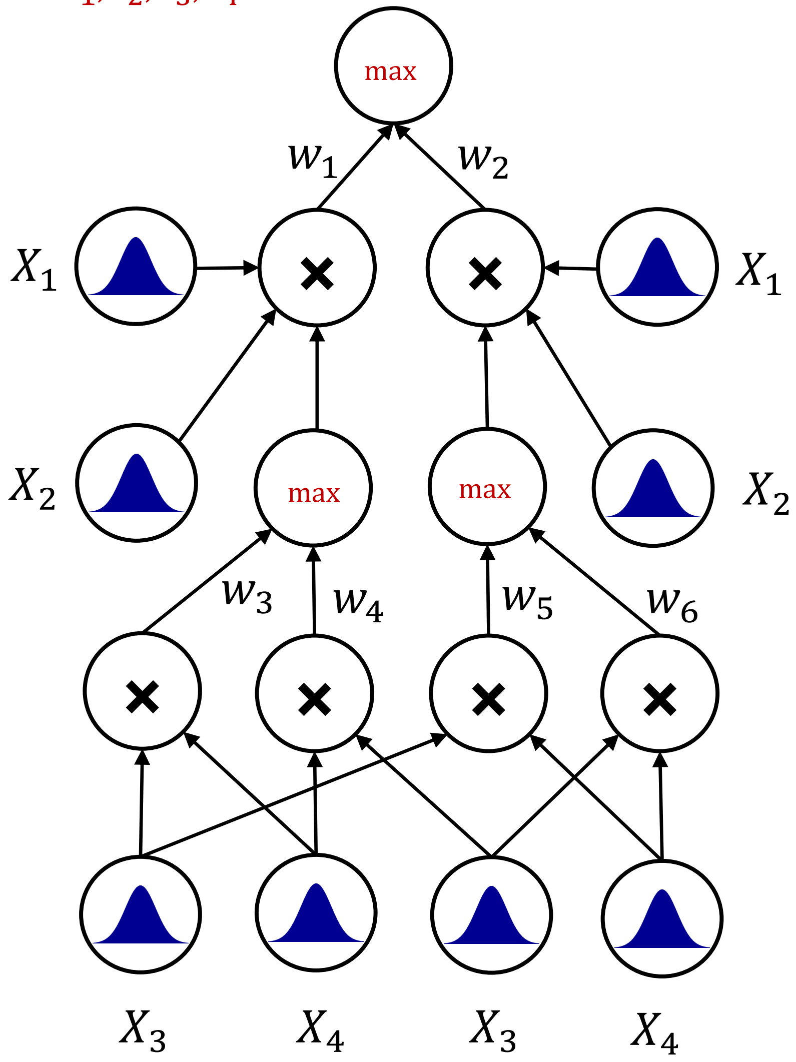
$$\max_{X_1,\,X_2} P(X_1)\,P(X_2) = \max_{X_1} P(X_1)\,\max_{X_2} P(X_2)$$

If **deterministic**,

- Only one input to the sum node is non-zero
- Thus sum equals weighted max of inputs
- We can replace + with max operator

Tractability for MAP

$\max_{X_1,X_2,X_3,X_4} P(X_1, X_2, X_3, X_4)$



Suppose we wish to compute

$$\max_{X_1,X_2,X_3,X_4} P(X_1, X_2, X_3, X_4)$$

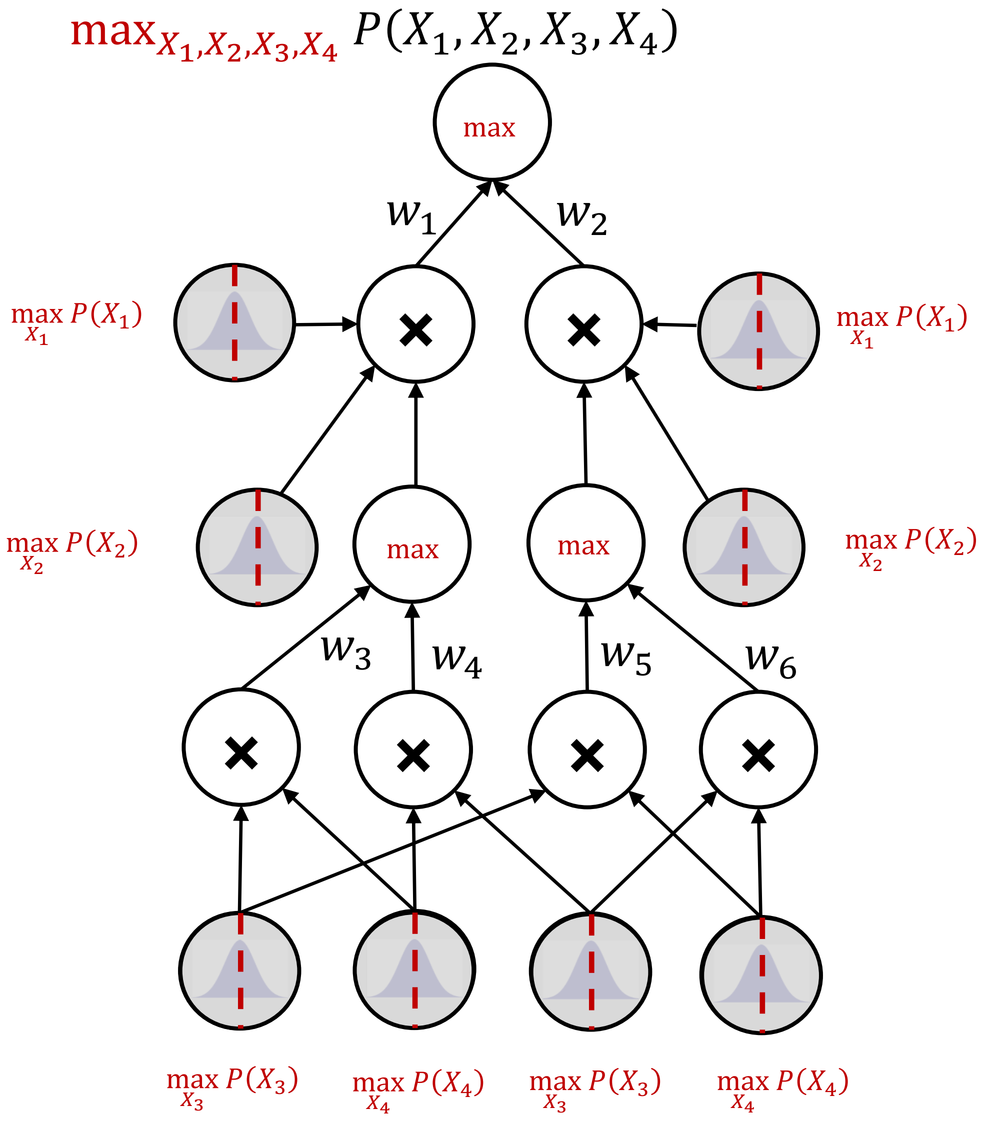
If **decomposable**, max reduces to max over children of product nodes as

$$\max_{X_1, X_2} P(X_1)\, P(X_2) = \max_{X_1} P(X_1) \max_{X_2} P(X_2)$$

If **deterministic**,

- Only one input to the sum node is non-zero
- Thus sum equals weighted max of inputs
- We can replace + with max operator

Tractability for MAP



Suppose we wish to compute

$$\max_{X_1,X_2,X_3,X_4} P(X_1,X_2,X_3,X_4)$$

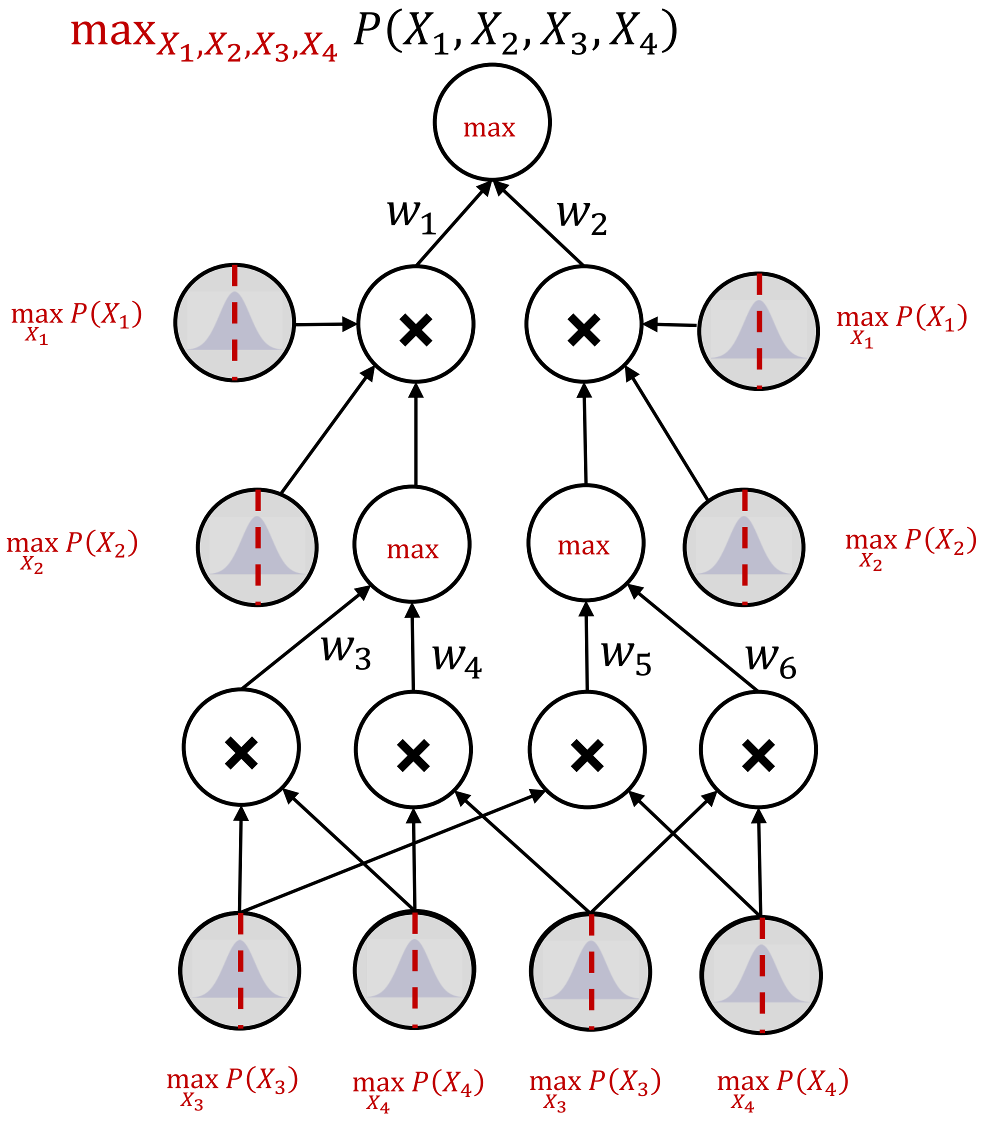
If **decomposable**, max reduces to max over children of product nodes as

$$\max_{X_1, X_2} P(X_1)\, P(X_2) = \max_{X_1} P(X_1) \max_{X_2} P(X_2)$$

If **deterministic**,
- Only one input to the sum node is non-zero
- Thus sum equals the weighted max of inputs
- We can replace + with max operator

Reduces to
- Plugging in modes of leaf distribution
- Evaluating bottom up, replacing + with max

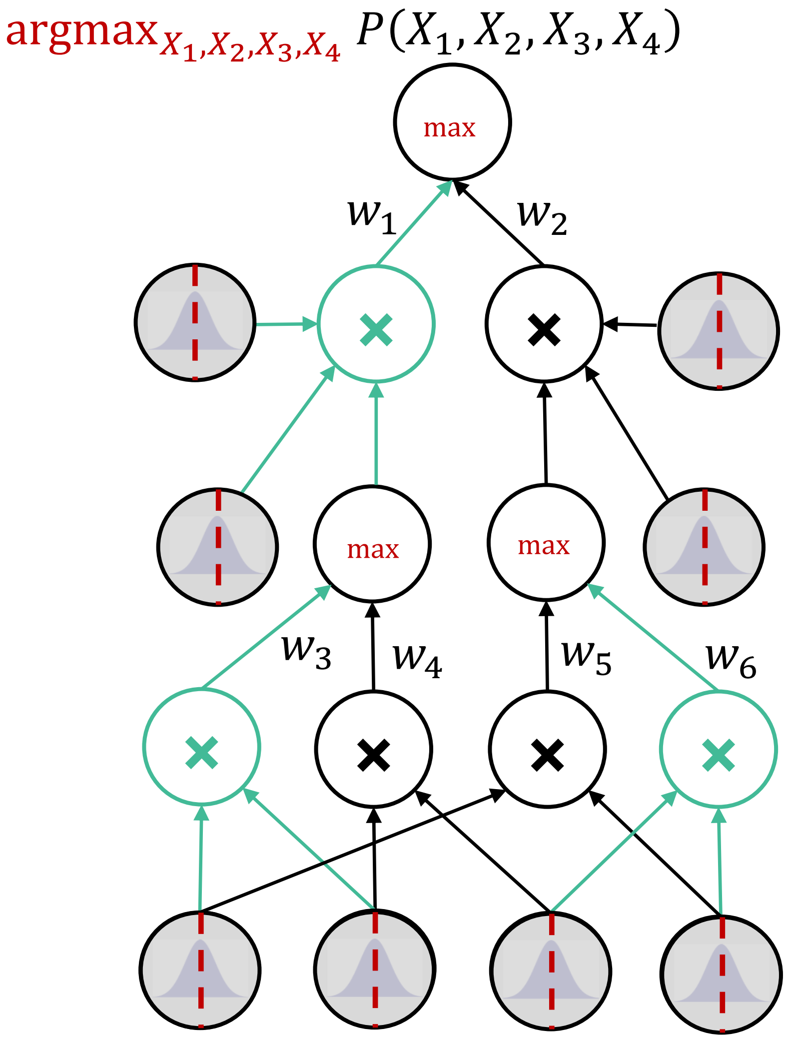
Tractability for MAP



But we are interested in argmax, not max

$$\text{argmax}_{X_1,X_2,X_3,X_4} P(X_1,X_2,X_3,X_4)$$

Tractability for MAP



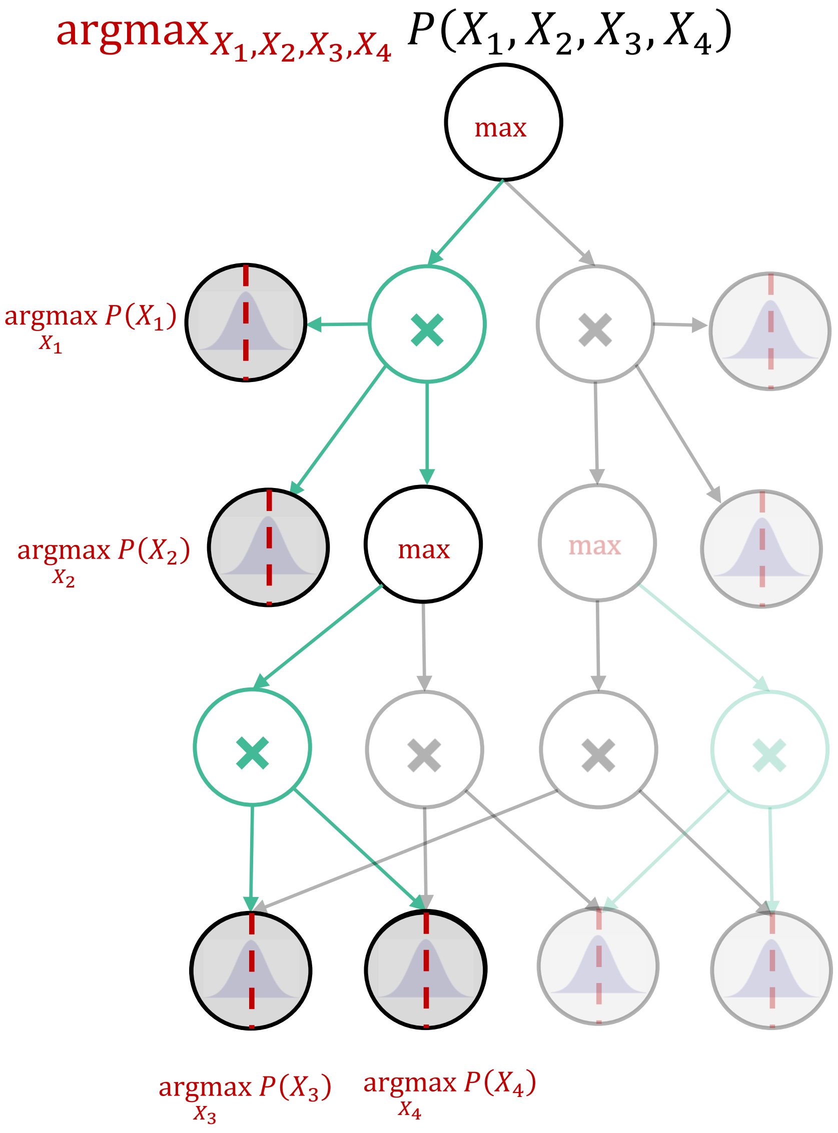
But we are interested in argmax, not max

$$\text{argmax}_{X_1,X_2,X_3,X_4}\ P(X_1,X_2,X_3,X_4)$$

Make **forward pass** (bottom-up) to compute max,

- Keep track of maximizing input to sum

Tractability for MAP



But we are interested in argmax, not max

$$\text{argmax}_{X_1,X_2,X_3,X_4}\ P(X_1, X_2, X_3, X_4)$$

Make **forward pass** (bottom-up) to compute max,

- Keep track of maximizing input to sum

Backtrack from the root to get the maximizing assignments

- Retrieve max activations top-down
- Compute MAP at corresponding leaves

Linear Time!

Tractable Inference via Structural Properties

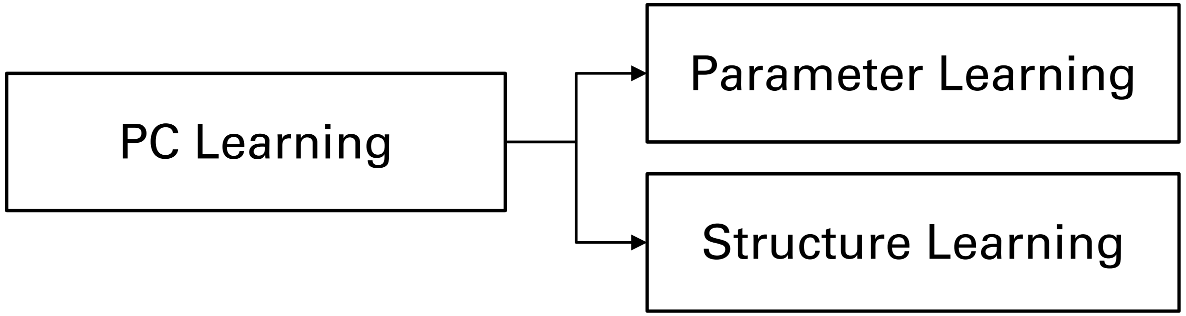
	EVI	MAR	CON	MAP
Smoothness	✓			
+Decomposability	✓	✓	✓	
+Determinism	✓	✓	✓	✓

DTPMs

Probabilistic Circuits

1. Representing distributions using PCs
2. Tractable Inference on PCs
3. **Learning PCs**
4. Expressive Deep Parameterizations

Learning Probabilistic Circuits
Maximum Likelihood



- **What are the parameters of a PC?**
  - Leaf distribution parameters, sum node weights

- **PC enable computing the likelihood of datapoints explicitly**
  - Maximum Likelihood Training!

$$\theta^{\mathsf{MLE}} = \operatorname*{argmax}_{\theta} L(\theta; \mathbf{D}) = \operatorname*{argmax}_{\theta} \prod_{i} p_\theta(\mathbf{d}_i)$$

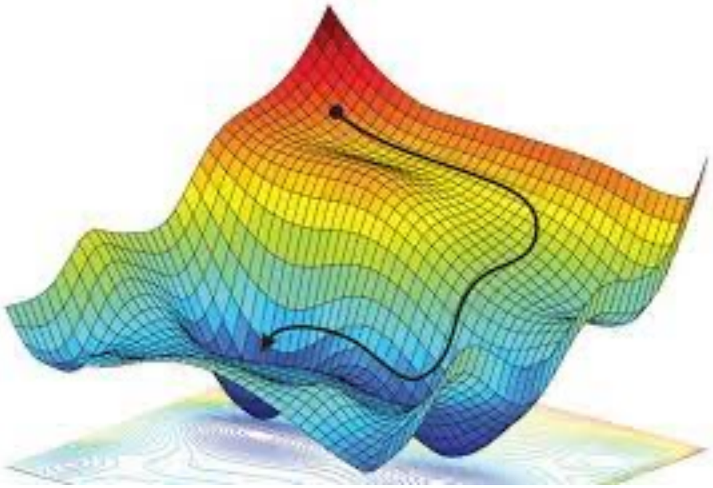
Parameter Learning
Deterministic PCs

- **If Deterministic, we have closed form solution for MLE**
  - Reduces to counting certain sufficient statistics
  - Requires only a single pass on the dataset

Kisa, Doga, et al. "Probabilistic sentential decision diagrams." *Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*. 2014.
Liang, Yitao, and Guy Van den Broeck. "Learning logistic circuits." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. No. 01. 2019.

Parameter Learning
Non-Deterministic PCs – Gradient Descent

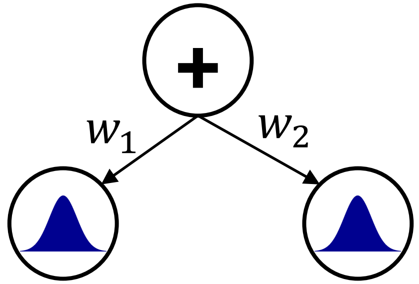
- Deterministic PCs are not very expressive.
- How to learn non-deterministic, but smooth and decomposable PCs?
  - PCs are computational graphs
  - Use backpropagation and gradient descent to train them like neural networks



1. Hoifung Poon, and Pedro Domingos. "Sum-product networks: A new deep architecture." *2011 IEEE International Conference on Computer Vision Workshops*, 2011.
2. Robert Gens, and Pedro Domingos. "Discriminative learning of sum-product networks." *Advances in Neural Information Processing Systems*, 2012
3. Robert Peharz, et al. "Random sum-product networks: A simple and effective approach to probabilistic deep learning." *Uncertainty in Artificial Intelligence*, 2020.

Parameter Learning
Non-Deterministic PCs – Expectation Maximization

- SGD can be slow for PCs
- Can interpret sum nodes as introducing latent variables



  - $+(X) = w_1 P_1(X) + w_2 P_2(X)$
    $= P(Z = 1)P(X|Z = 1) + P(Z = 2)P(X|Z = 2)$

- Maximum likelihood under missing data – Expectation Maximization

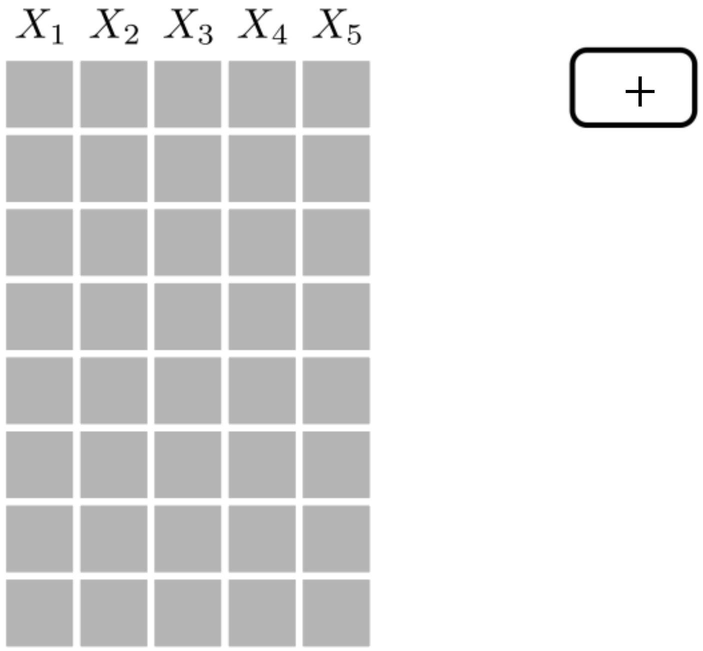
$$\boldsymbol{\theta}^{new} \leftarrow \arg\max_{\boldsymbol{\theta}} \ \mathbb{E}_{p(Z\,|\,\mathbf{X};\boldsymbol{\theta}^{old})}\left[\log p(\mathbf{X}, Z; \boldsymbol{\theta})\right]$$

EM gives better likelihoods faster than GD

Peharz, Robert, et al. "On the latent variable interpretation in sum-product networks." *IEEE transactions on pattern analysis and machine intelligence*, 2016.
Darwiche, Adnan. "A differential approach to inference in Bayesian networks." *Journal of the ACM (JACM)*, 2003.

