recent andvancements
in tractable probabilistic inference

antonio vergari (he/him)

@tetraduzione

26th Sept 2024 - **TransferLab Seminar**

april

april-tools.github.io

april

***a**utonomous &*
***p**rovably*
***r**eliable*
***i**ntelligent*
***l**earners*

april

***a**bout*
***pr**obabilities*
***i**ntegrals &*
***l**ogic*

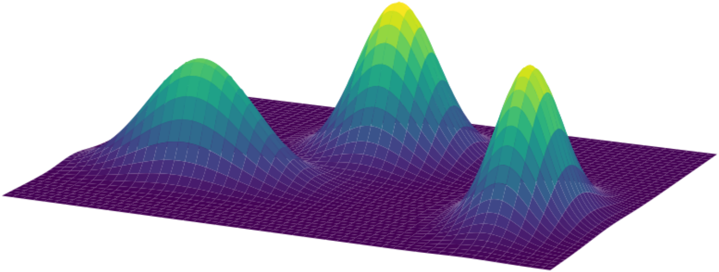
april

***a**pril is*
***p**robably a*
***r**ecursive*
***i**dentifier of a*
***l**ab*

deep generative models

+

flexible and reliable
(logic &) probabilistic reasoning?



a love letter to mixture models...

image taken from Hao Tang's course on ASR

Reasoning about ML models

q_1 *"What is the probability of a treatment for a patient with* ***unavailable records****?"*

q_2 *"How* ***fair*** *is the prediction is a certain protected attribute changes?"*

q_3 *"Can we certify no* ***adversarial examples*** *exist?"*

Reasoning about ML models

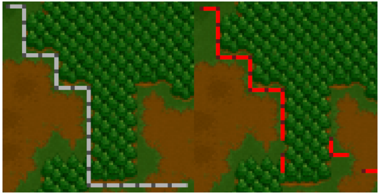
q_1 $\int p(\mathbf{x}_o, \mathbf{x}_m) d\mathbf{X}_m$
(missing values)

q_2 $\mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 0)} \left[f_0(\mathbf{x}_c) \right] - \mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 1)} \left[f_1(\mathbf{x}_c) \right]$
(fairness)

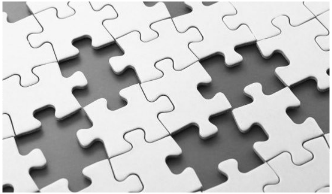
q_3 $\mathbb{E}_{\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_D)} \left[f(\mathbf{x} + \mathbf{e}) \right]$
(adversarial robust.)

...in the language of probabilities

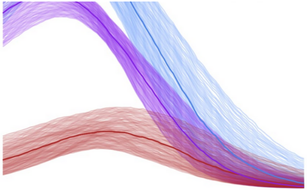
more complex reasoning



neuro-symbolic AI



probabilistic programming



computing uncertainties (Bayesian inference)

...and more application scenarios

Reasoning about ML models

q1 $\int p(\mathbf{x}_o, \mathbf{x}_m) d\mathbf{X}_m$
(missing values)

q2 $\mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 0)} \left[f_0(\mathbf{x}_c) \right] - \mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 1)} \left[f_1(\mathbf{x}_c) \right]$
(fairness)

q3 $\mathbb{E}_{\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_D)} \left[f(\mathbf{x} + \mathbf{e}) \right]$
(adversarial robust.)

hard to compute in general!

Reasoning about ML models

q_1 $\int p(\mathbf{x}_o, \mathbf{x}_m) d\mathbf{X}_m$

(missing values)

q_2 $\mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 0)} \left[f_0(\mathbf{x}_c) \right] - \mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 1)} \left[f_1(\mathbf{x}_c) \right]$

(fairness)

q_3 $\mathbb{E}_{\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_D)} \left[f(\mathbf{x} + \mathbf{e}) \right]$

(adversarial robust.)

it is crucial we compute them exactly and in polytime!

Reasoning about ML models

q_1 $\int p(\mathbf{x}_o, \mathbf{x}_m) d\mathbf{X}_m$
(missing values)

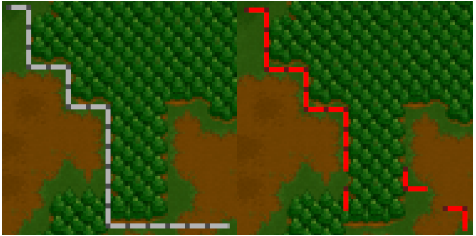
q_2 $\mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 0)} \left[f_0(\mathbf{x}_c) \right] - \mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 1)} \left[f_1(\mathbf{x}_c) \right]$
(fairness)

q_3 $\mathbb{E}_{\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_D)} \left[f(\mathbf{x} + \mathbf{e}) \right]$
(adversarial robust.)