Structure Learning: Learn-SPN

Learning structure from data via recursive slicing

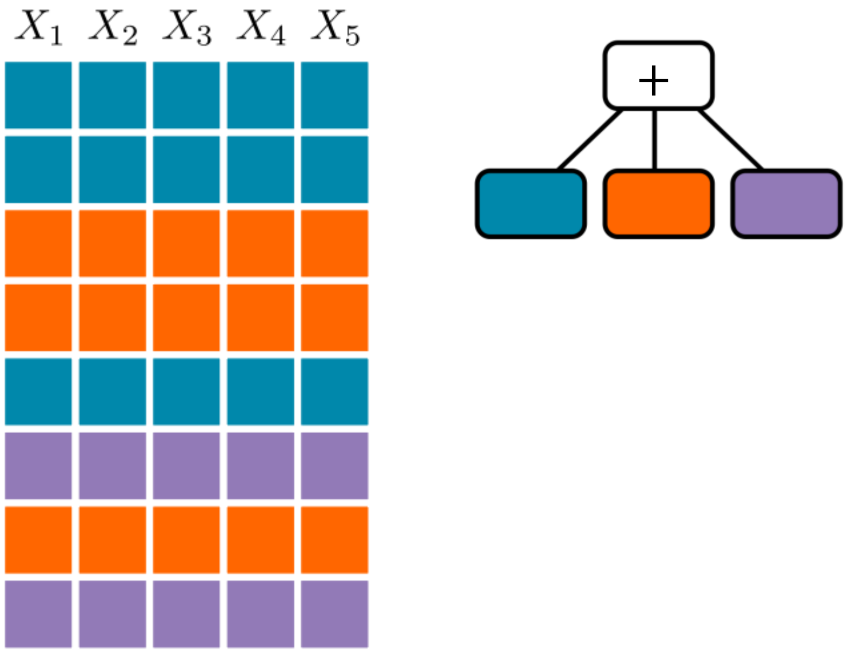


- Start with all variables at the root sum region

Gens, Robert, and Domingos Pedro. "Learning the structure of sum-product networks." *International conference on machine learning*. PMLR, 2013.

Structure Learning: Learn-SPN

Learning structure from data via recursive slicing

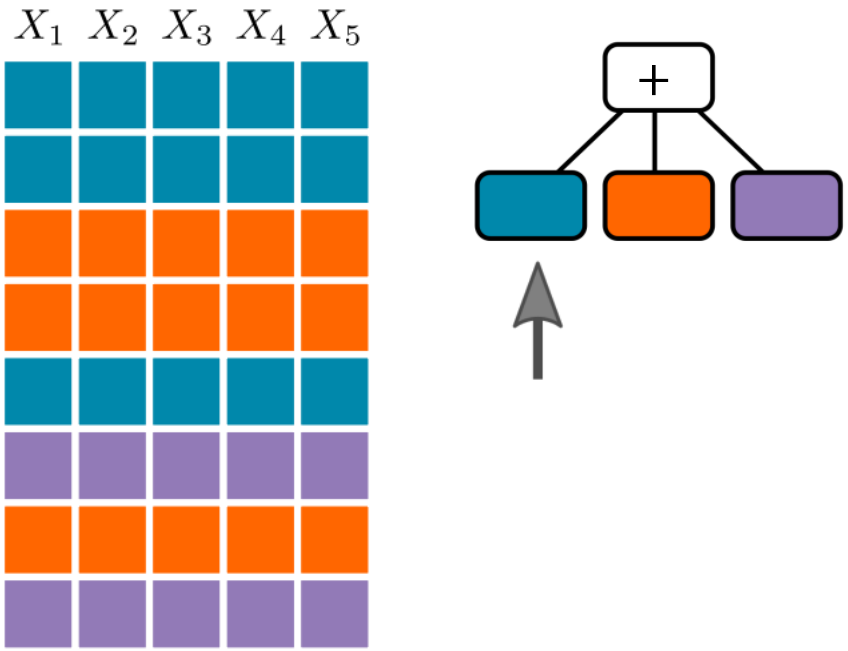


- Start with all variables at the root sum region
- Sum nodes are similar to mixtures
  - Cluster to get child regions!

Gens, Robert, and Domingos Pedro. "Learning the structure of sum-product networks." *International conference on machine learning*. PMLR, 2013.

Structure Learning: Learn-SPN

Learning structure from data via recursive slicing

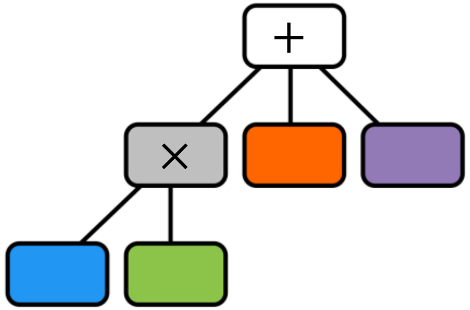


- Start with all variables at the root sum region
- Sum nodes are similar to mixtures
  - Cluster to get child regions!
- Product nodes encode factorizations
  - Try to find independent groups of variables via independence tests

Gens, Robert, and Domingos Pedro. "Learning the structure of sum-product networks." *International conference on machine learning*. PMLR, 2013.

Structure Learning: Learn-SPN

Learning structure from data via recursive slicing

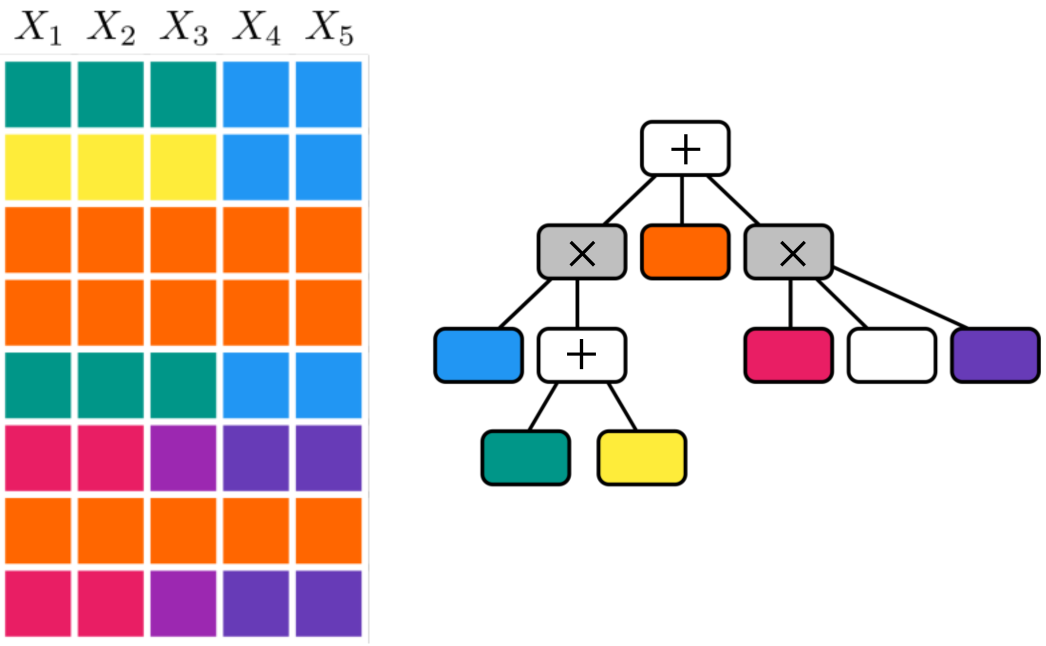


- Start with all variables at the root sum region
- Sum nodes are similar to mixtures
  - Cluster to get child regions!
- Product nodes encode factorizations
  - Try to find independent groups of variables via independence tests
  - Partition into new regions if successful

Gens, Robert, and Domingos Pedro. "Learning the structure of sum-product networks." *International conference on machine learning*. PMLR, 2013.

Structure Learning: Learn-SPN

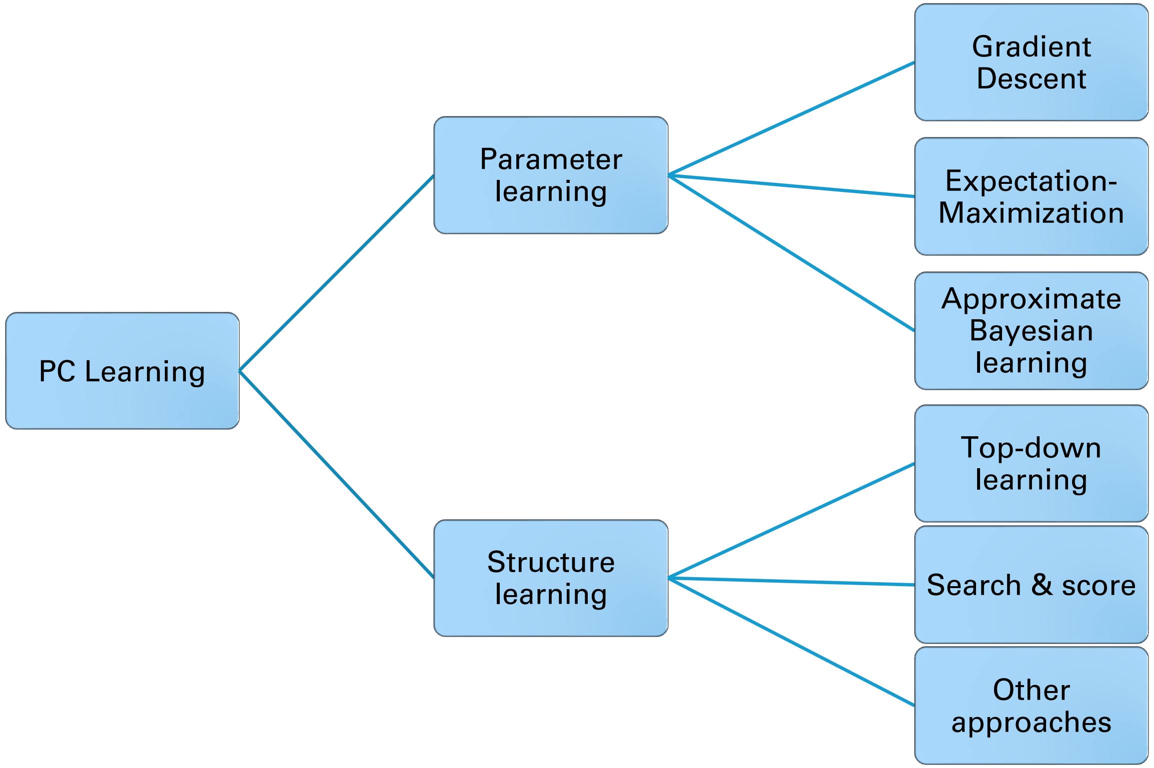
Learning structure from data via recursive slicing



- Start with all variables at the root sum region
- Sum nodes are similar to mixtures
  - Cluster to get child regions!
- Product nodes encode factorizations
  - Try to find independent groups of variables via independence tests
  - Partition into new regions if successful
- Recurse until single variables or stopping criterion reached.

Gens, Robert, and Domingos Pedro. "Learning the structure of sum-product networks." *International conference on machine learning*. PMLR, 2013.

Probabilistic Circuits Learning Methods



SGD, Adam, RMSProp

Concave-convex procedure (CCCP), hard EM

Bayesian moment matching, collapsed variational Bayesian inference (CVB-SPN)

LearnSPN, mixed-SPN learning, SVD-SPN

SearchSPN, oSLRAU

ID-SPN, Bayesian learning of SPNs

Hands On Demo
Building Simple PCs using SPFlow

https://bit.ly/cods-dtpm-1

DTPMs

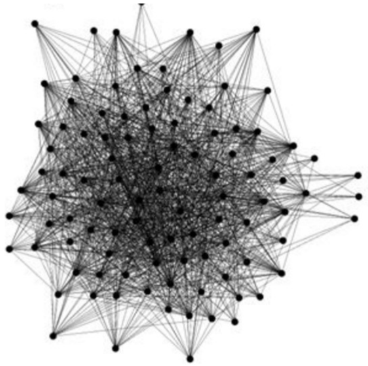
Probabilistic Circuits

1. Representing distributions using PCs
2. Tractable Inference on PCs
3. Learning PCs
4. **Expressive Deep Parameterizations**

Building Denser Computational Graph

Learned structures, while useful, can

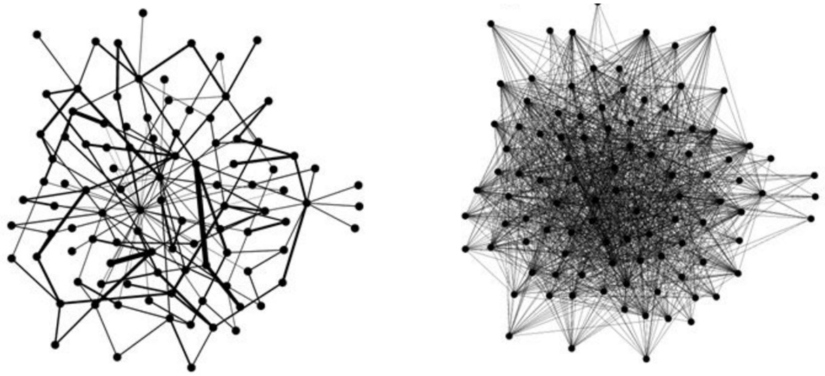
- ❑ Be tedious to tune
- ❑ Lead to <u>Sparse Computation Graphs</u>
    - ❑ Not easily integrated with deep learning frameworks
    - ❑ Not easily deployable on GPUs
    - ❑ Do not scale easily



Building Denser Computational Graph

Learned structures, while useful, can

- Be tedious to tune
- Lead to Sparse Computation Graphs
  - Not easily integrated with deep learning frameworks
  - Not easily deployable on GPUs
  - Do not scale easily

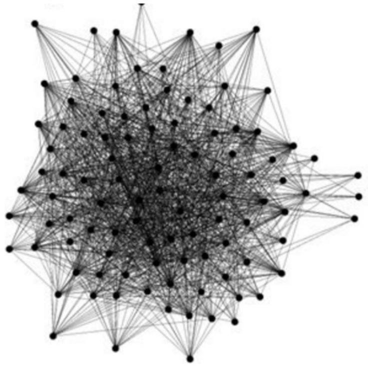


How to build PCs that can utilize deep learning frameworks and scale to millions of parameters ?

Building Denser Computational Graph

Learned structures, while useful, can

- ❑ Be tedious to tune
- ❑ Lead to Sparse Computation Graphs
  - ❑ Not easily integrated with deep learning frameworks
  - ❑ Not easily deployable on GPUs
  - ❑ Do not scale easily



How to build PCs that can utilize deep learning frameworks and scale to millions of parameters ?

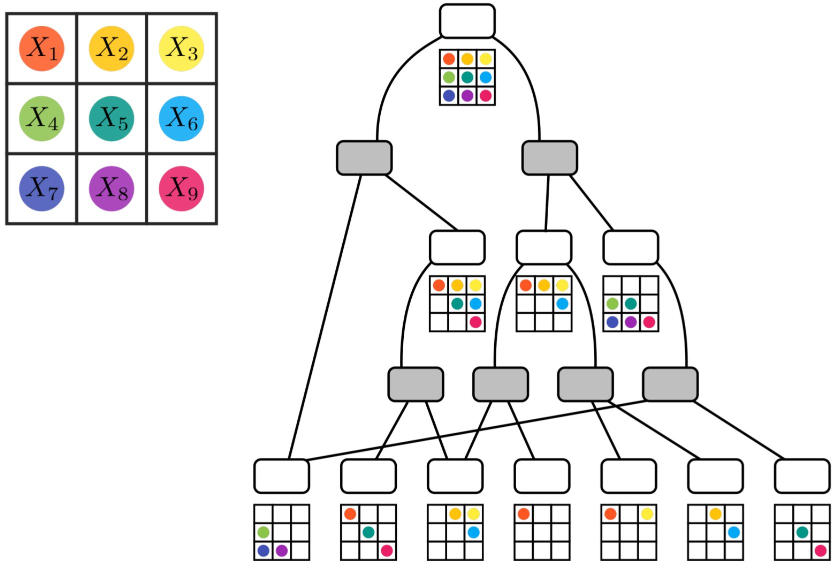
A sufficiently large ensemble of **random structures** can perform as well as a learned structure!

Di Mauro, Nicola, et al. "Fast and accurate density estimation with extremely randomized cutset networks." *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 10*. Springer International Publishing, 2017.

Peharz, Robert, et al. "Random sum-product networks: A simple and effective approach to probabilistic deep learning." *Uncertainty in Artificial Intelligence*. PMLR, 2020

Building Random Homogenous Structured PCs
Region Graphs

- But how do we build random structures that satisfy smoothness and decomposability ?
- Define the variable decompositions using a **Region Graphs**



Region Nodes represents variable sets

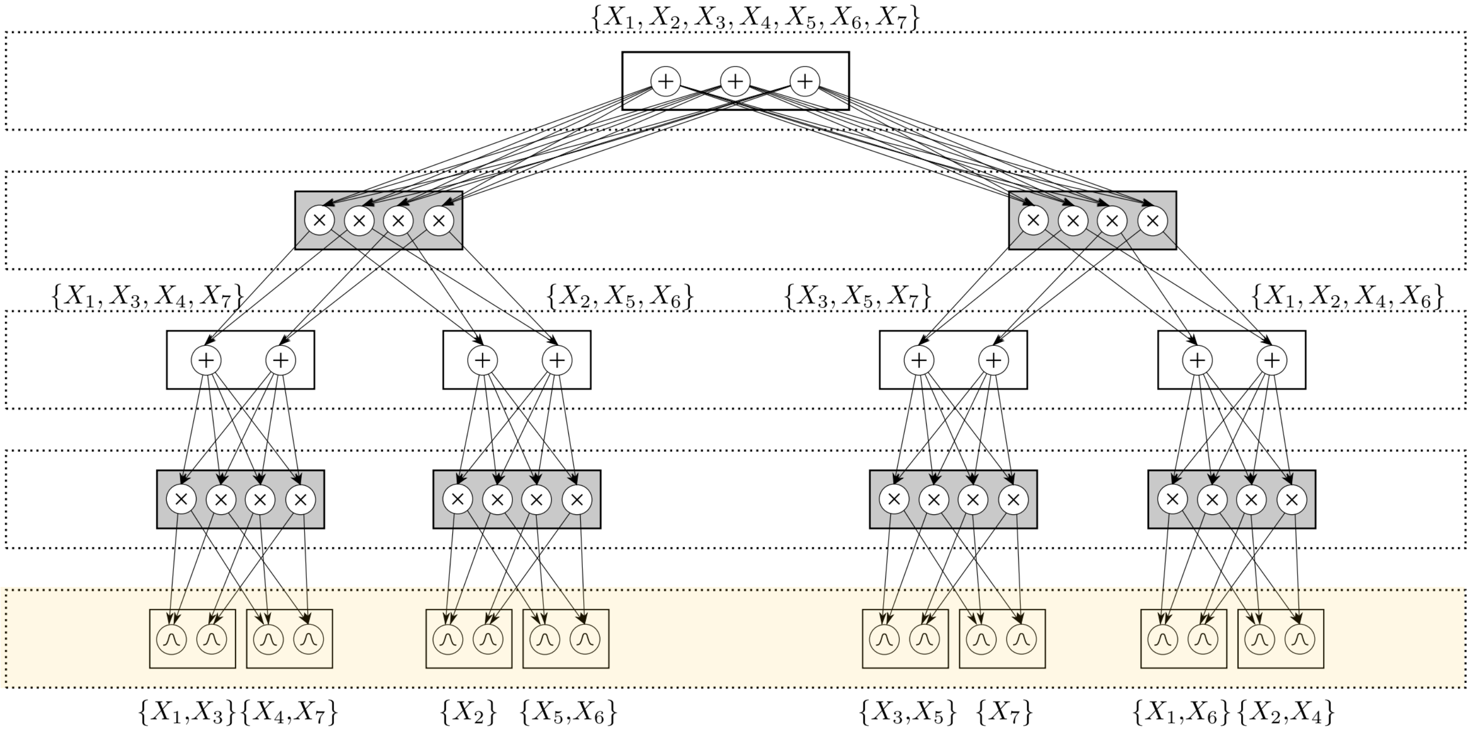
Partition Nodes split regions into disjoint sub-regions

Split Randomly at partition nodes!

Peharz, Robert, et al. "Random sum-product networks: A simple and effective approach to probabilistic deep learning." *Uncertainty in Artificial Intelligence*. PMLR, 2020

From Region Graphs to PCs

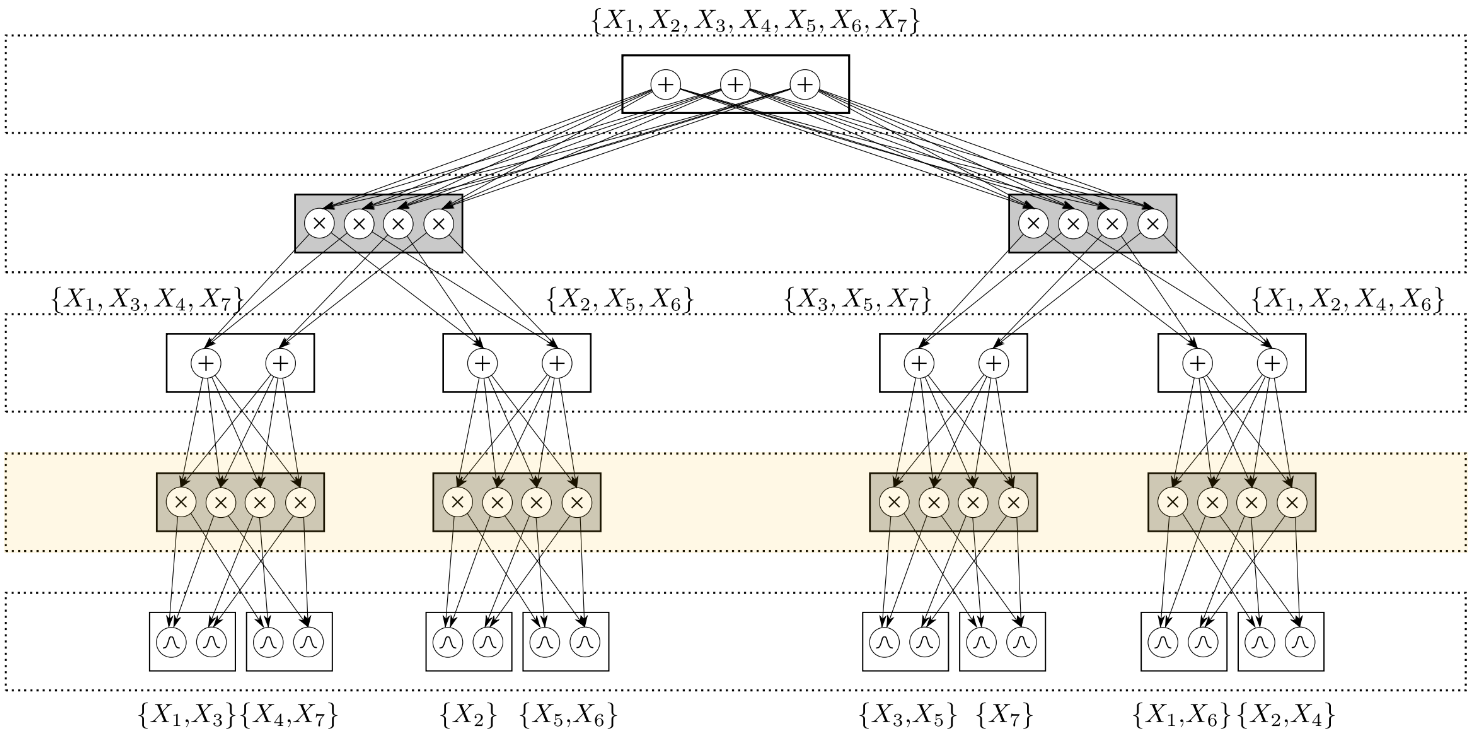
1. Equip leaf regions with vectorized leaf distributions



Peharz, Robert, et al. "Random sum-product networks: A simple and effective approach to probabilistic deep learning." *Uncertainty in Artificial Intelligence*. PMLR, 2020

From Region Graphs to PCs

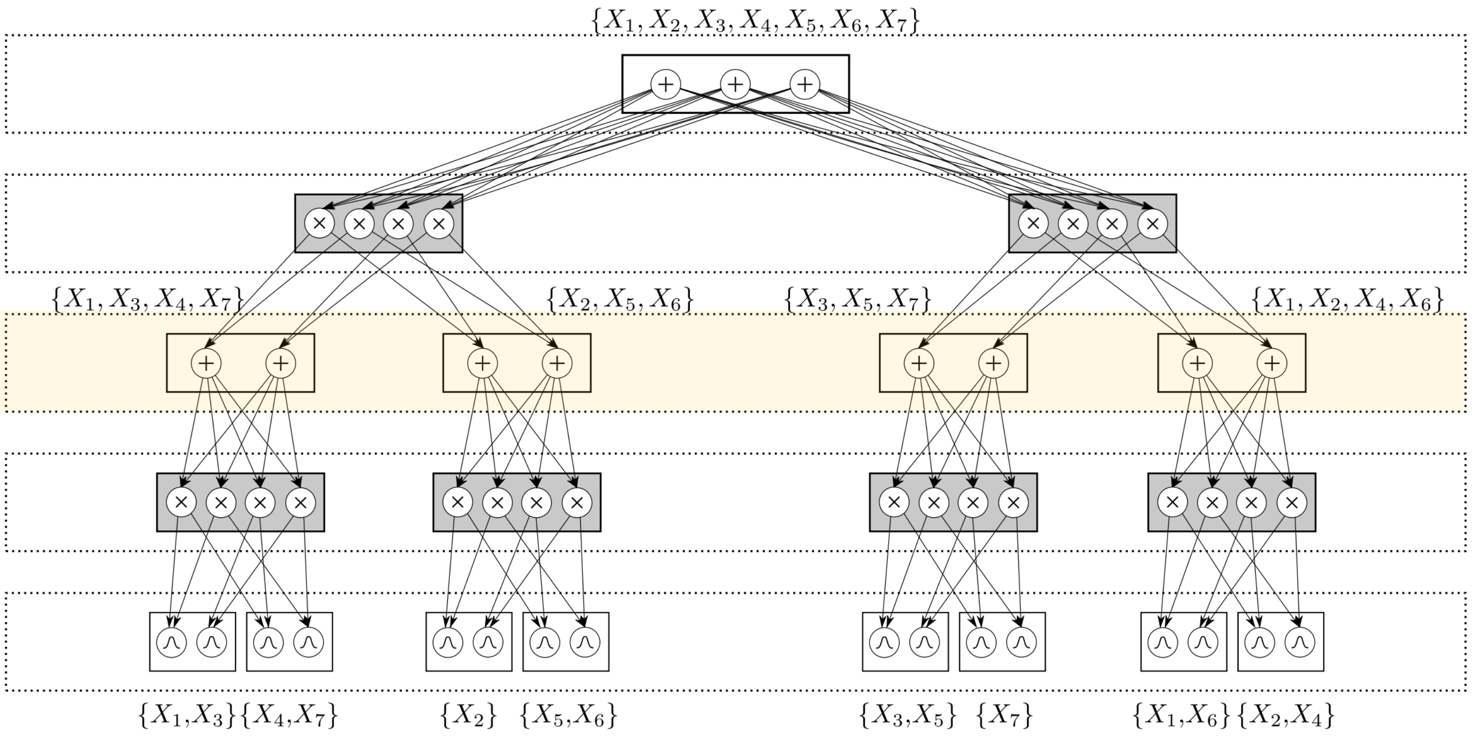
1. Equip leaf regions with vectorized leaf distributions
2. Equip partition nodes with product nodes combining input regions



Peharz, Robert, et al. "Random sum-product networks: A simple and effective approach to probabilistic deep learning." *Uncertainty in Artificial Intelligence*. PMLR, 2020

From Region Graphs to PCs

1. Equip leaf regions with vectorized leaf distributions
2. Equip partition nodes with product nodes combining input regions
3. Equip internal regions with vectorized sum units, connected to all product nodes in its child partition



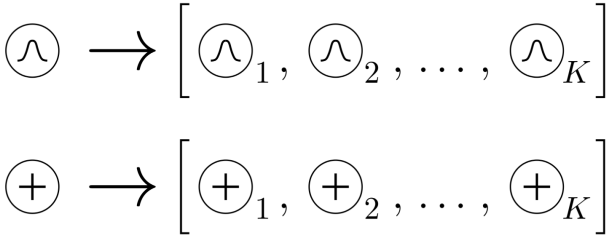
A vectorized PC that is smooth, decomposable, easy to overparameterize and scale on GPUs

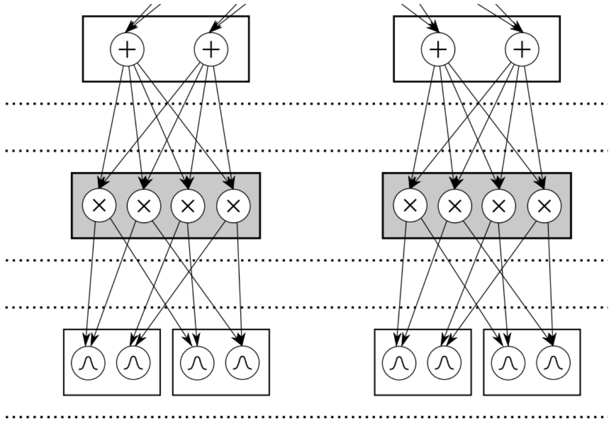
Peharz, Robert, et al. "Random sum-product networks: A simple and effective approach to probabilistic deep learning." *Uncertainty in Artificial Intelligence*. PMLR, 2020

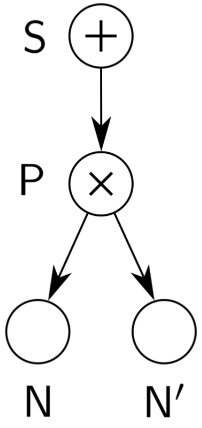
Efficient Monolithic Neural-Network Like Layers
Einsum Networks

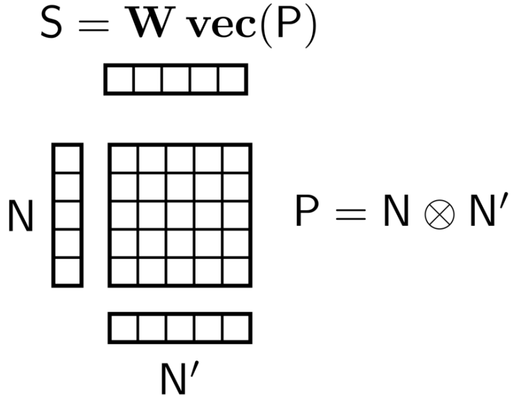
Suppose we equip

- Leaf regions with K vectorized distributions
- Internal regions with K vectorized sum units
- Each sum node is a mixture of K^2 distributions





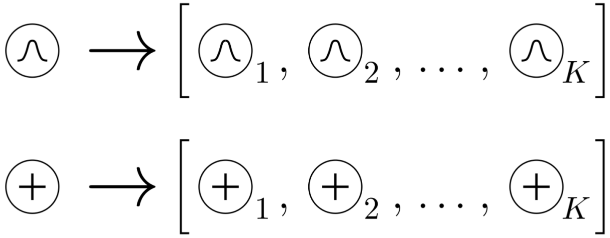


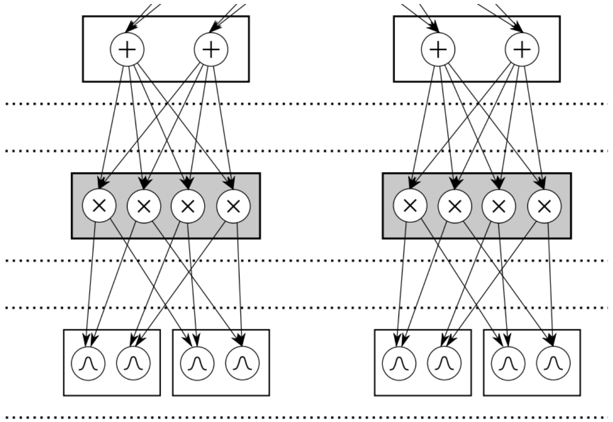


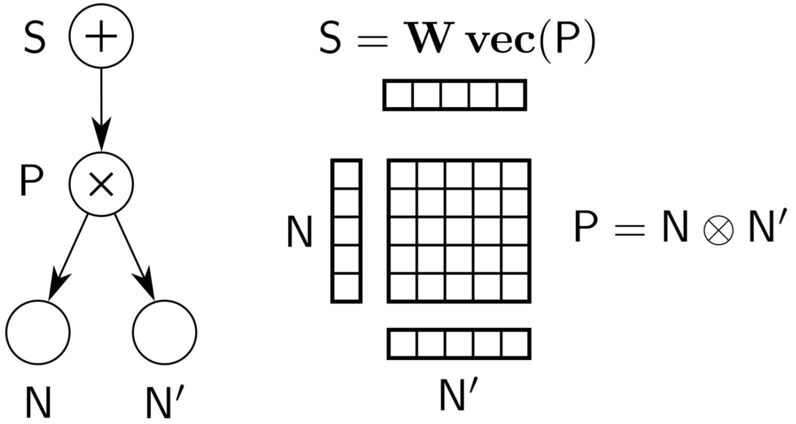
Efficient Monolithic Neural-Network Like Layers
Einsum Networks

Suppose we equip

- Leaf regions with K vectorized distributions
- Internal regions with K vectorized sum units
- Each sum node is a mixture of K^2 distributions



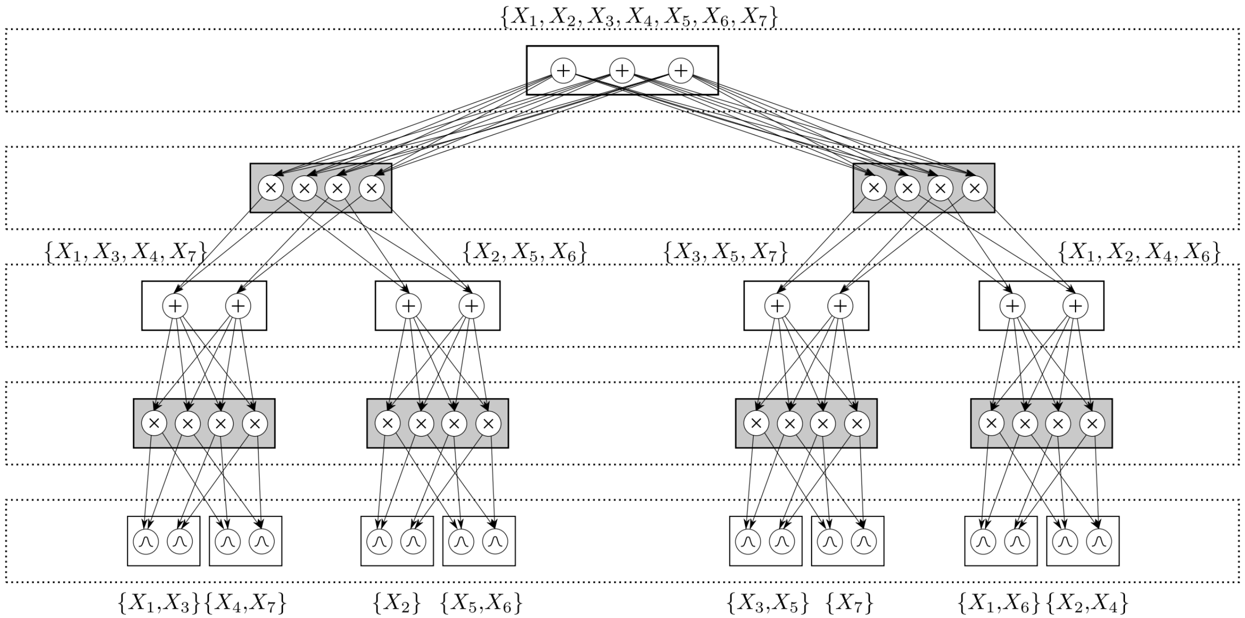


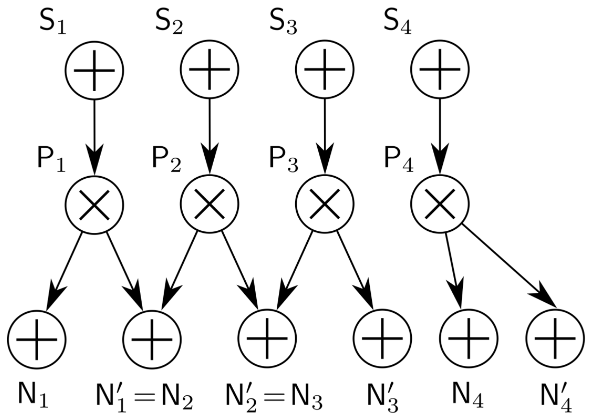


$\mathsf{S}_k = W_{kij}\mathsf{N}_i\mathsf{N}'_j$ single `einsum`-operation

Can combine the sums of products into a single einsum operation with a 3D weight tensor W!

Efficient Monolithic Neural-Network Like Layers
Einsum Networks





$$\mathbf{S}_{lk} = \mathbf{W}_{lkij}\mathbf{N}_{li}\mathbf{N}'_{lj}$$ single `einsum`-operation

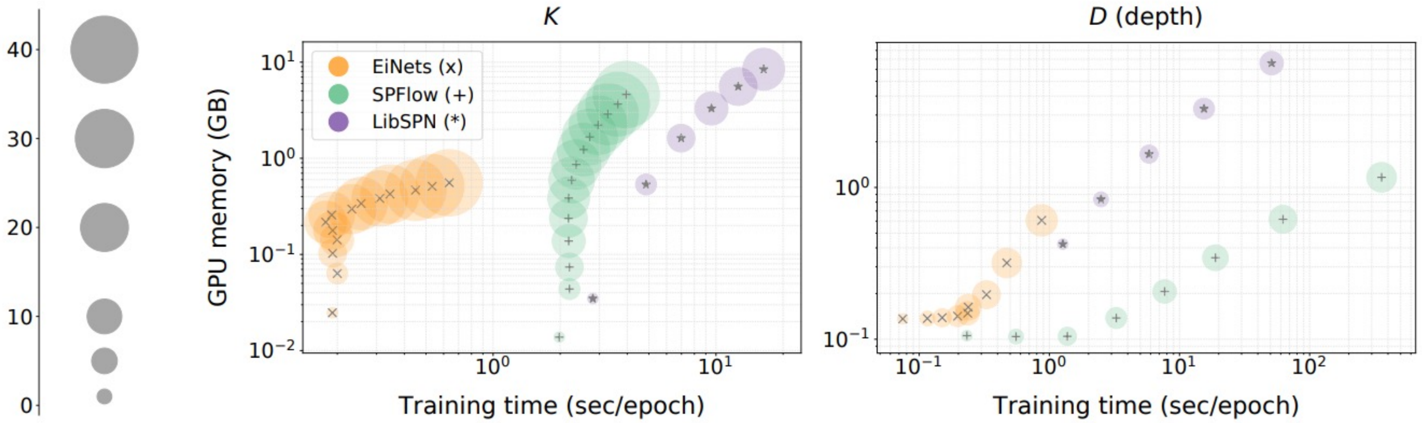
Can combine all such operations at the same depth into an einsum **layer**

- Just an additional dimension for the weight tensor, same einstein notation
- Can topologically arrange into layers using BFS

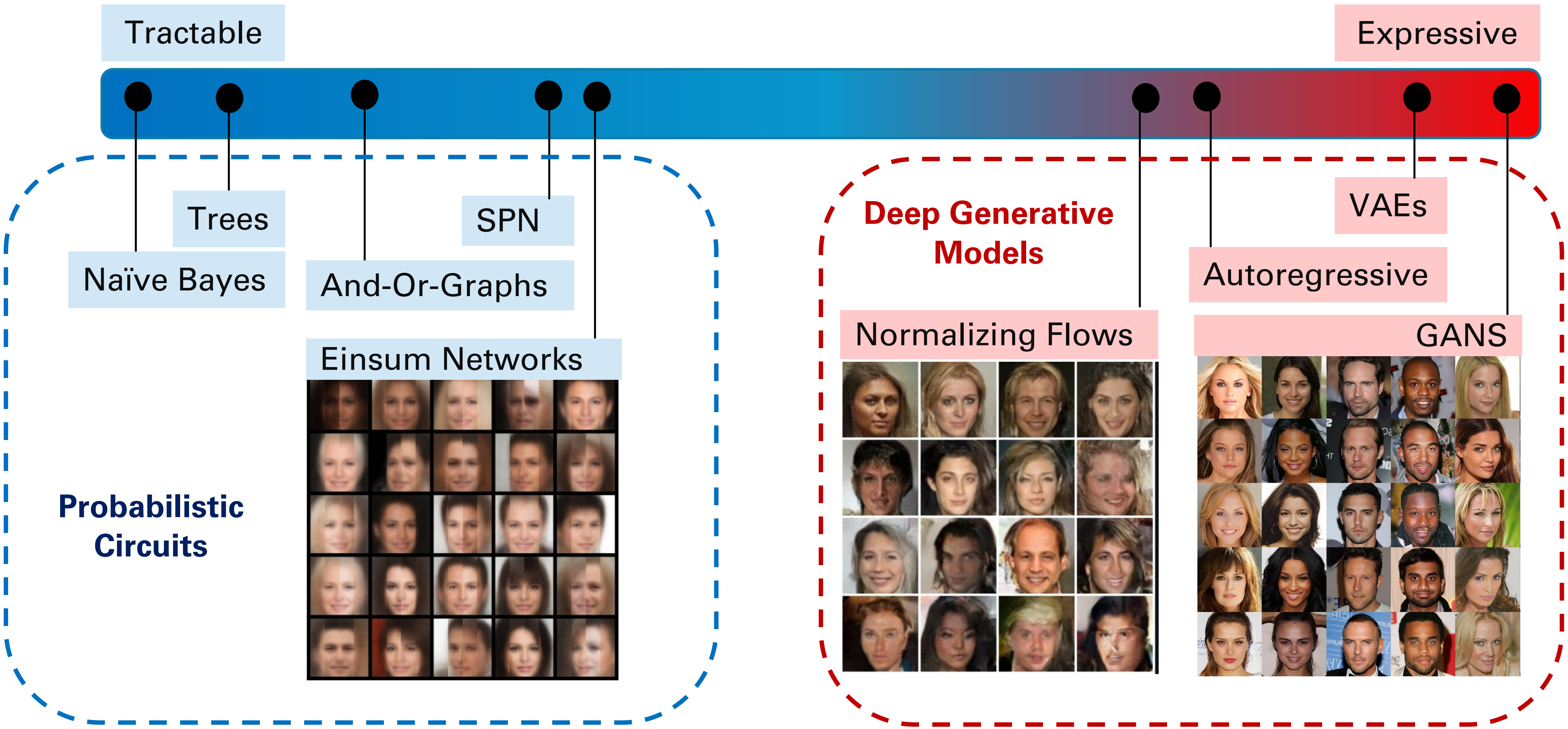
Stack multiple such layers like a neural network!

Efficient Monolithic Neural-Network Like Layers
Einsum Networks

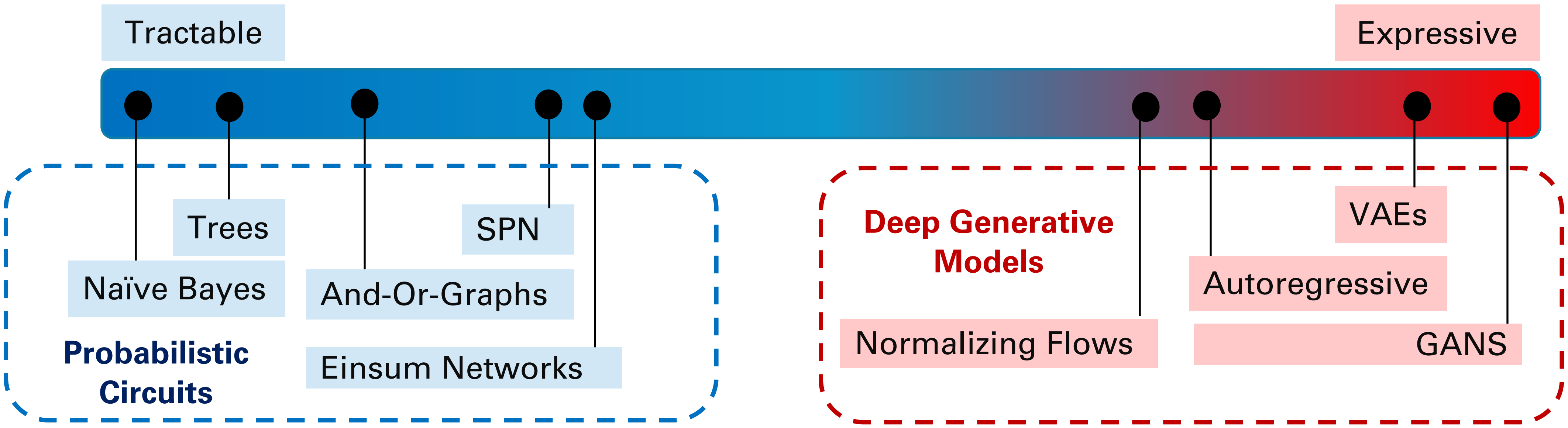
- All operations in a layer can be computed in parallel
- Can also stack multiple replicas of PCs themselves as an additional tensor dimension
    - Efficiently represent ensembles of PCs
- Can run and train PCs with 100s of millions of parameters up to 2 orders of magnitude faster
- Scale PCs to higher dimensional datasets previously not possible



Bridging the Gap between PCs and DGMs



Bridging the Gap between PCs and DGMs

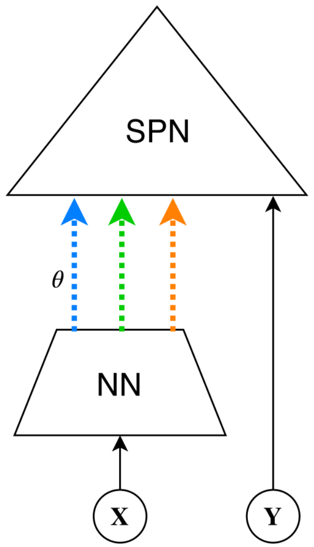


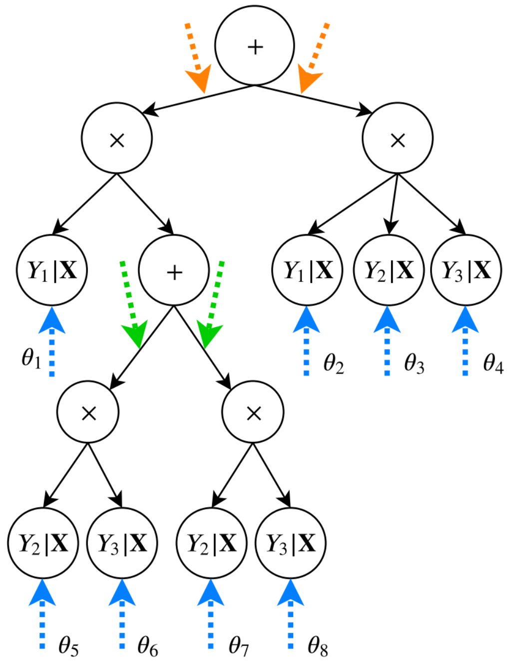
Tractability achieved via:
Efficient encoding of (conditional) **independence** assumptions

Expressivity achieved via:
Utilizing deep **neural** network architectures

Borrowing concepts from both ends of the spectrum to bridge the gap?