it is crucial we compute them tractably!

why tractable models?

exactness can be crucial in safety-driven applications



guarantee constraint satisfaction

[Ahmed et al. 2022]

estimation error is bounded (0)

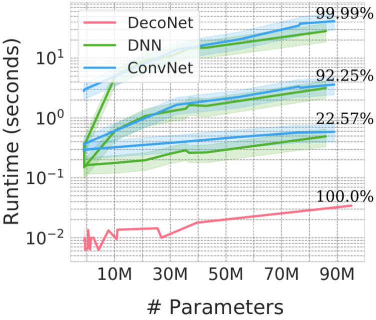
[Choi 2022]

why tractable models?

they can be much faster than intractable ones!

Method	MNIST (10,000 test images)		
	Theoretical bpd	Comp. bpd	En- & decoding time
PC (small)	1.26	1.30	**53**
PC (large)	**1.20**	**1.24**	168
IDF	1.90	1.96	880
BitSwap	1.27	1.31	904

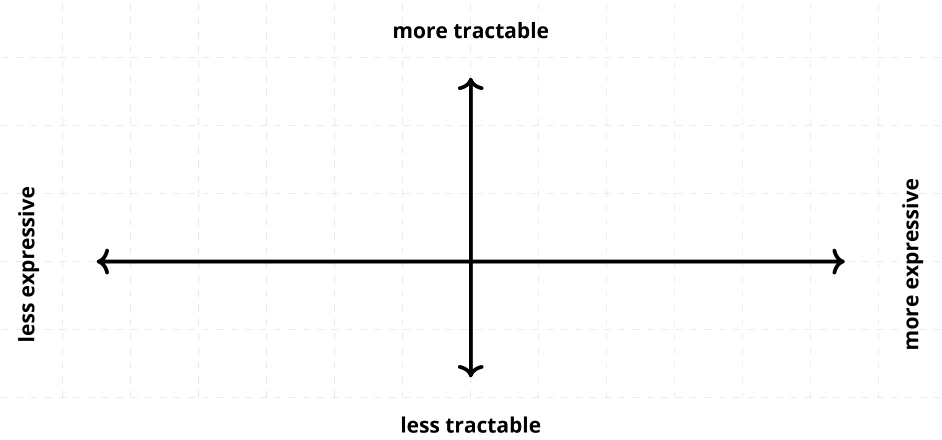
[Liu, Mandt, and Broeck 2022]

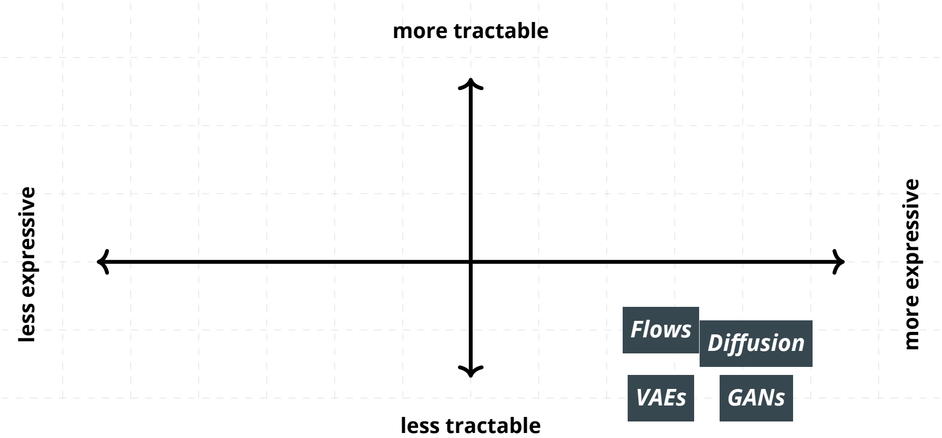


[Subramani et al. 2021]

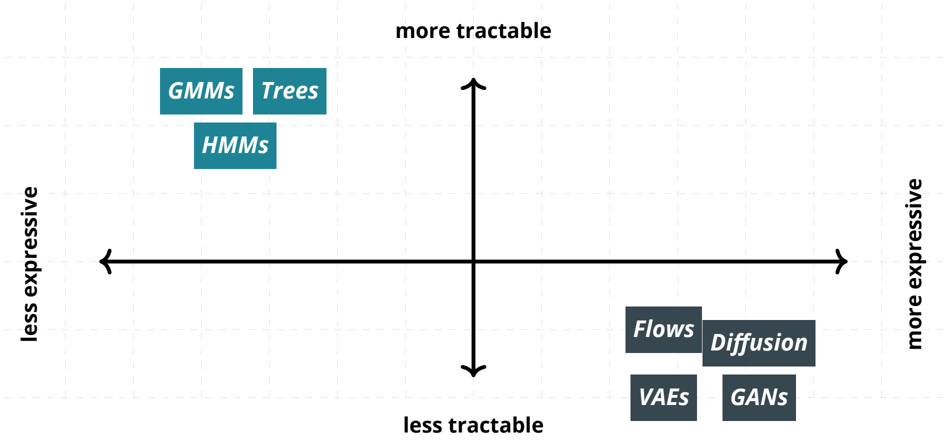
Goal

"Can we find a middle ground between tractability and expressiveness?"