Conditional PCs
Using Neural Networks as Gating Modules





Suppose we are only interested in modeling conditional distribution over a set of variables $\mathrm{P}(Y|X)$

- Can use a neural network to transform X into the parameters of the PC that models the distribution over Y
- Tractable inference still possible over variables in Y

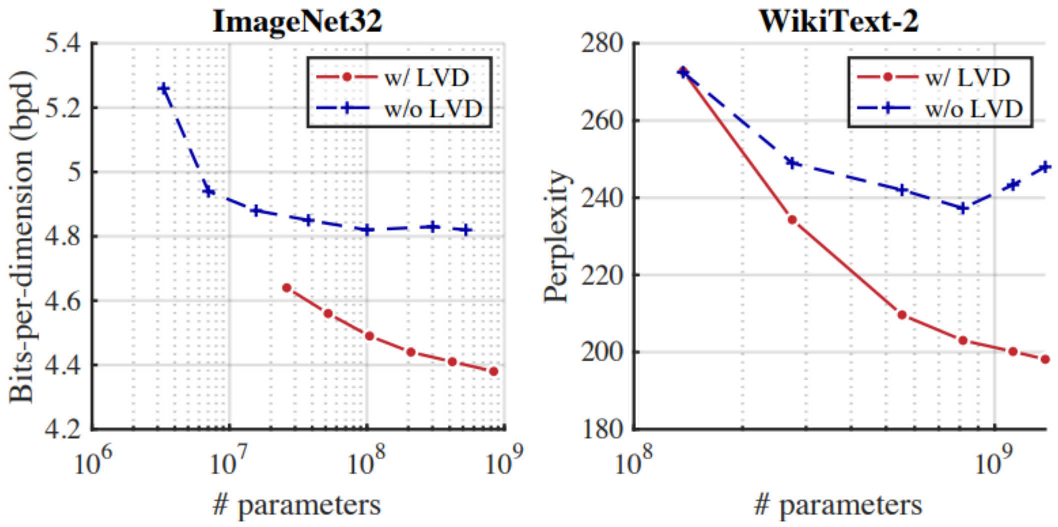
Shao, Xiaoting, et al. "Conditional sum-product networks: Modular probabilistic circuits via gate functions." *International Journal of Approximate Reasoning* 140 (2022):

Latent Variable Distillation using VAEs

Random vectorized parameterization has enabled deep PCs

But overparameterization does not always lead to improvement in performance

Performance plateaus in overparameterized regime



How to make expressivity scale with #parameters?

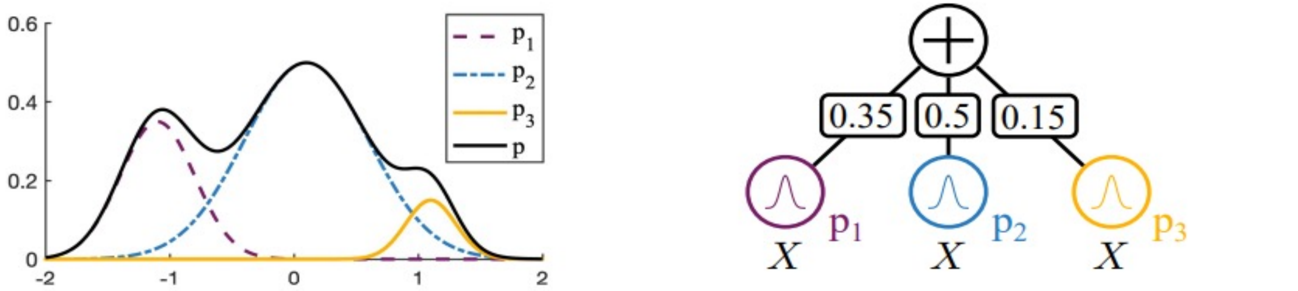
Liu, Anji, Honghua Zhang, and Guy Van den Broeck. "Scaling up probabilistic circuits by latent variable distillation." *International Conference on Learning Representations*, 2022.
Liu, Xuejie, et al. "Understanding the distillation process from deep generative models to tractable probabilistic circuits." *International Conference on Machine Learning*, 2023.

Latent Variable Distillation using VAEs

Issue	PCs can be viewed as latent variable models with a deep hierarchy of latent variables

Sum nodes can be seen as marginalizing out a latent variable Z

$$
\begin{aligned}
+(\mathbf{x}) &= \sum_{N_i \in \text{child}(+)} w_i N_i(\mathbf{x}) \\
&\equiv \sum_{N_i \in \text{child}(+)} P(\mathbf{Z}_+ = i) P(\mathbf{X} = \mathbf{x} \mid \mathbf{Z}_+ = i)
\end{aligned}
$$



(a) A mixture of three Gaussians. (b) A PC that encodes the distribution.

Liu, Anji, Honghua Zhang, and Guy Van den Broeck. "Scaling up probabilistic circuits by latent variable distillation." *International Conference on Learning Representations*, 2022.
Liu, Xuejie, et al. "Understanding the distillation process from deep generative models to tractable probabilistic circuits." *International Conference on Machine Learning*, 2023.

Latent Variable Distillation using VAEs

Issue	PCs can be viewed as latent variable models with a deep hierarchy of latent variables
	As we scale them up, size of their latent space increases
	Renders the landscape of the marginal likelihood over observed variables highly complex.
Solution	Ease the optimization bottleneck by **latent variable distillation - LVD**

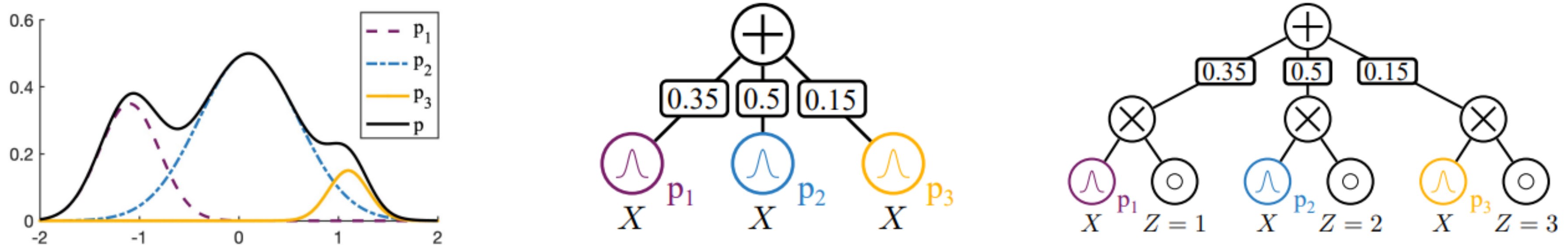
Use less-tractable yet more expressive DGMs → induce semantics-aware assignments to the latent variables of PCs → provide extra supervision to PC optimizers

Liu, Anji, Honghua Zhang, and Guy Van den Broeck. "Scaling up probabilistic circuits by latent variable distillation." *International Conference on Learning Representations*, 2022.
Liu, Xuejie, et al. "Understanding the distillation process from deep generative models to tractable probabilistic circuits." *International Conference on Machine Learning*, 2023.

Latent Variable Distillation using VAEs

LVD Pipeline

1. **Materialize** the latent variables of sum nodes in a PC explicitly



(a) A mixture of three Gaussians. (b) A PC that encodes the distribution. (c) An equivalent deterministic PC.

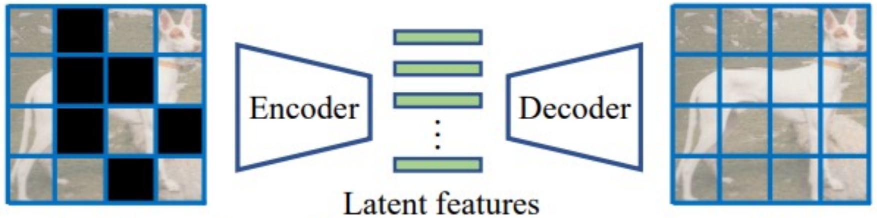
1. Introduce a new latent variable Z_i for each sum node S_i
2. Replace each child N_{ij} of S_i as a **product of $\boldsymbol{N_{ij}}$** and an **indicator variable $\boldsymbol{\delta(Z_i = j)}$**

Latent Variable Distillation using VAEs

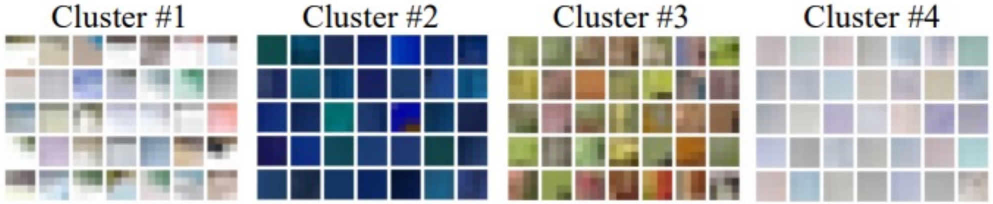
LVD Pipeline

1. **Materialize** the latent variables of sum nodes in a PC explicitly

2. **Induce Assignments** for the materialized variables



(a) Illustration of the Masked Autoencoder model.



(b) Example of image patches belonging to the same cluster.

1. Train a DGM that learns a meaningful latent space, e.g. a VAE
2. Cluster the learned latent space using K-means ($K =$ no. of child nodes for the sum node)

Use the cluster index as the value for Z_i → Get an augmented dataset $\{\boldsymbol{x_i}, \boldsymbol{z_i}\}_{i=1}^{N}$

Latent Variable Distillation using VAEs

LVD Pipeline

1. **Materialize** the latent variables of sum nodes in a PC explicitly → 2. **Induce Assignments** for the materialized variables → 3. Use the induced assignments to better **learn PC parameter**

We can optimize the following lower bound as a proxy for MLE

$$\sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)};\theta) = \sum_{i=1}^{N} \log \sum_{\boldsymbol{z}} p_{\text{aug}}(\boldsymbol{x}^{(i)}, \boldsymbol{z};\theta) \geq \sum_{i=1}^{N} \log p_{\text{aug}}(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)};\theta);$$

Latent Variable Distillation using VAEs

LVD Pipeline

1. **Materialize** the latent variables of sum nodes in a PC explicitly

2. **Induce Assignments** for the materialized variables

3. Use the induced assignments to better **learn PC parameter**

We can optimize the following lower bound as a proxy for MLE

$$\sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)};\theta) = \sum_{i=1}^{N} \log \sum_{\boldsymbol{z}} p_{\text{aug}}(\boldsymbol{x}^{(i)}, \boldsymbol{z};\theta) \geq \sum_{i=1}^{N} \log p_{\text{aug}}(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)};\theta);$$

Train using this objective to get a good initialization for the model

θ^*

Latent Variable Distillation using VAEs

LVD Pipeline

1. **Materialize** the latent variables of sum nodes in a PC explicitly → 2. **Induce Assignments** for the materialized variables → 3. Use the induced assignments to better **learn PC parameter**

We can optimize the following lower bound as a proxy for MLE

$$\sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)};\theta) = \sum_{i=1}^{N} \log \sum_{\boldsymbol{z}} p_{\text{aug}}(\boldsymbol{x}^{(i)}, \boldsymbol{z};\theta) \geq \sum_{i=1}^{N} \log p_{\text{aug}}(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)};\theta);$$

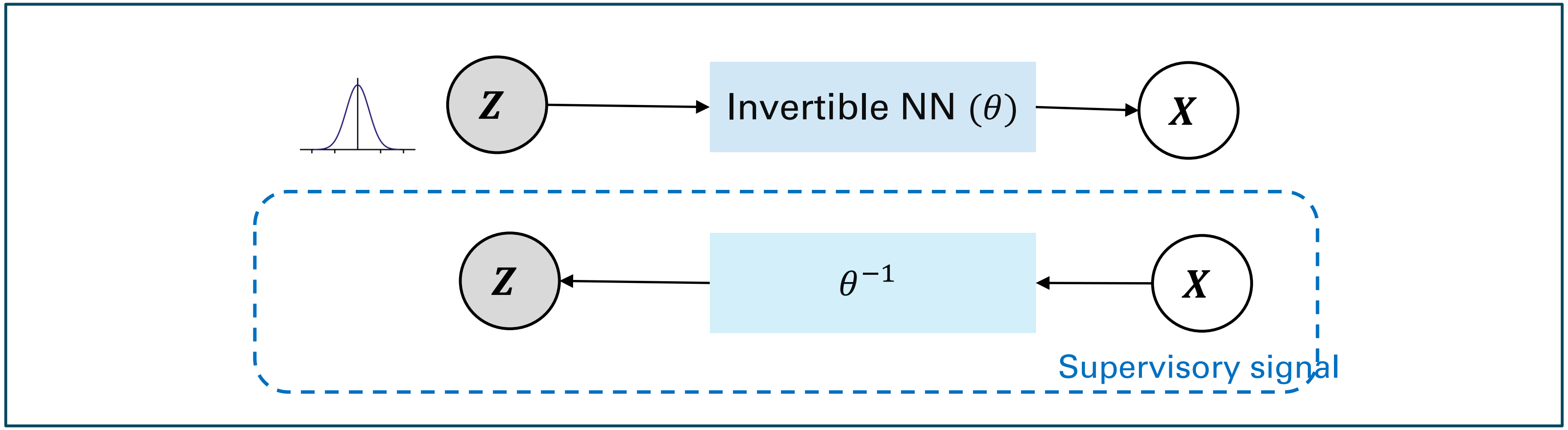
Train using this objective to get a good initialization for the model

θ^*

Finetune on the original MLE objective

Integrating PCs with Normalizing Flows

Normalizing flows use an invertible neural network to map the latent space to the data space



Invertibility allows computing data density **exactly** using the change of variables formula

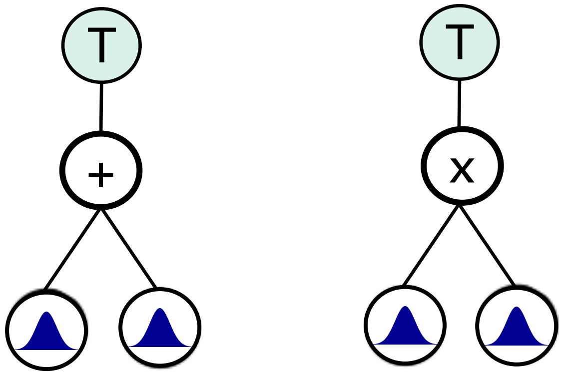
Change of Variables: Z and X be random variables which are related by a mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ such that $X = f(Z)$ and $Z = f^{-1}(X)$. Then

$$p_X(\mathrm{x}) = p_Z(f^{-1}(\mathrm{x})) \left| \det \left(\frac{\partial f^{-1}(\mathrm{x})}{\partial \mathrm{x}} \right) \right|$$

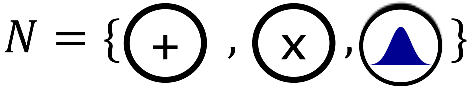
Tractable EVI!

Integrating PCs with Normalizing Flows

Can use the change of variables within PCs using invertible neural transformations?



$$T(N(\boldsymbol{x})) = N(f(\boldsymbol{x}))|\det J_f|$$



Transform nodes

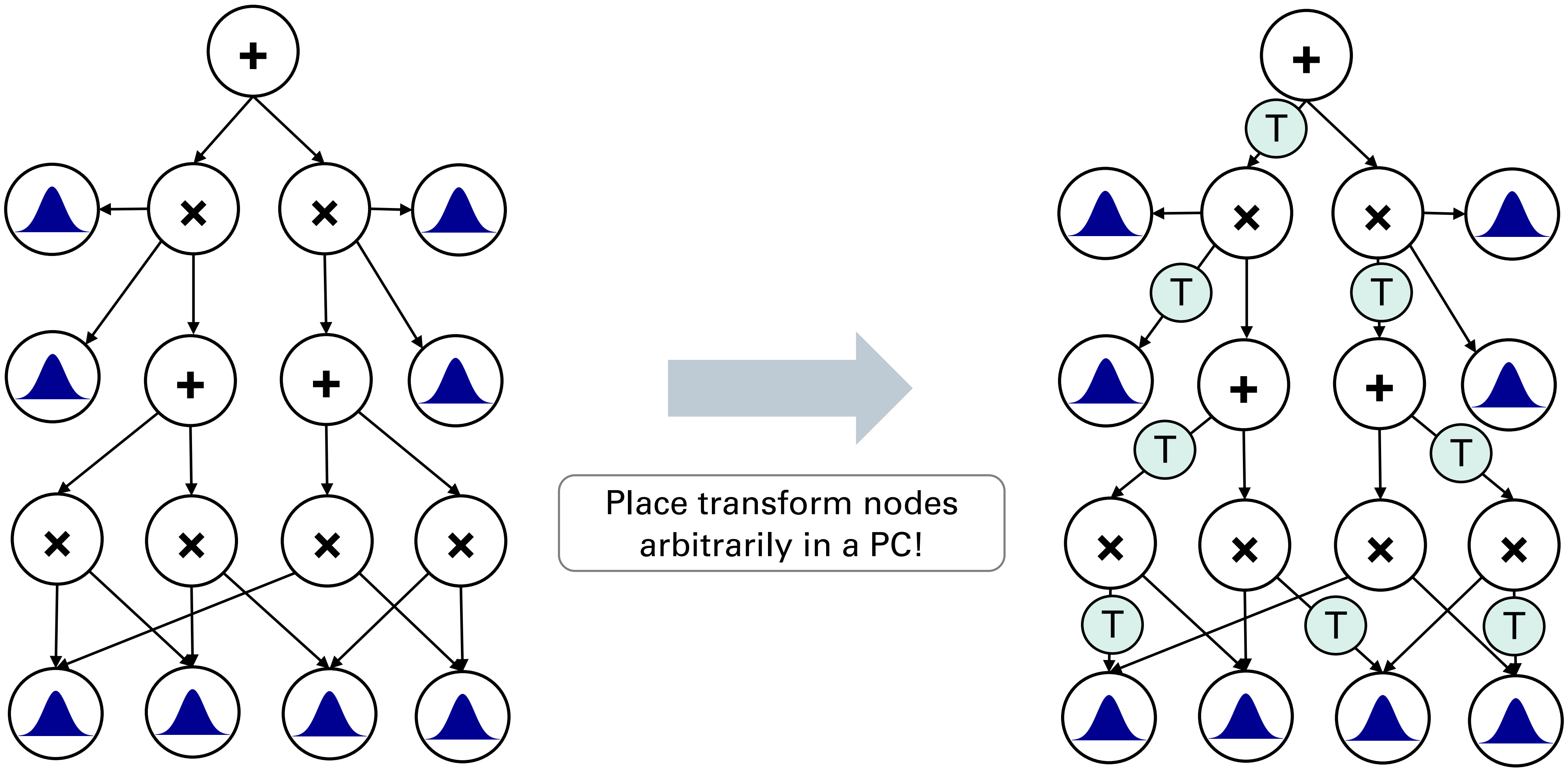
Introduce new **transform nodes** in PCs!

- Represents **normalizing** flows
- Adds expressivity

Pevný, Tomáš, et al. "Sum-product-transform networks: Exploiting symmetries using invertible transformations." *International Conference on Probabilistic Graphical Models*, 2020.

Integrating PCs with Normalizing Flows

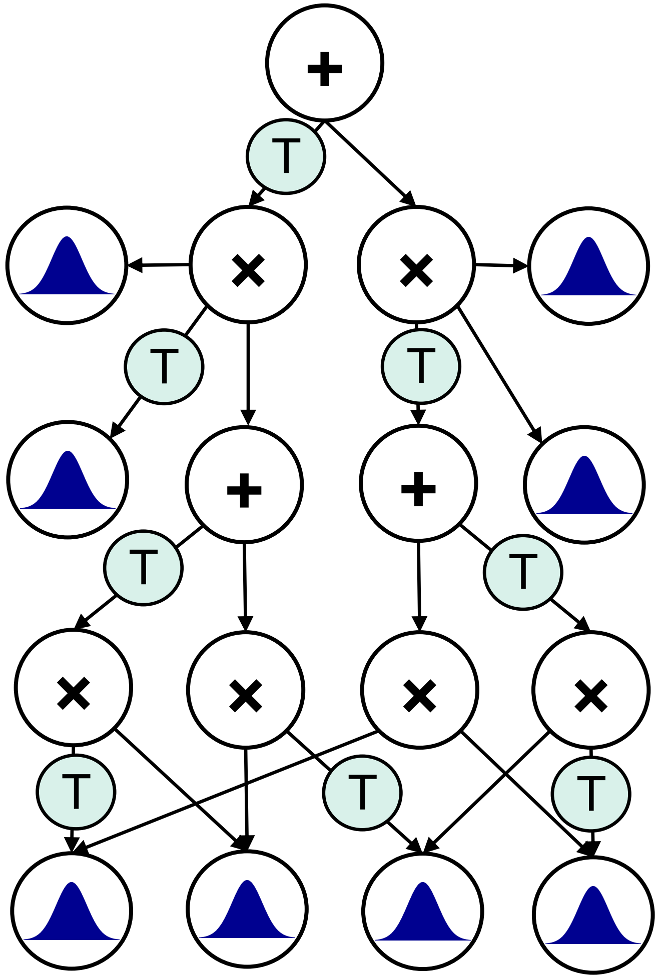
Introduce new **transform nodes** in PCs



Pevný, Tomáš, et al. "Sum-product-transform networks: Exploiting symmetries using invertible transformations." *International Conference on Probabilistic Graphical Models*, 2020.

Integrating PCs with Normalizing Flows

Place transform nodes arbitrarily in a PC!



- But this violates decomposability of the PC
- Cannot push down integrals over transform nodes
- Not a tractable model for MAR, CON, MAP
- Need more structural properties on transform nodes

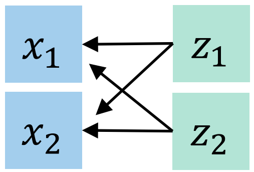
Sahil Sidheekh, Kristian Kersting, and Sriraam Natarajan. "Probabilistic flow circuits: towards unified deep models for tractable probabilistic inference." *Uncertainty in Artificial Intelligence*, 2023

Integrating PCs with Normalizing Flows

Defining Structural Properties for Transform Nodes

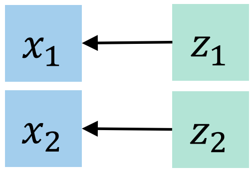
τ –Decomposability

When defined over a product node, f needs to transform the variables involved in the scope of its children **independently**



Not τ-Decomposable

$\boldsymbol{z} = f(\boldsymbol{x})$



τ-Decomposable

$\boldsymbol{z} = [z_1, z_2] = [f_1(x_1), f_2(x_2)]$

A necessary condition for tractability of MAR, CON and MAP

Sahil Sidheekh, Kristian Kersting, and Sriraam Natarajan. "Probabilistic flow circuits: towards unified deep models for tractable probabilistic inference." *Uncertainty in Artificial Intelligence*, 2023

Integrating PCs with Normalizing Flows

Defining Structural Properties for Transform Nodes

τ –Decomposability	When defined over a product node, f needs to transform the variables involved in the scope of its children **independently**

A necessary condition for tractability of MAR, CON and MAP

Integrals on **τ-Decomposable** Transform Nodes can be pushed down

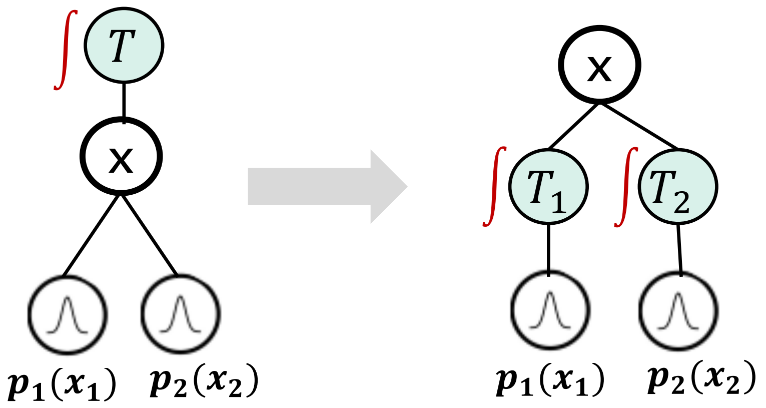
$$\int p(\boldsymbol{x_1}, \boldsymbol{x_2})\, d\boldsymbol{x} \qquad (d\boldsymbol{x} = d\boldsymbol{x_1} d\boldsymbol{x_2})$$

$$= \int T(\times(\boldsymbol{x_1}, \boldsymbol{x_2}))\, d\boldsymbol{x}$$

$$= \int \times(\boldsymbol{f}(\boldsymbol{x_1}, \boldsymbol{x_2}))|\det J_f|\ d\boldsymbol{x}$$

$$= \int p_1(f(\boldsymbol{x_1}, \boldsymbol{x_2}))p_2(f(\boldsymbol{x_1}, \boldsymbol{x_2}))|\det J_f|\ d\boldsymbol{x}$$

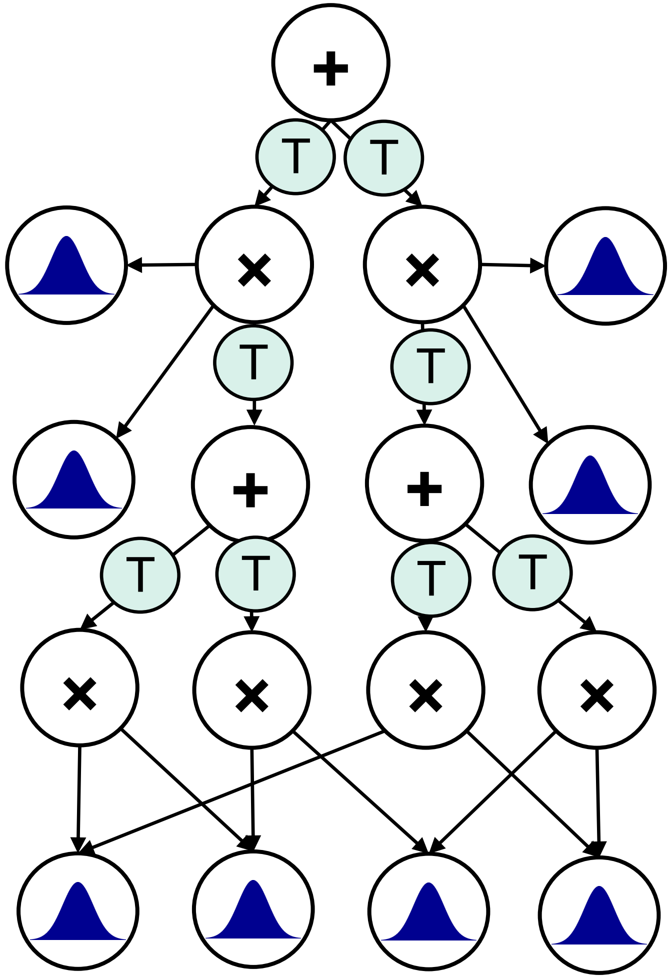
$$= \int p_1(f_1(\boldsymbol{x_1}))p_2(f_2(\boldsymbol{x_2}))|\det J_f| d\boldsymbol{x} = \left(\int p_1(f_1(\boldsymbol{x_1}))|\det J_{f_1}| d\boldsymbol{x_1}\right)\left(\int p_2(f_2(\boldsymbol{x_2}))|\det J_{f_2}| d\boldsymbol{x_2}\right)$$



Sahil Sidheekh, Kristian Kersting, and Sriraam Natarajan. "Probabilistic flow circuits: towards unified deep models for tractable probabilistic inference." *Uncertainty in Artificial Intelligence*, 2023

Integrating PCs with Normalizing Flows

Implications of τ −Decomposability

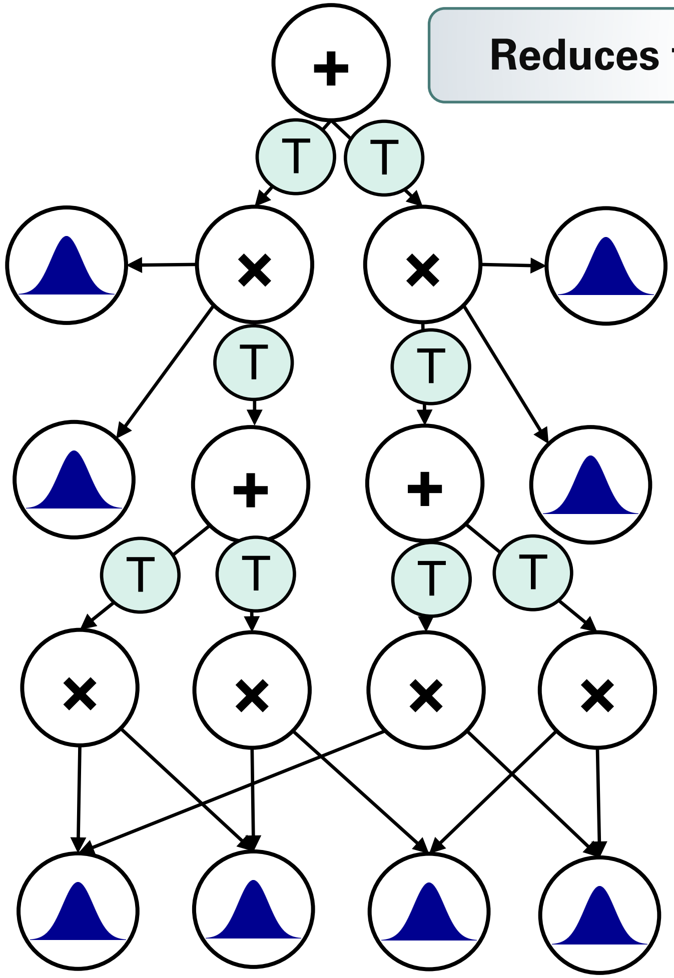


Sahil Sidheekh, Kristian Kersting, and Sriraam Natarajan. "Probabilistic flow circuits: towards unified deep models for tractable probabilistic inference." *Uncertainty in Artificial Intelligence*, 2023

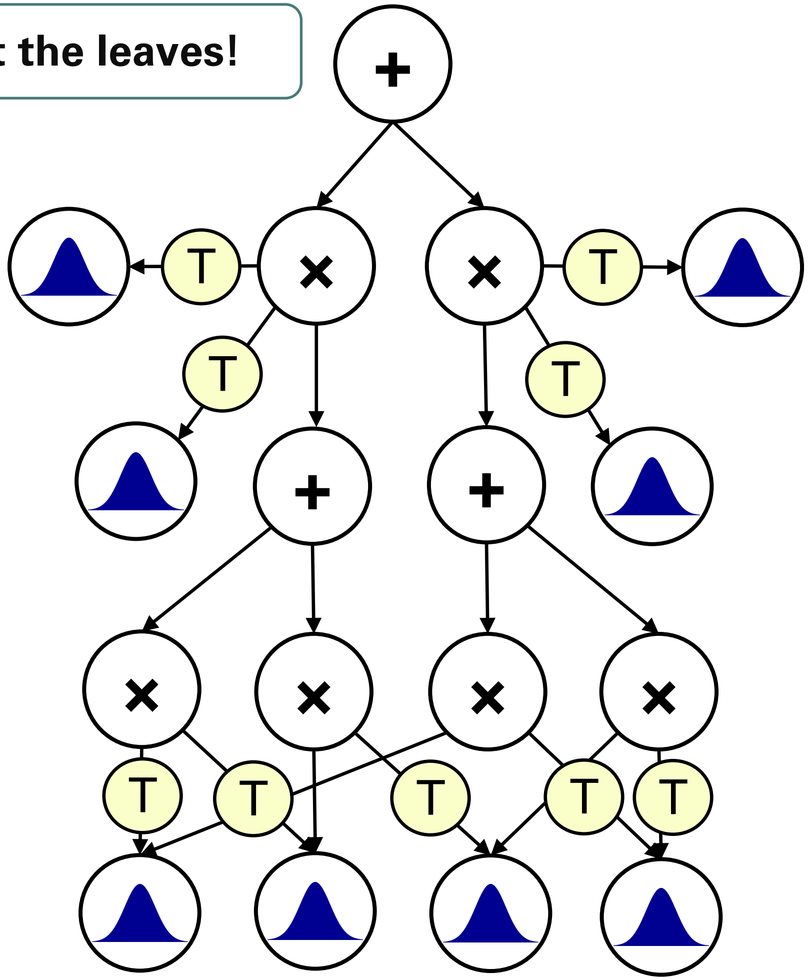
Integrating PCs with Normalizing Flows

Implications of τ –**Decomposability**

Reduces to having normalizing flows at the leaves!

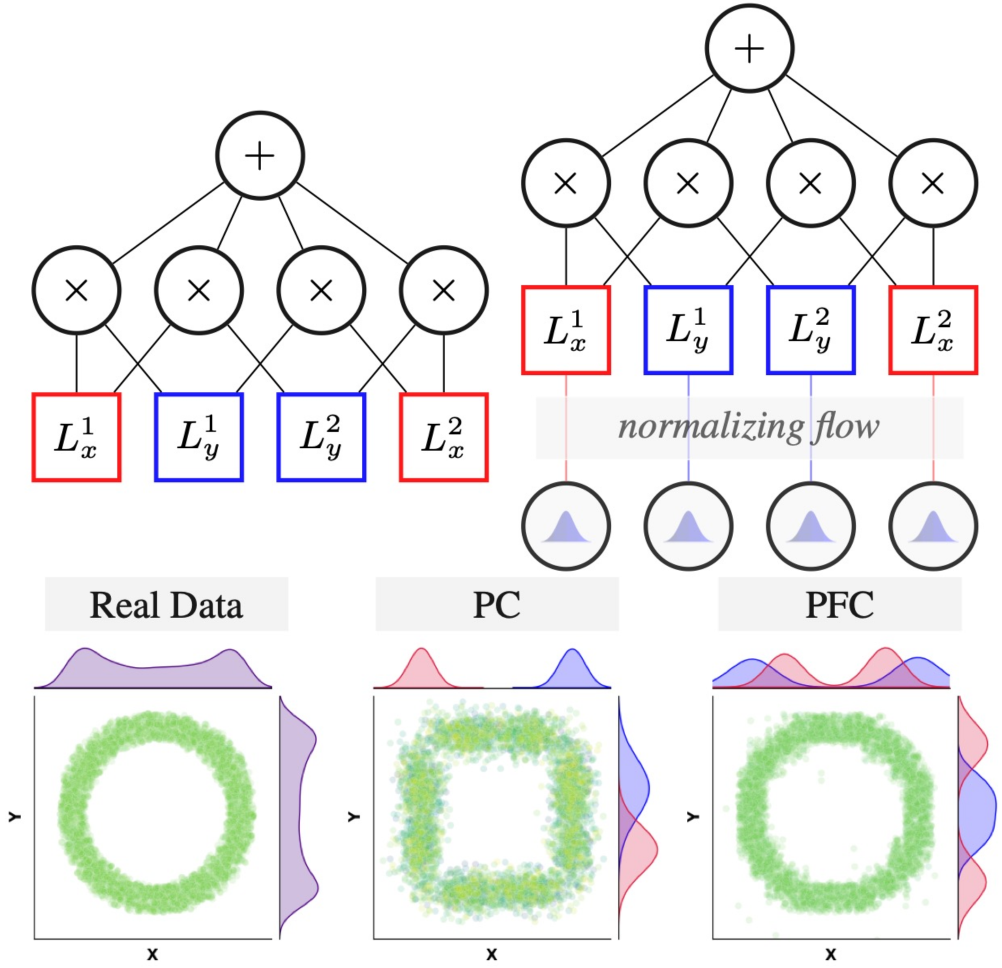


Compositions of transforms from leaf to root!



Sahil Sidheekh, Kristian Kersting, and Sriraam Natarajan. "Probabilistic flow circuits: towards unified deep models for tractable probabilistic inference." *Uncertainty in Artificial Intelligence*, 2023

Integrating PCs with Normalizing Flows



Probabilistic Flow Circuits

PCs with normalizing flows at the leaves

Has added Expressivity
Can model arbitrarily complex distributions at the leaves

Retains Tractability
As it encodes the same factorizations of the PC

Sahil Sidheekh, Kristian Kersting, and Sriraam Natarajan. "Probabilistic flow circuits: towards unified deep models for tractable probabilistic inference." *Uncertainty in Artificial Intelligence*, 2023

Use cases of PCs

A note on tractability

1. A given BN $\mathcal{M}$' can be compiled into a PC $\mathcal{M}$
2. Marginal inference on the PC $\mathcal{M}$ is O(poly(|$\mathcal{M}$ |))
3. Marginal inference on BNs is #P-complete.
4. NP = P? No.

Cooper, Gregory F. "The computational complexity of probabilistic inference using Bayesian belief networks." *Artificial intelligence* 42, no. 2-3 (1990):

DTPMs

Use cases of PCs

1. Explainability & Interpretability

2. Knowledge Intensive Learning

3. Structured object prediction

Explainability & Interpretability

Explainable Models $\supseteq$ **Interpretable Models**

Can give reasons for patterns embedded in the model

Exact working of model can be understood by a human

E.g., MLP can be explained by TREPAN

E.g., short decision tree is interpretable to a domain expert

Mark W. Craven and Jude W. Shavlik, ‘Extracting tree-structured representations of trained networks’, NeurIPS (1995)

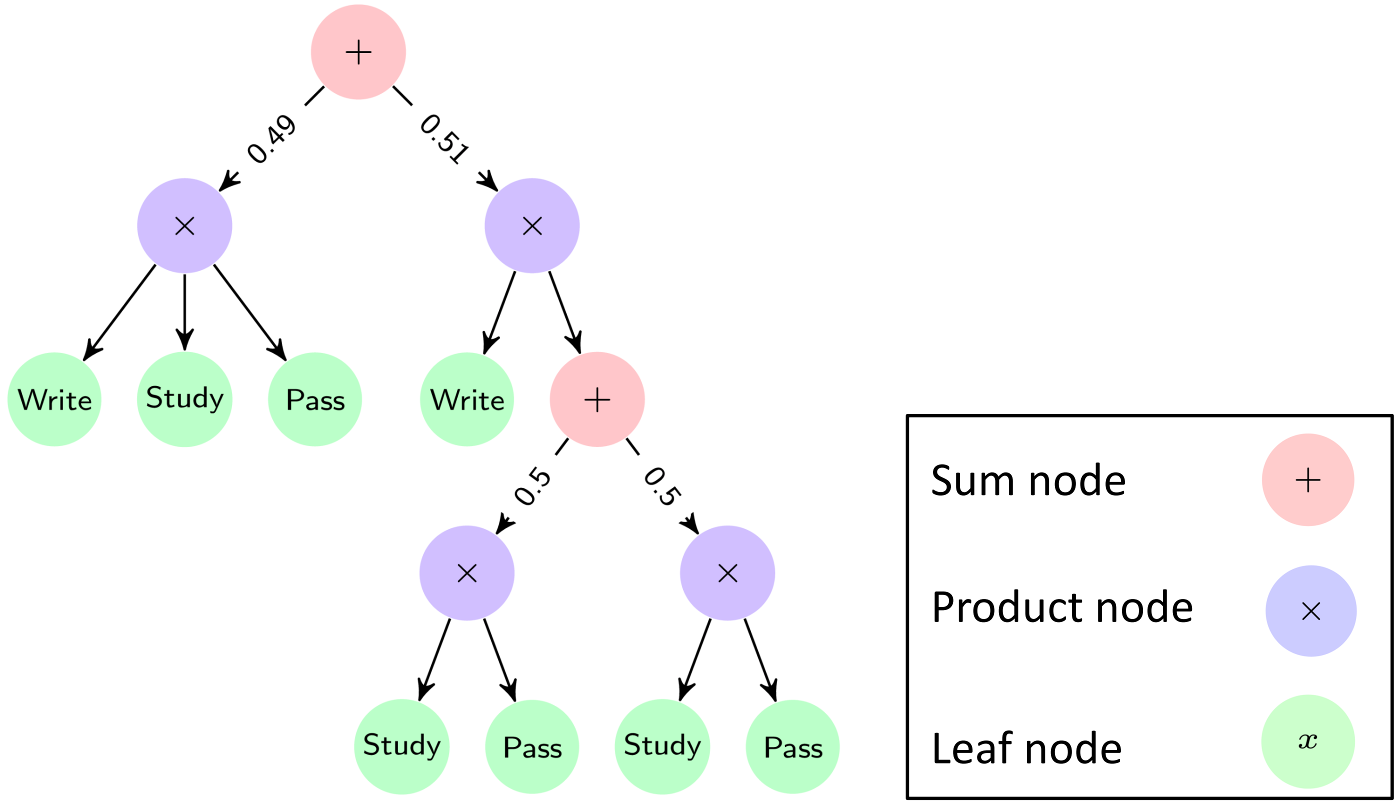
Delfosse, Quentin, Hikaru Shindo, Devendra Dhami, and Kristian Kersting. "Interpretable and Explainable Logical Policies via Neurally Guided Symbolic Abstraction." NeurIPS (2023).

Explaining Deep Tractable Probabilistic Models: The sum-product network case

Athresh Karanam*, Saurabh Mathur*, David M Haas, Predrag Radivojac, Kristian Kersting, Sriraam Natarajan

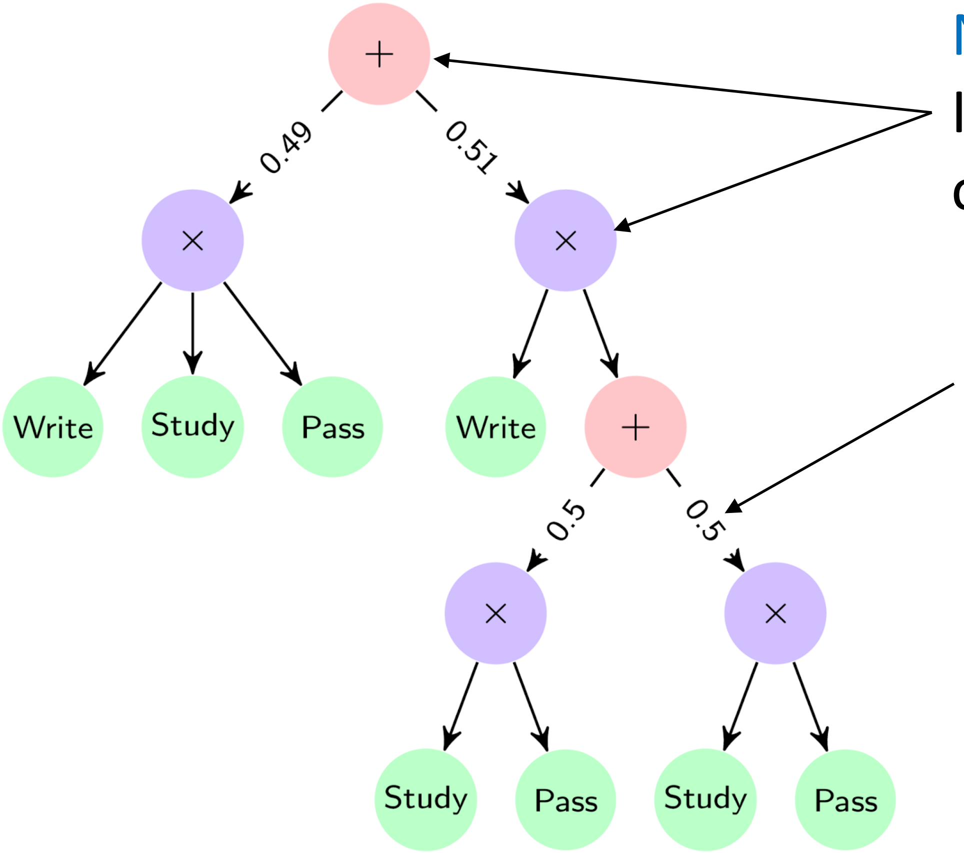
PGM (2022).

Sum-product networks



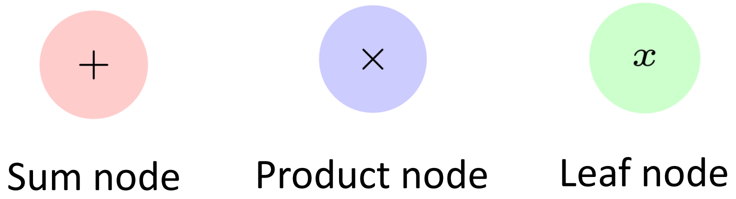
Hoifun Poon and Pedro Domingos, “Sum-product networks: A new deep architecture”, Proceedings of the Twenty-Seventh international conference on Uncertainty in artificial intelligence. 2011

But are they interpretable?

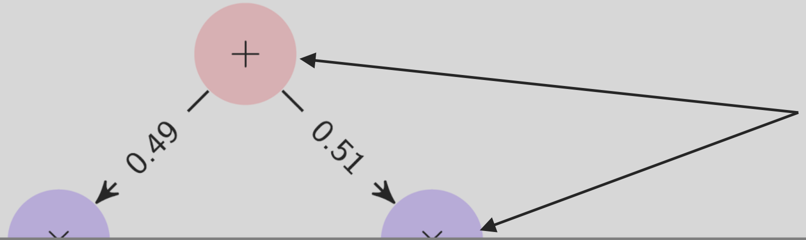


Not interpretable!
Internal nodes do not correspond to observable variables

Internal weights not interpretable



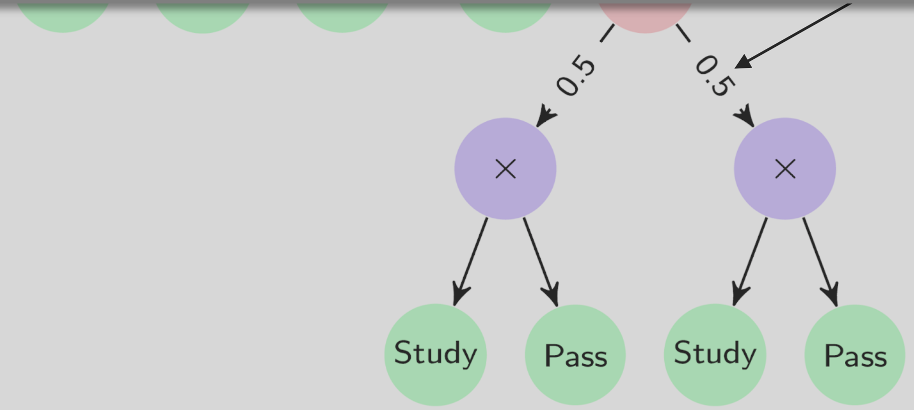
But are they interpretable?

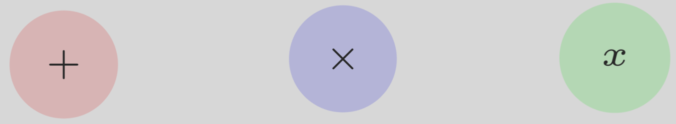


Not interpretable!

Internal nodes do not correspond to observable variables

Can we explain internal nodes in terms of observed variables?





Context-Specific Independence

"Passing the exam is independent of Studying if you do not write your answers."

$$Pass \perp Study | \neg Write$$

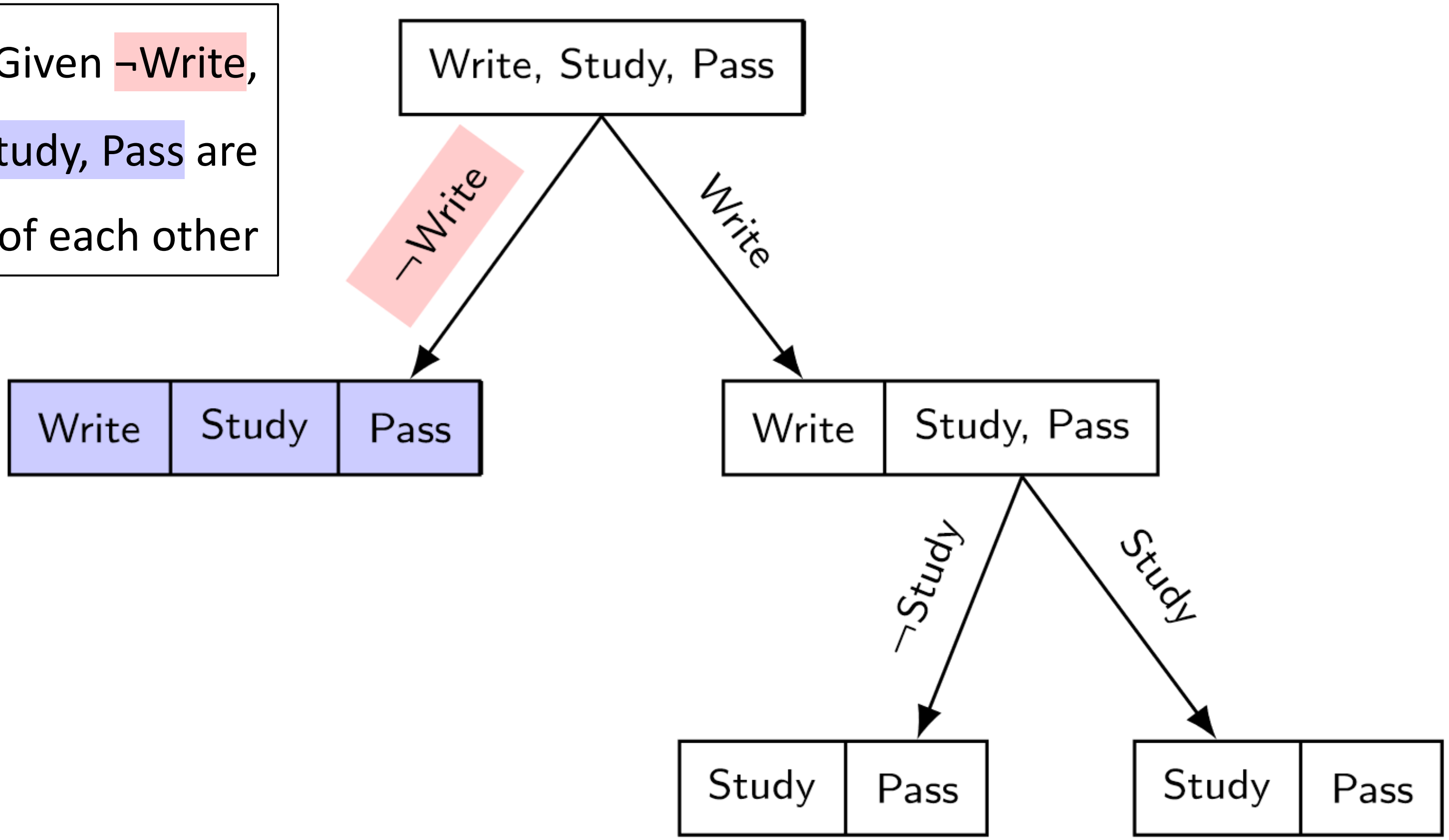
SPNs capture CSIs present in the data.