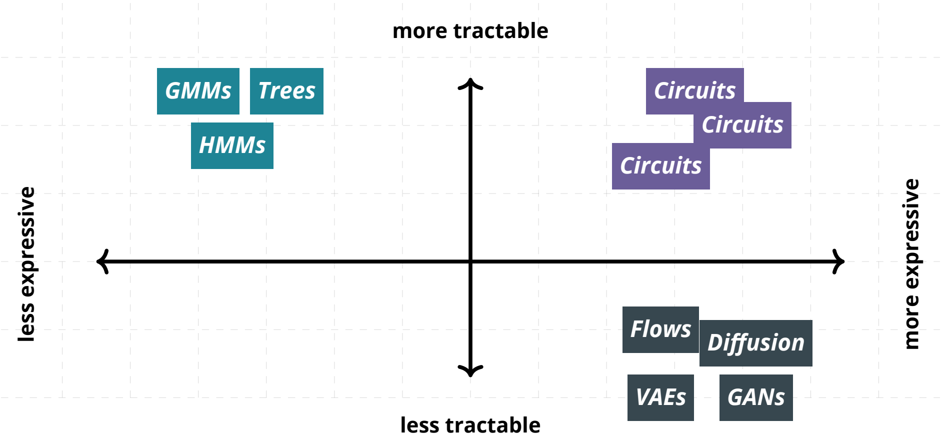

more tractable
less expressive
more expressive
less tractable



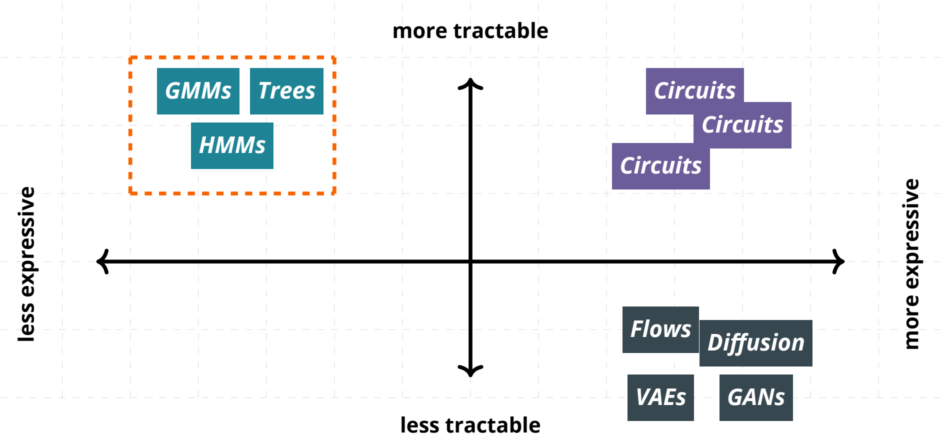
expressive models are not much tractable...



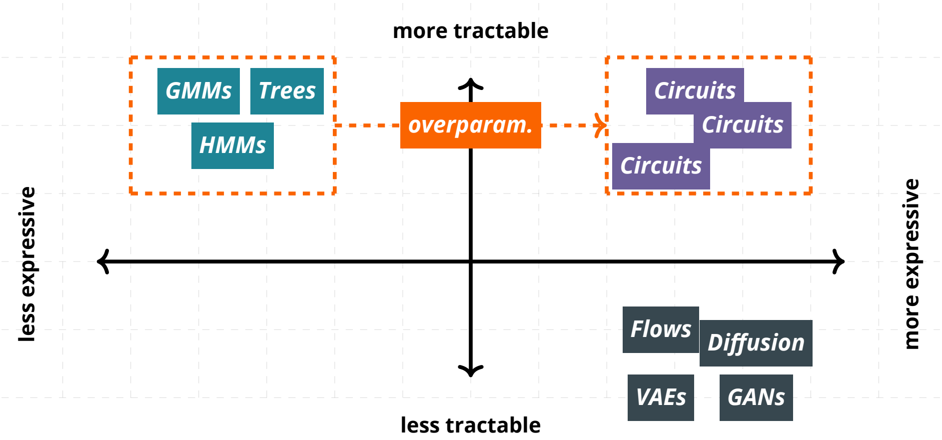
tractable models are not that expressive...