Knowledge as Context-specific independence

Given ¬Write,

Write, Study, Pass are

independent of each other



Craig Boutilier, Nir Friedman, Moises Goldszmidt and Daphne Koller, “Context-specific independence in bayesian networks”, Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence. 1996

Explaining DTPMs: Problem Statement

Given:

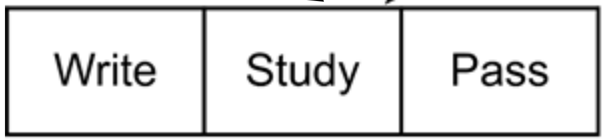
SPN $S = (G, \psi, w, \theta)$ and Data set $\mathcal{D}$

To Do:

Extract CSI-Tree $\tau = (G_{csi}, \psi_{csi}, \chi, \zeta)$

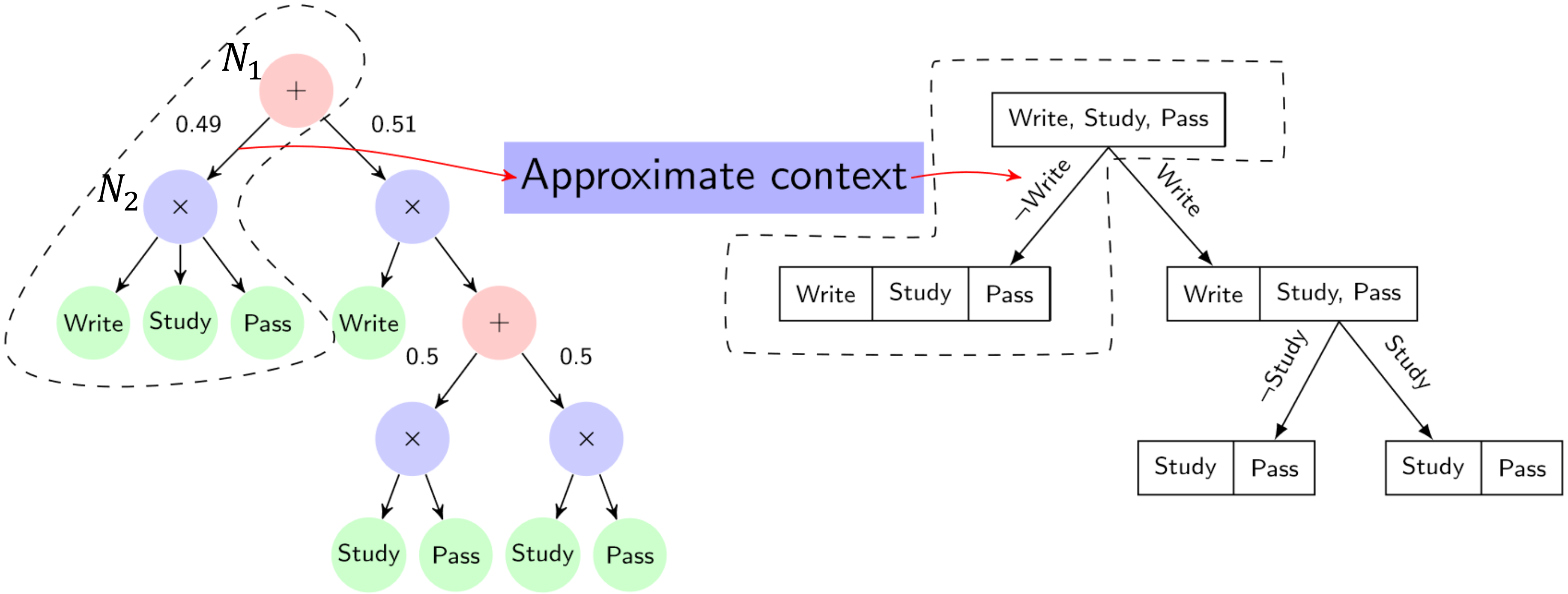
Partitions

Edge labels



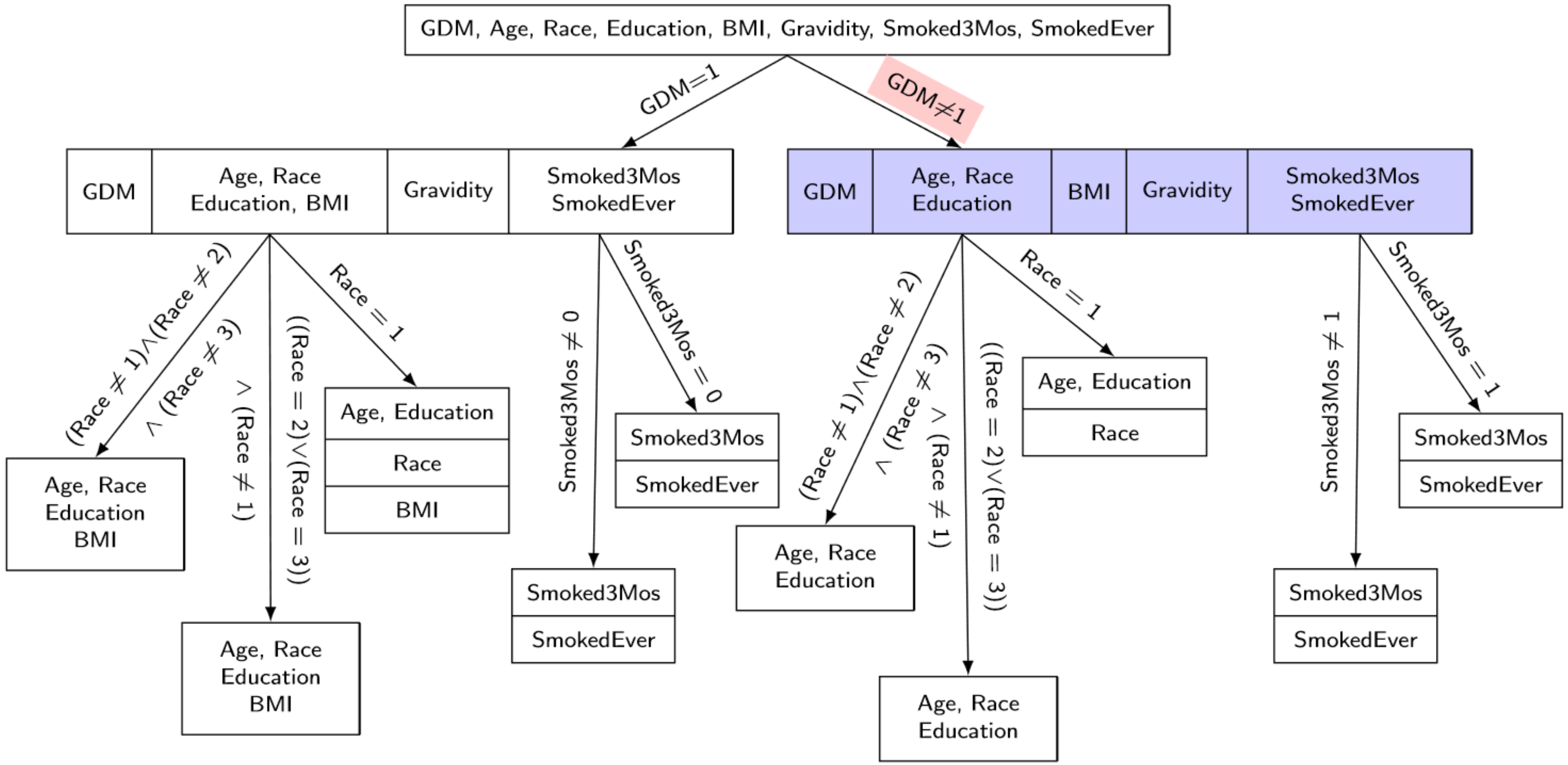
$\neg Write$

ExSPN: Explaining DTPMs using CSI-trees

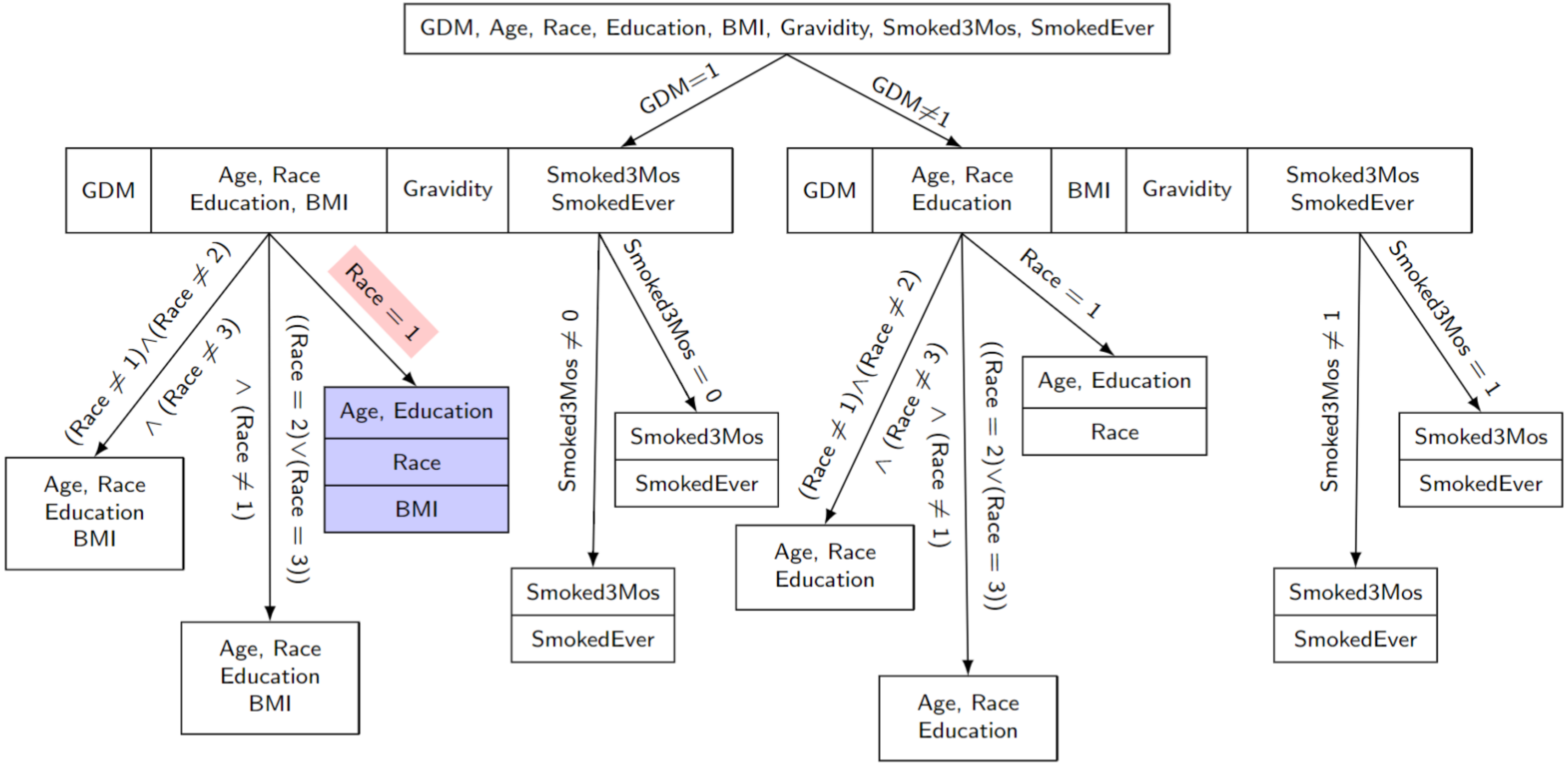


Write, Study & Pass are independent in the context of $\{\boldsymbol{x} \in \sup(N_2)\}$
They are not independent in $\{\boldsymbol{x} \in \sup(N_1)\backslash\sup(N_2)\}$

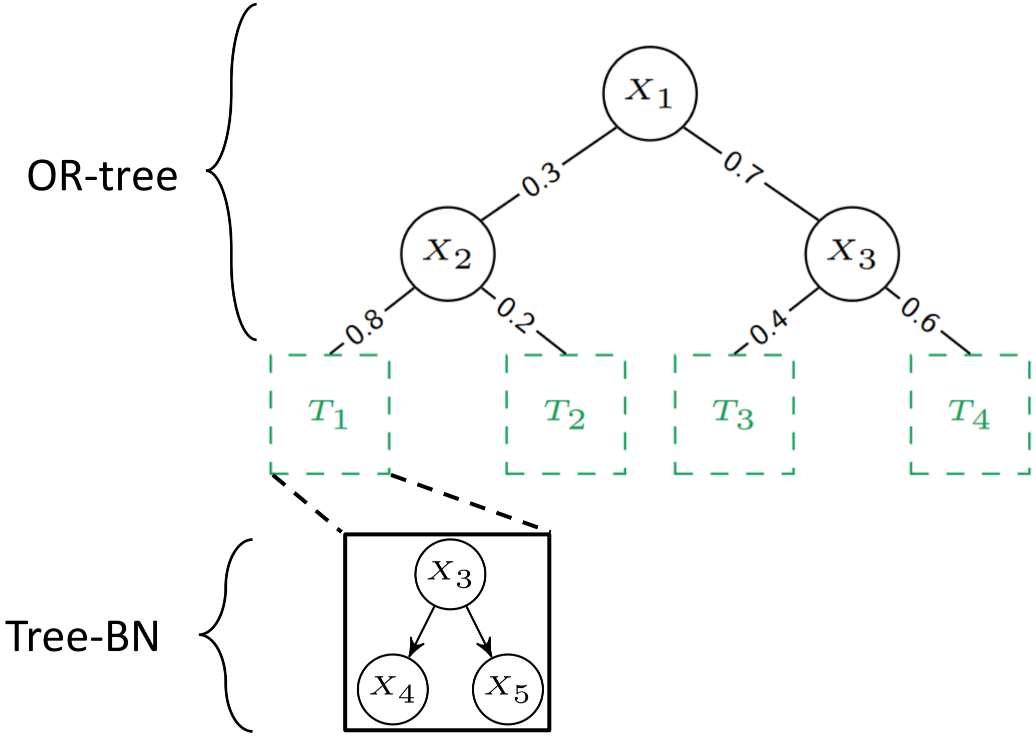
BMI is independent of Age, Race, Education in the cohort without GDM



BMI is independent of Age, Education in the non-Hispanic, white cohort with GDM



Cutset Networks

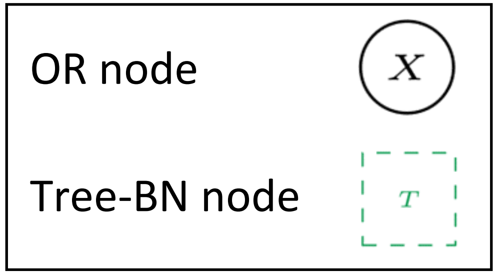


Combination of OR-trees & Tree-BNs

Tractable EVI, MAR and MAP queries

No latent variables

Naturally encodes variety of knowledge



Rahman, Tahrima, Prasanna Kothalkar, and Vibhav Gogate. "Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees." ECML-PKDD (2014)

LearnCNet: Learning Cutset Networks

Given: Data set $\mathcal{D}$ over n variables **X**

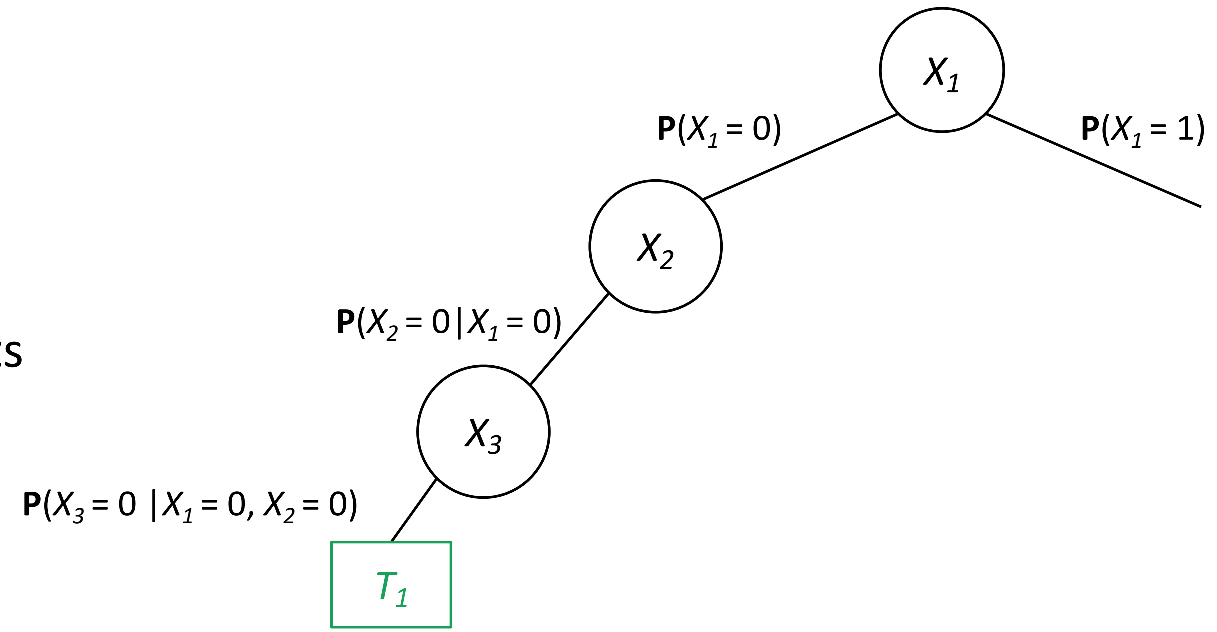
To Do: Learn a Cutset network $\mathcal{M}$

1. Select variable X_i using heuristic
2. Split on X_i, estimate edge weights
3. IF *stopping condition* not met,

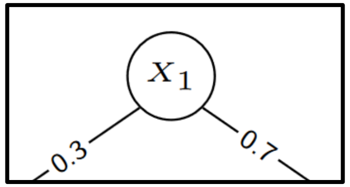
   Recurse into children

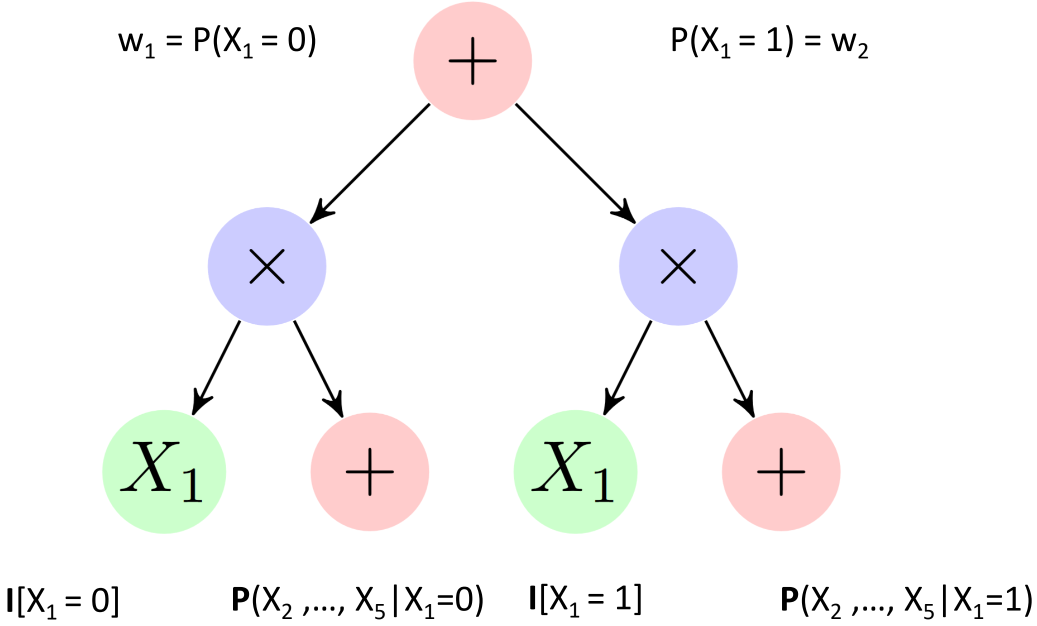
4. ELSE

   Learn tree-BN over remaining variables



Cutset Networks as deterministic PCs





DTPMs

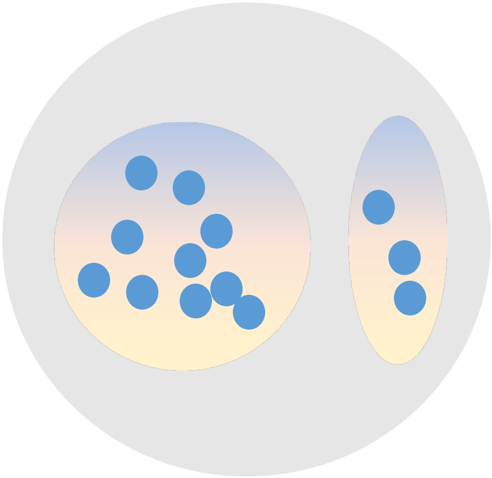
Use cases of PCs

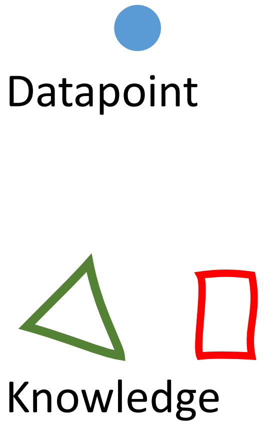
1. Explainability & Interpretability
2. **Knowledge Intensive Learning**
3. Structured object prediction

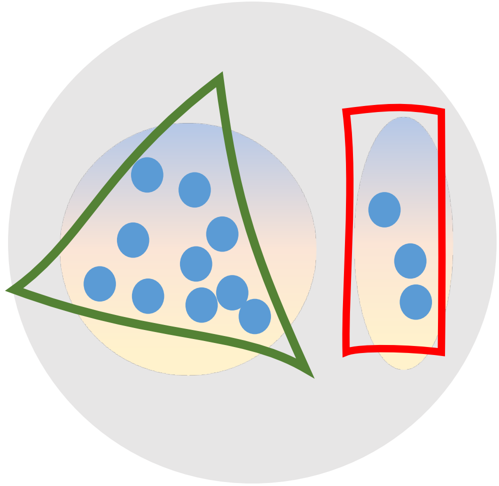
Knowledge Intensive Learning of Cutset Networks

Saurabh Mathur, Vibhav Gogate, Sriraam Natarajan

Uncertainty in AI (2023)







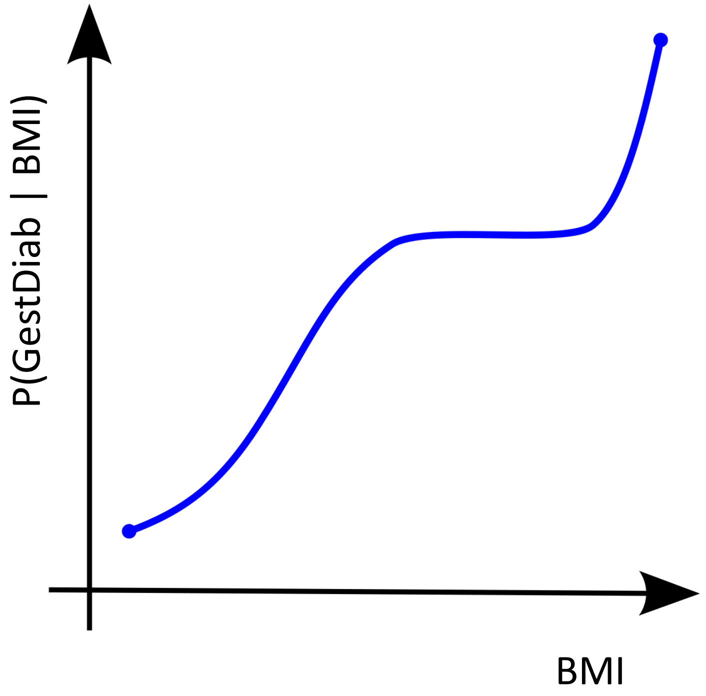
Can knowledge help us learn accurate generative models in data-scarce domains?

Domain Knowledge

- **Generalization.** Similar data points classified similarly
- **Qualitative influences.** General trends in distribution
- **Class imbalance.** Cost of FPR vs. FNR
- **Privileged information.** For eg, Fine-grained data
- **Fairness.** Similar performance for similar data points

Phillip Odom and Sriraam Natarajan. "Human-guided learning for probabilistic logic models." Frontiers in Robotics and AI (2018)

Qualitative Influence Statements



Risk of GestDiab ↑, as BMI ↑

"Risk of Gestational Diabetes *increases* as Body Mass Index *increases.*"

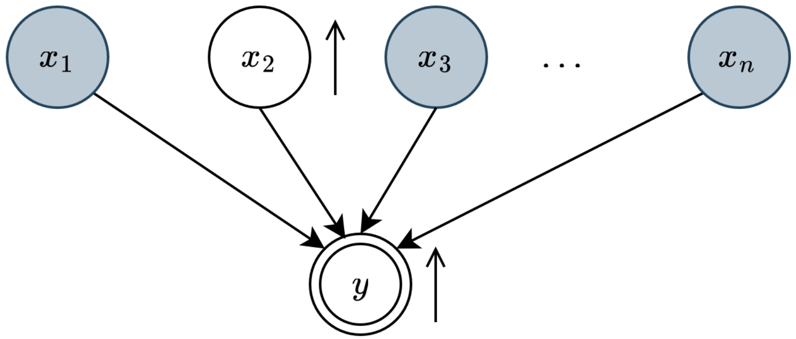
$$\mathrm{BMI}\prec^{M+}\mathrm{GestDiab}$$

- **Concisely** expresses a trend in a conditional distribution
- Human **interpretable**
- **Aligns** with how experts think about risk

Eric Altendorf, Angelo C. Restificar, and Thomas G. Dietterich. Learning from sparse data by exploiting monotonicity constraints. UAI (2005).

Qualitative Influences as constraints in BNs

$X_2 \overset{M+}{\prec} Y$ *ceteris paribus*



Altendorf, Eric E., Angelo C. Restificar, and Thomas G. Dietterich. "Learning from sparse data by exploiting monotonicity constraints." UAI (2005).

Knowledge Intensive Learning of Cutset Networks

Does **qualitative knowledge** integrate well with the **patterns learned from data** by cutset networks?

KICN: Problem Statement

Given:

$\mathcal{D}$, a data set of over variables **X**

C, a set of monotonic influence statements

To Do:

Learn a cutset network, $\mathcal{M}$ that models **P**(**X**)

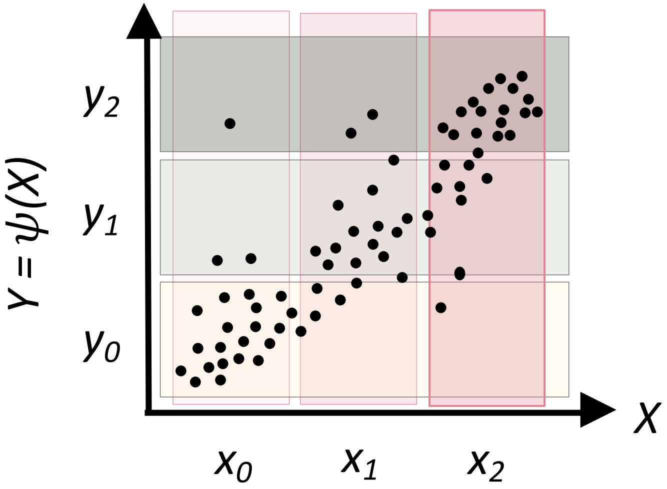
KICN: Penalized objective function

$$\arg\max_{\mathcal{M}} \; \underbrace{\mathcal{L}(\mathcal{M}, \mathcal{D})}_{\text{Data}} - \lambda \underbrace{\sum_{i,j} \zeta_{i,j}(\mathcal{M}, C)}_{\text{Knowledge}}$$

- Knowledge-based penalty serves as regularization
- Measures the deviation from the knowledge
- Computed efficiently due to tractability

Qualitative Influences as constraints in BNs

$$X \overset{M+}{\underset{\prec}{}} Y$$



Order Restricted Constraints

$$x_0 < x_1 \implies \psi(x_0) \leq \psi(x_1)$$

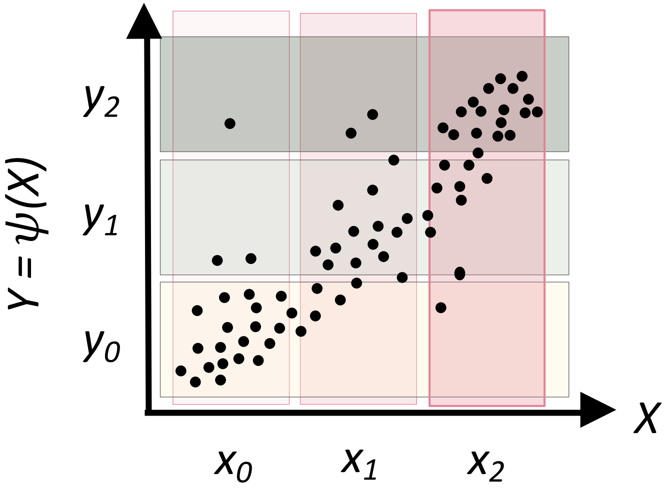
Conditional Probability Constraints

$$x_0 < x_1 \implies \underbrace{P(Y \leq y_1 \mid x_0)}_{\textit{ceteris paribus}} \geq P(Y \leq y_1 \mid x_1)$$

Altendorf, Eric E., Angelo C. Restificar, and Thomas G. Dietterich. "Learning from sparse data by exploiting monotonicity constraints." UAI (2005).

Qualitative Influences as constraints ~~in BNs~~

$$X \overset{M+}{\underset{\prec}{}} Y$$



Order Restricted Constraints

$$x_0 < x_1 \implies \psi(x_0) \leq \psi(x_1)$$

Conditional Probability Constraints

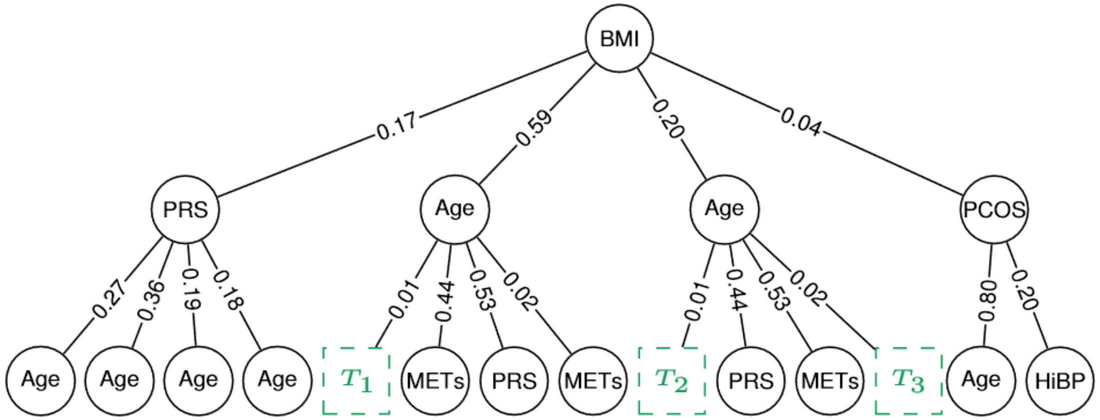
$$x_0 < x_1 \implies \underbrace{P(Y \leq y_1 \mid x_0)}_{\text{\sout{ceteris paribus} marginal inference}} \geq P(Y \leq y_1 \mid x_1)$$

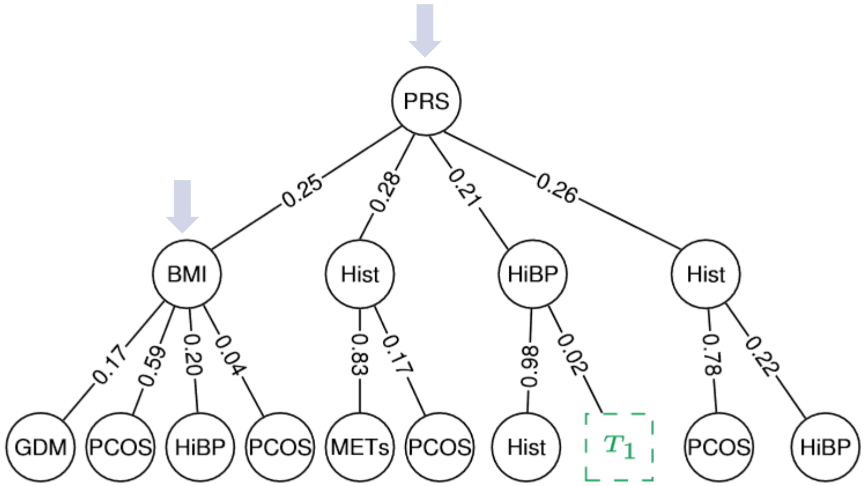
Altendorf, Eric E., Angelo C. Restificar, and Thomas G. Dietterich. "Learning from sparse data by exploiting monotonicity constraints." UAI (2005).

KICN learns more **concise** and **accurate** models

Data set	Edge count		Parameter count		MSE on queries	
	LearnCNet	KICN	LearnCNet	KICN	LearnCNet	KICN
ppd	**113.8**	114.1	205.7	**198.8**	0.2043	**0.1963**
adni	121.9	**57.8**	343.3	**246.4**	0.1825	**0.1636**
numom2b-a	179.4	**108.6**	422.2	**366.3**	0.0397	**0.0383**
numom2b-b	416.5	**220.9**	1,069.9	**905.7**	0.0515	**0.0445**

KICN learns more **concise** and **accurate** models





Pagel, Kymberleigh A., et al. "Association of Genetic Predisposition and Physical Activity With Risk of Gestational Diabetes in Nulliparous Women." JAMA network open (2022)

DTPMs

Use cases of PCs

1. Explainability & Interpretability
2. Knowledge Intensive Learning
3. **Structured object prediction**

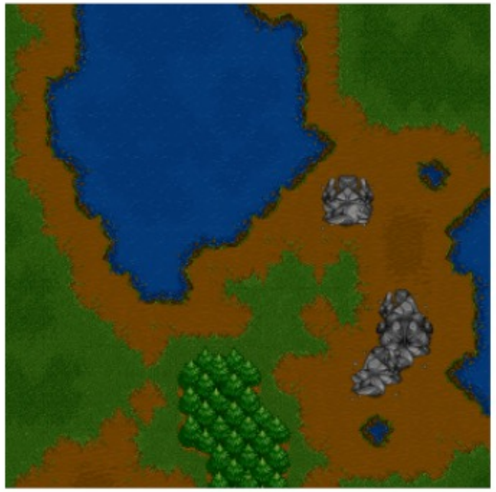
Semantic Probabilistic Layers for Neuro-Symbolic Learning

Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, Antonio Vergari

NeurIPS (2022).

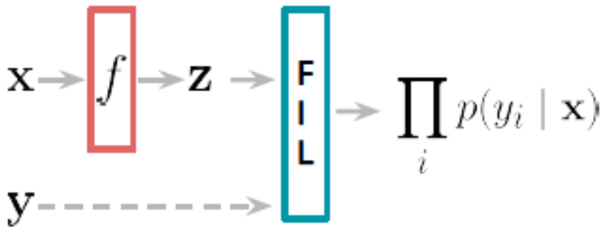
Structured object prediction in Warcraft

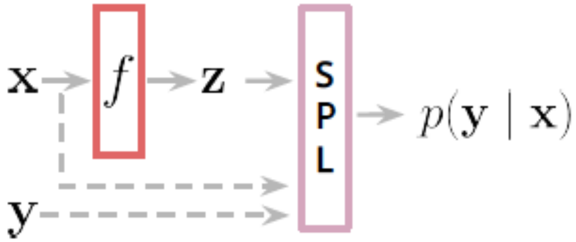
Input



$$\rightarrow \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

Structured object prediction in Warcraft

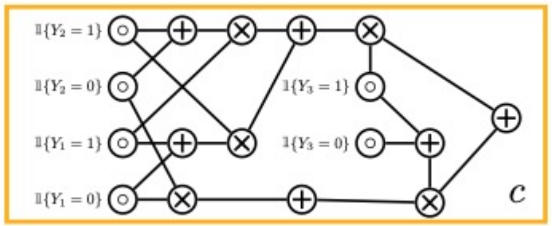


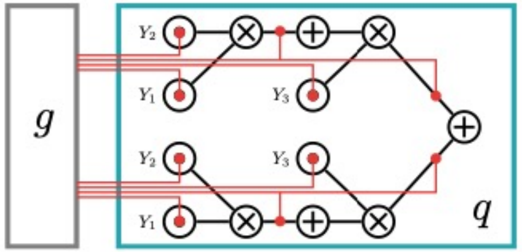


The Semantic Probabilistic Layers Framework

Overall Pipeline

$$\mathsf{K}: (Y_1 = 1 \implies Y_3 = 1)$$
$$\wedge \quad (Y_2 = 1 \implies Y_3 = 1)$$





1 Take your logical constraint

2 Compile it into a constraint circuit

3 Multiply it by a circuit distribution

4 Train end to end using gradient descent

The Semantic Probabilistic Layers Framework

Building q and c as tractable circuits.

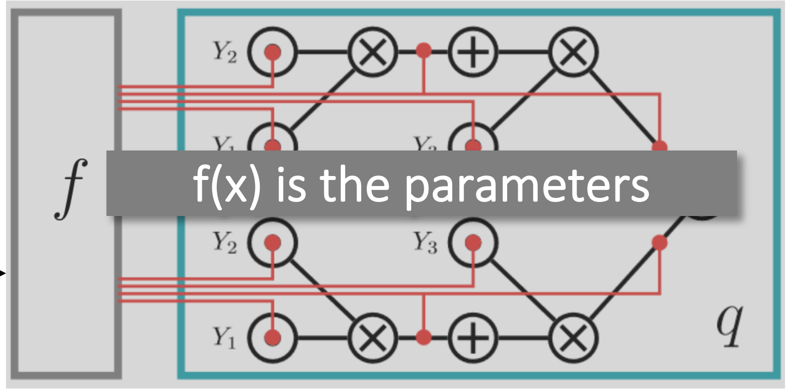
$\mathbf{x}$ Input Features

$\mathbf{y}$ Structured Target

$f(\mathbf{x})$ MLP Encoder

$q_\Theta(\mathbf{y} \mid f(\mathbf{x}))$ Conditional Probabilistic Circuit

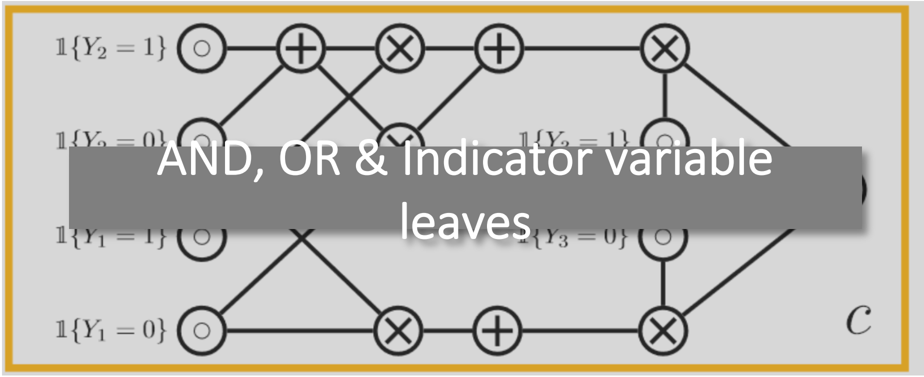
Smooth, Decomposable, Compatible



$c_\mathrm{K}(\mathbf{x}, \mathbf{y})$ Logical Constraint Circuit

Smooth, Decomposable, Compatible

Deterministic



Generating text with constraints

S

Write a sentence about CODS-COMAD using the words Machine Learning, Research and Artificial Intelligence in the given order.

CODS-COMAD is a premier conference that fosters cutting-edge research at the intersection of Machine Learning and Artificial Intelligence.

Zhang, Honghua, Meihua Dang, Nanyun Peng, and Guy Van den Broeck. "Tractable control for autoregressive language generation. " ICML (2023).

Generating text with constraints

S Write a sentence about AI

Artificial Intelligence (AI) is a branch of computer science that focuses on the development of systems and algorithms capable of performing tasks that typically require human intelligence, such as learning, problem-solving, and pattern recognition.

Zhang, Honghua, Meihua Dang, Nanyun Peng, and Guy Van den Broeck. "Tractable control for autoregressive language generation. " ICML (2023).

Text generation

Prefix:	The weather in Bangalore is

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}}\ \underbrace{\text{typically warm and sunny during the summer months.}}_{\boldsymbol{x}_{t:n}}$$

$$\arg\max_{\boldsymbol{x}_{t:n}} \boldsymbol{P}_M(\text{Suffix} = \boldsymbol{x}_{t:n} | \text{ Prefix} = \boldsymbol{x}_{1:t-1})$$

Constrained text generation

Prefix:	The weather in Bangalore is
Constraint, α:	Must contain the word "summer"

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}} \; \underbrace{\text{typically warm and sunny during the summer months.}}_{\boldsymbol{x}_{t:n}}$$

$$\underset{\substack{\boldsymbol{x}_{t:n} \\ \text{s.t. } \alpha(\boldsymbol{x}) \text{ is true}}}{\arg\max} \quad \boldsymbol{P}_M(\text{Suffix} = \boldsymbol{x}_{t:n} \mid \text{Prefix} = \boldsymbol{x}_{1:t-1})$$

Constrained text generation

Prefix:	The weather in Bangalore is
Constraint, α:	Must contain the word "summer"

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}}\ \underbrace{\text{typically}}_{x_t}\ \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

$$\arg\max_{x_t}\ \boldsymbol{P}_M(\text{NextWord} = x_t \mid \text{Prefix} = \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = \text{true})$$

Constrained text generation

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}} \underbrace{\text{typically}}_{x_t} \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

$\boldsymbol{P}_M(\text{NextWord} = x_t | \text{ Prefix} = \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true) = 0$, If generating x_t makes $\alpha(\boldsymbol{x}) = false$

Constrained text generation

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}}\ \underbrace{\text{typically}}_{x_t}\ \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

$$\boldsymbol{P}_M(\text{NextWord} = x_t |\ \text{Prefix} = \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true) \propto$$
$$\boldsymbol{P}_M(\alpha(\boldsymbol{x}) = true |\ \text{Prefix} = \boldsymbol{x}_{1:t-1}, \text{NextWord} = x_t)\ \boldsymbol{P}_M(\text{NextWord} = x_t |\ \text{Prefix} = \boldsymbol{x}_{1:t-1})$$

Constrained text generation is intractable

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}}\ \underbrace{\text{typically}}_{x_t}\ \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

$$\boldsymbol{P}_M(\text{NextWord} = x_t | \text{ Prefix} = \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true) \propto$$

$$\underbrace{\boldsymbol{P}_M(\alpha(\boldsymbol{x}) = true | \text{ Prefix} = \boldsymbol{x}_{1:t-1}, \text{NextWord} = x_t)}\ \boldsymbol{P}_M(\text{NextWord} = x_t | \text{ Prefix} = \boldsymbol{x}_{1:t-1})$$

$$\boldsymbol{P}_M(\alpha(\boldsymbol{x}) = true \mid \text{Prefix} = \boldsymbol{x}_{1:t}) = \sum_{\alpha(\boldsymbol{x}) \text{ is true}} \boldsymbol{P}_M(x_{t+1:n} | \text{ Prefix} = \boldsymbol{x}_{1:t})$$

Tractable constrained text generation

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}}\ \underbrace{\text{typically}}_{x_t}\ \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

Learn a PC Q such that

$$\boldsymbol{P}_M(\text{NextWord} = x_t \mid \text{Prefix} = \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true) \approx \boldsymbol{P}_Q(\text{NextWord} = x_t \mid \text{Prefix} = \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true)$$

1. Sample sequences $\boldsymbol{x}$ from M.
2. Build data set of $\langle\, \boldsymbol{x}, \alpha(\boldsymbol{x})\, \rangle$
3. Build PC Q over $\boldsymbol{P}_Q(\boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}))$ to compute

$$\boldsymbol{P}_Q(\boldsymbol{x} \mid \alpha(\boldsymbol{x})) = \prod_t \boldsymbol{P}_Q(x_t \mid \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}))$$

Tractable constrained text generation

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}} \underbrace{\text{typically}}_{x_t} \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

Generate from

$$\boldsymbol{P}_{M'}(x_t |\, \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true) \propto \boldsymbol{P}_Q(\alpha(\boldsymbol{x}) = true |\, \boldsymbol{x}_{1:t})\, \boldsymbol{P}_M(x_t |\, \boldsymbol{x}_{1:t-1})$$

$= 0$, If generating x_t makes $\alpha(\boldsymbol{x}) = false$

Tractable constrained text generation

$$\underbrace{\text{The weather in Bangalore is}}_{\boldsymbol{x}_{1:t-1}} \underbrace{\text{typically}}_{x_t} \underbrace{\text{warm and sunny during the summer months.}}_{\boldsymbol{x}_{t+1:n}}$$

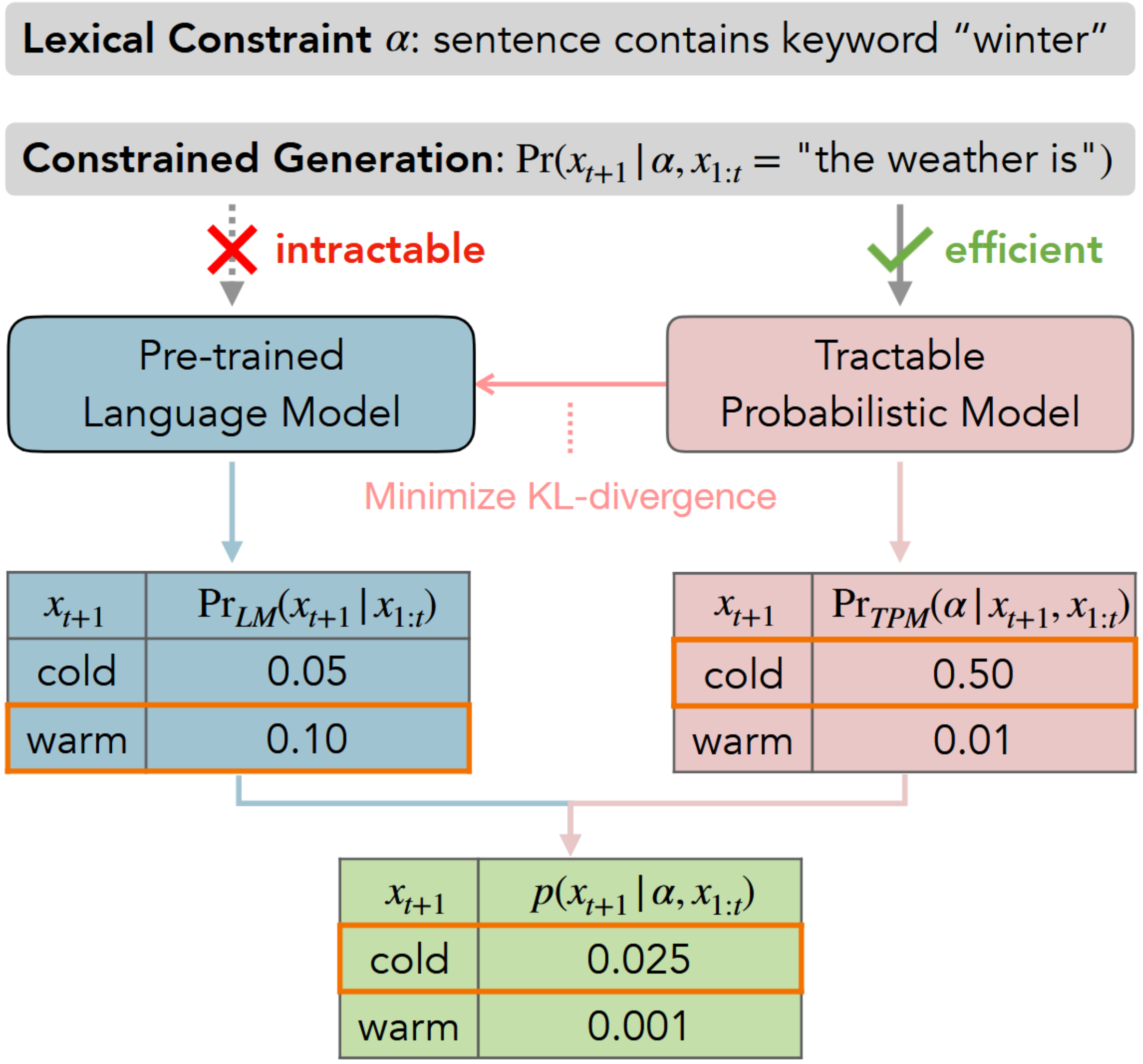
Generate from

$$\boldsymbol{P}_{M'}(x_t | \boldsymbol{x}_{1:t-1}, \alpha(\boldsymbol{x}) = true) \propto \boldsymbol{P}_Q(\alpha(\boldsymbol{x}) = true | \boldsymbol{x}_{1:t})^w \, \boldsymbol{P}_M(x_t | \boldsymbol{x}_{1:t-1})^{1-w}$$

$= 0$, If generating x_t makes $\alpha(\boldsymbol{x}) = false$

The GeLaTo framework

Generating Language with Tractable Constraints

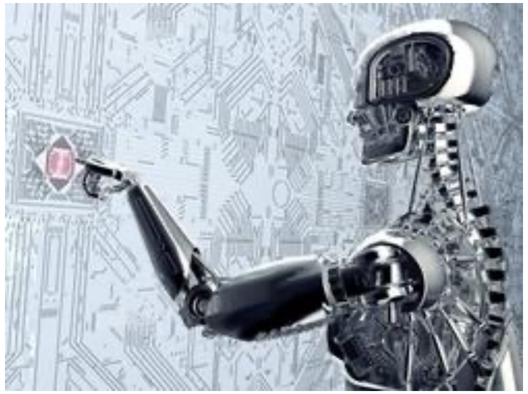


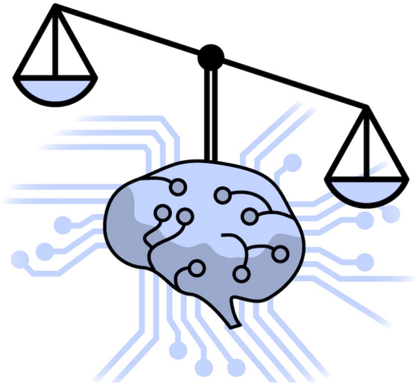
Summary & Takeaways

Tractable Probabilistic Inference

Allows performing complex reasoning over data effectively in the presence of uncertainty

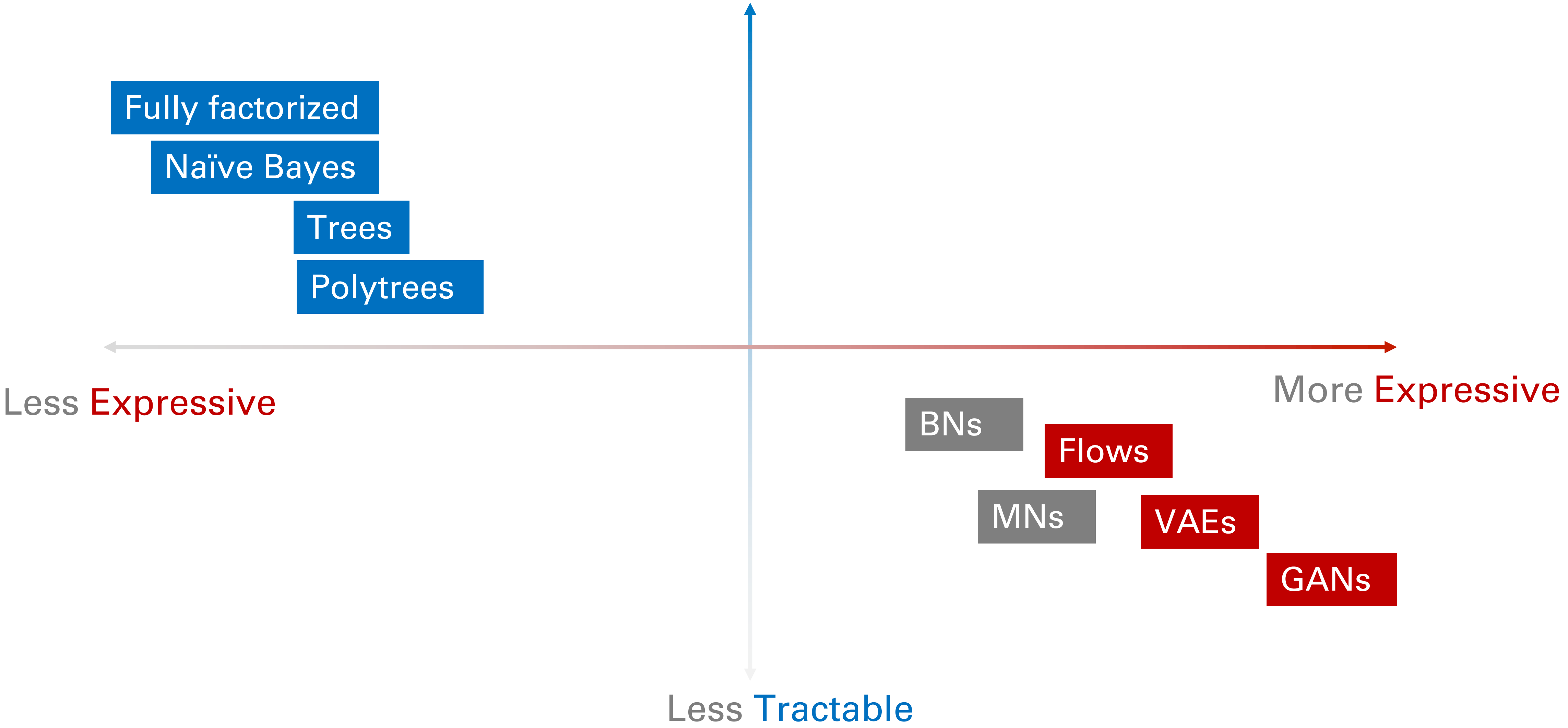
- Handle Missing Data
- Interpretable and Robust Decision Making
- Reason about arbitrary events of interest over random variables
- Marginalize out effects of sensitive features to ensure fairness
- Verifying and Incorporating domain knowledge



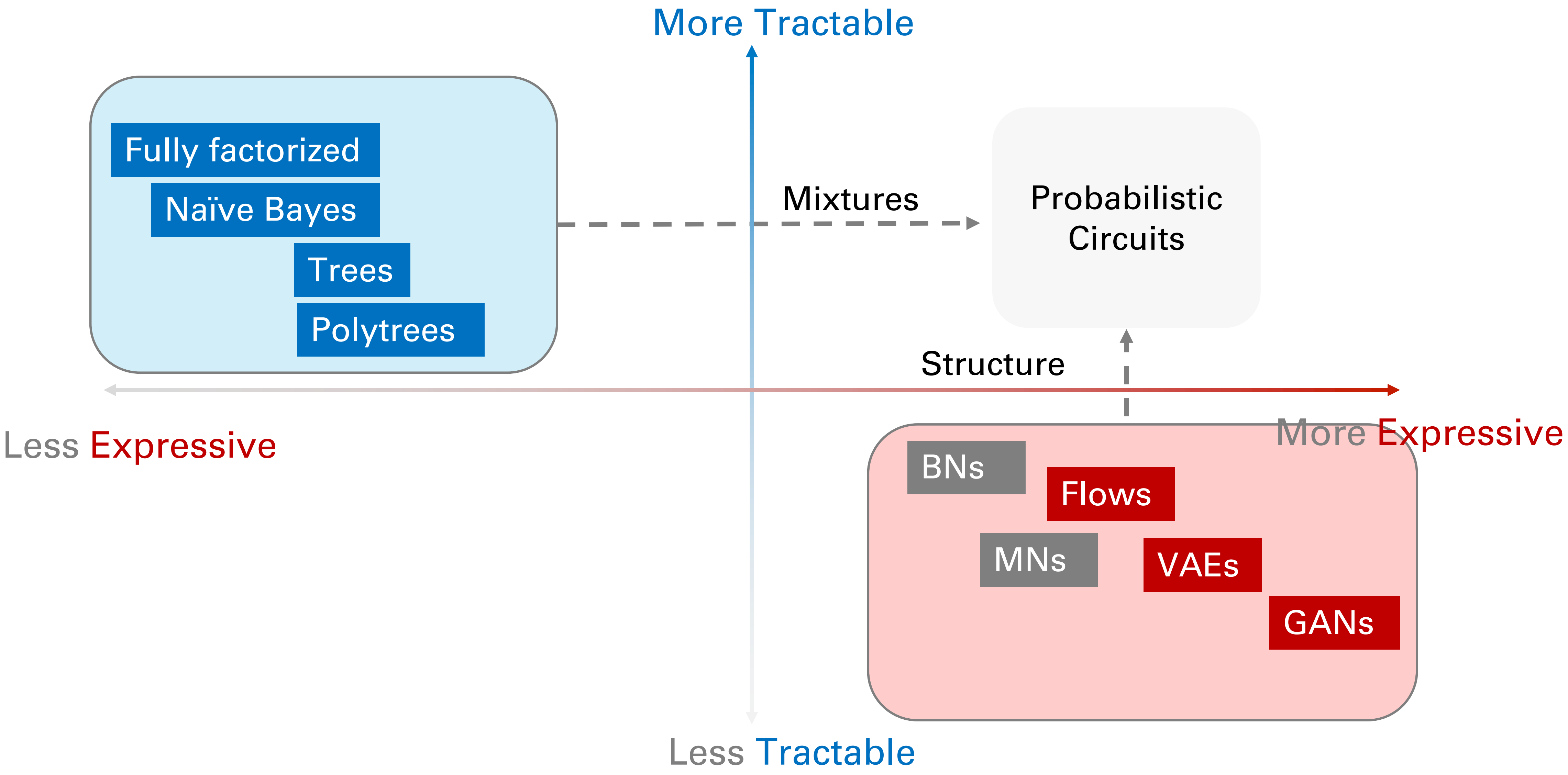


The Expressivity-Tractability Trade Off

However, performing exact inference tractably comes at the cost of expressivity

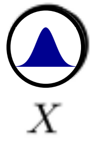


Probabilistic Circuits to achieve Best of Both

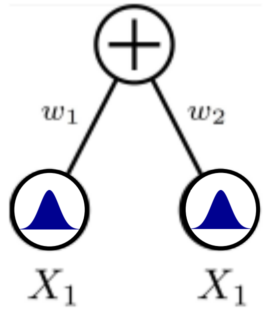


Probabilistic Circuits: Key Idea - Mixtures and Factors

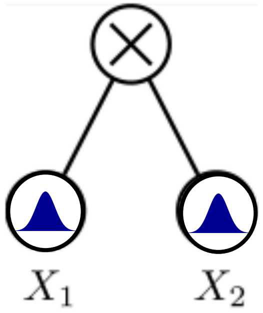
Computational graphs that recursively define distributions via 3 types of nodes



$$p(X) = Normal(X)$$



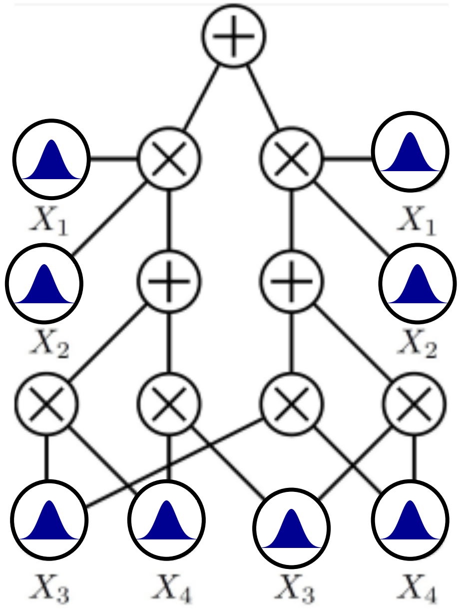
$$p(X_1) = w_1 p_1(X_1) + w_2 p_2(X_1)$$



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

Leaf nodes	Sum nodes	Product nodes
Simple univariate distributions	Represents **mixtures**	Represents **factorizations**
	Adds expressivity	Enables tractability

Stacked as a circuit

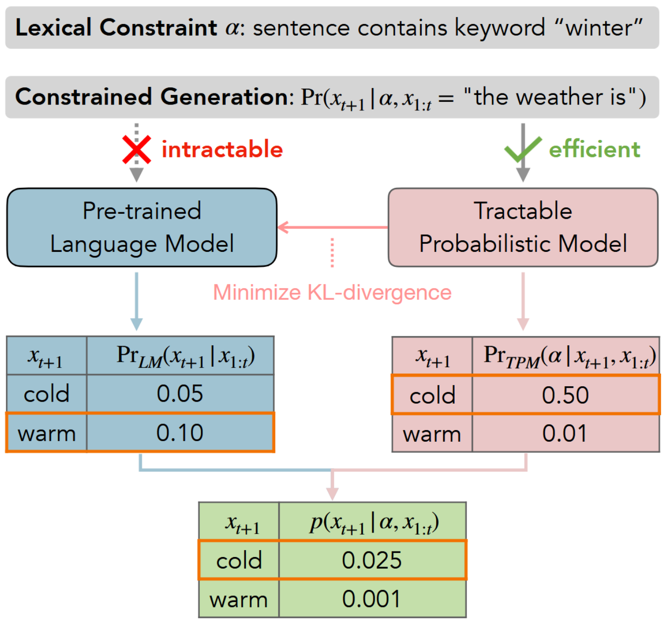


Probabilistic Circuits
Structure Helps Achieve Tractability!

Tractability is a spectrum
Need stronger structure for harder inference queries

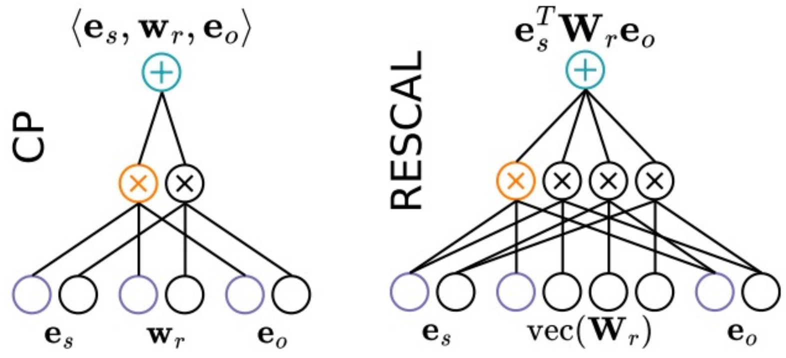
	EVI	MAR	CON	MAP
Smoothness	✓			
+Decomposability	✓	✓	✓	
+Determinism	✓	✓	✓	✓

Expressive and Tractable Models Can Benefit Diverse Domains



Controlled Text Generation

Structured-output prediction
Satisfying Constraints

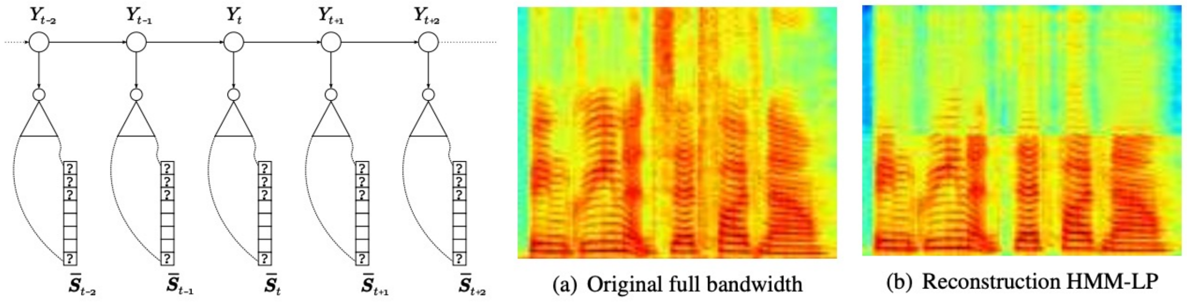


Knowledge Graph Embeddings

Zhang, Honghua, Meihua Dang, Nanyun Peng, and Guy Van den Broeck. "Tractable control for autoregressive language generation. " ICML (2023).
Ahmed, Kareem, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, and Antonio Vergari. "Semantic probabilistic layers for neuro-symbolic learning." NeurIPS (2022).
Loconte, Lorenzo, Nicola Di Mauro, Robert Peharz, and Antonio Vergari. "How to Turn Your Knowledge Graph Embeddings into Generative Models." NeurIPS (2023).

Expressive and Tractable Models Can Benefit Diverse Domains

Speech Reconstruction



Semantic Segmentation as MAP Inference

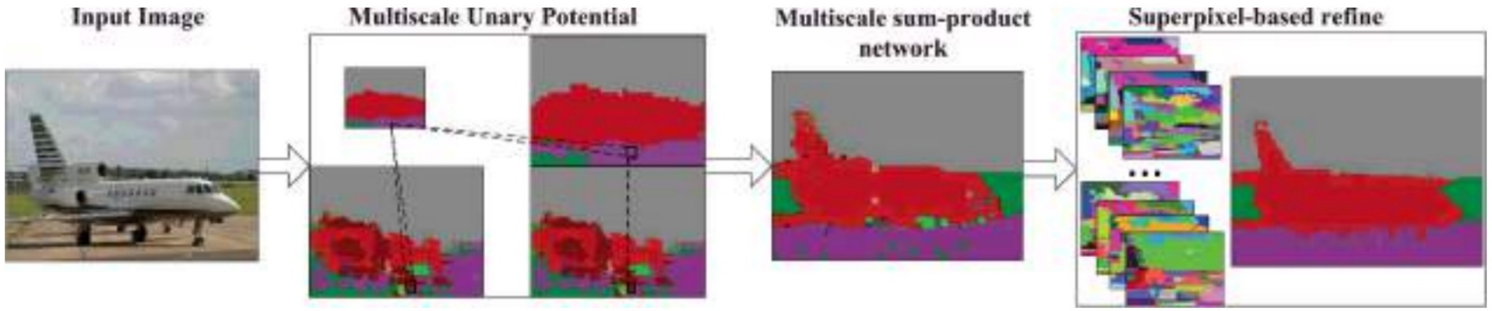
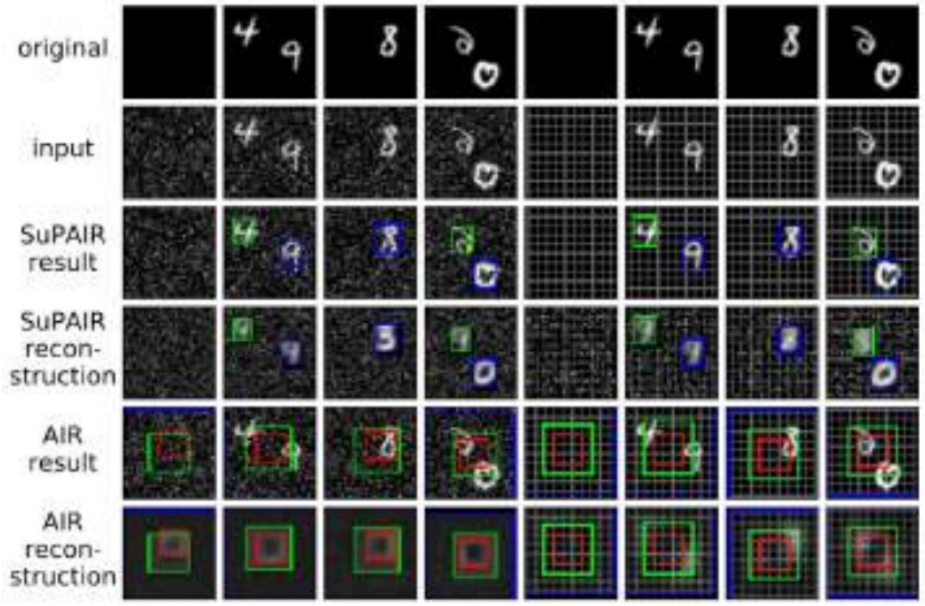
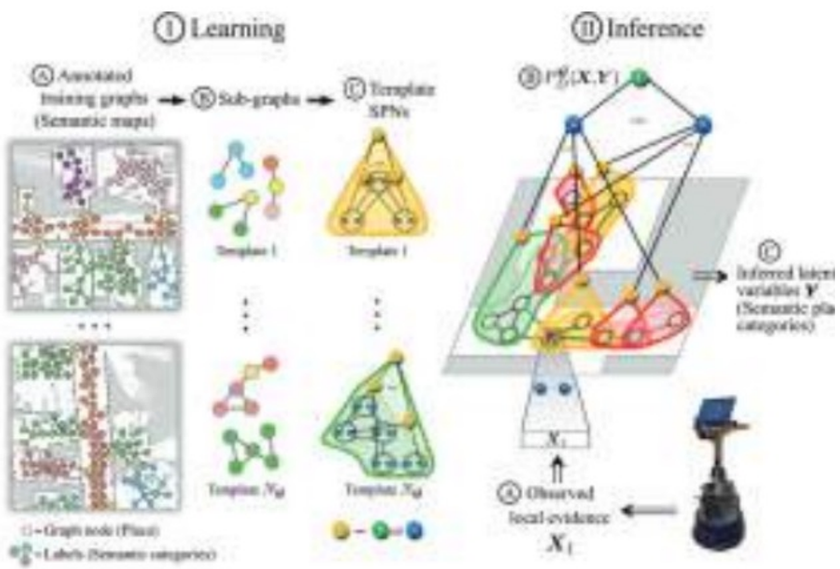


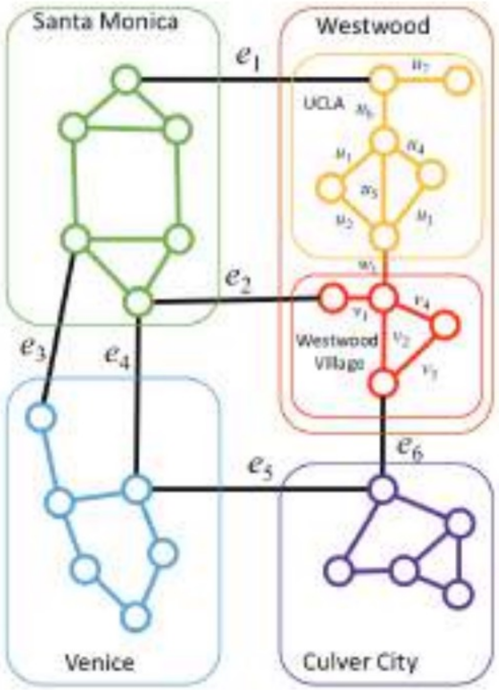
Image Inpainting

Yuan et al., “Modeling spatial layout for scene image understanding via a novel multiscale sum-product network”, 2016
Friesen et al., “Submodular Sum-product Networks for Scene Understanding”, 2016
Peharz, Robert, et al. "Modeling speech with sum-product networks: Application to bandwidth extension.” *International Conference on Acoustics, Speech and Signal Processing*, 2014.
Peharz, Robert, et al. "Einsum networks: Fast and scalable learning of tractable probabilistic circuits." *International Conference on Machine Learning*. PMLR, 2020.

Expressive and Tractable Models Can Benefit Diverse Domains







Scene Understanding **Robotics** **Routing**

Stelzner et al., "Faster Attend-Infer-Repeat with Tractable Probabilistic Models", 2019
Pronobis et al., "Learning Deep Generative Spatial Models for Mobile Robots",
2016 Pronobis et al., "Deep spatial affordance hierarchy: Spatial knowledge representation for planning in large-scale environments", 2017
Zheng et al., "Learning graph-structured sum-product networks for probabilistic semantic maps", 2018
Shen et al., "Conditional PSDDs: Modeling and learning with modular knowledge", 2018
Shen et al., "Structured Bayesian Networks: From Inference to Learning with Routes", 2019

Hands On Demo - Einsum Networks Implementing Deep PCs in Pytorch

https://bit.ly/cods-dtpm-2

Thank You!
Questions?