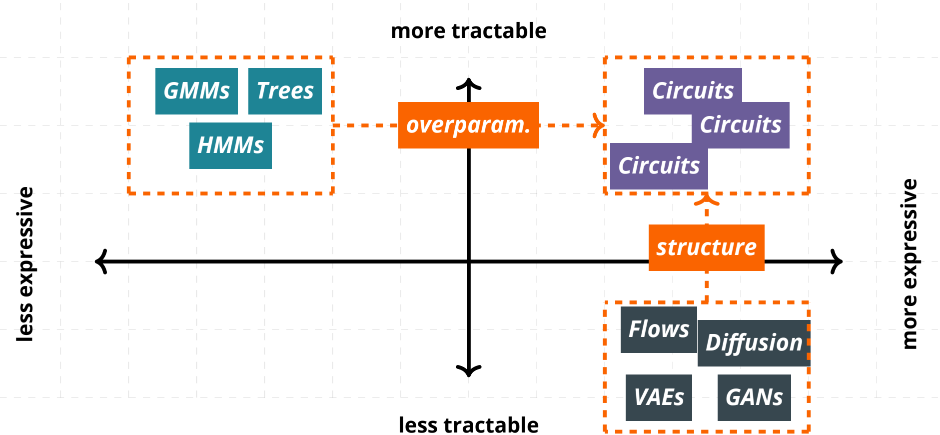
circuits can be both expressive and tractable!



start simple...



then make it more expressive!



impose structure!

Goal

"Can we design
computational graphs
that efficiently encode inference?"

Goal

"Can we design *computational graphs* that efficiently *encode inference*?"

$\Rightarrow$ **yes! with *circuits*!**

Probabilistic Circuits (PCs)

A grammar for tractable computational graphs

I. *A simple tractable function is a circuit*

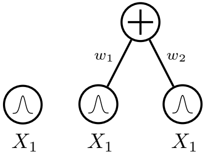
X_1

Probabilistic Circuits (PCs)

A grammar for tractable computational graphs

I. *A simple tractable function is a circuit*

II. *A weighted combination of circuits is a circuit*



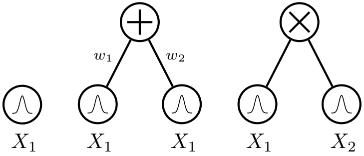
Probabilistic Circuits (PCs)

A grammar for tractable computational graphs

I. *A simple tractable function is a circuit*

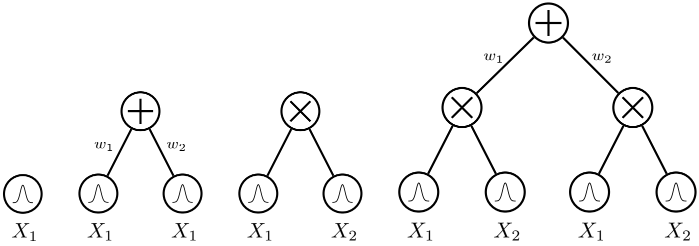
II. *A weighted combination of circuits is a circuit*

III. *A product of circuits is a circuit*



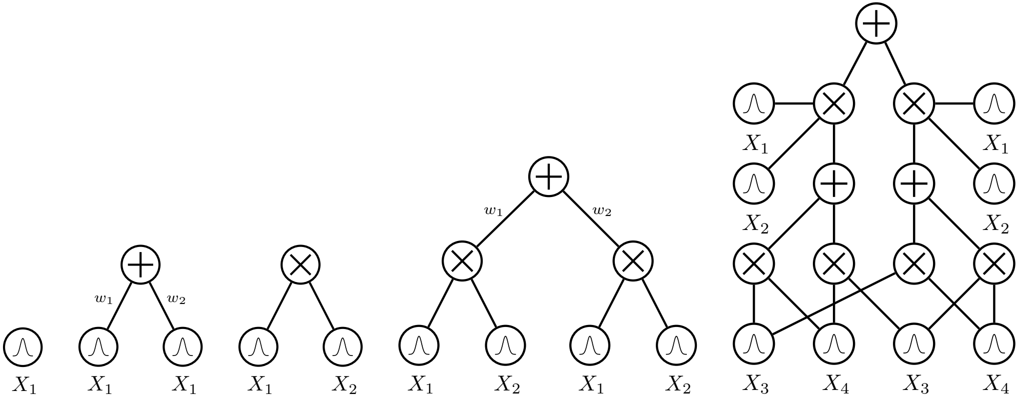
Probabilistic Circuits (PCs)

A grammar for tractable computational graphs



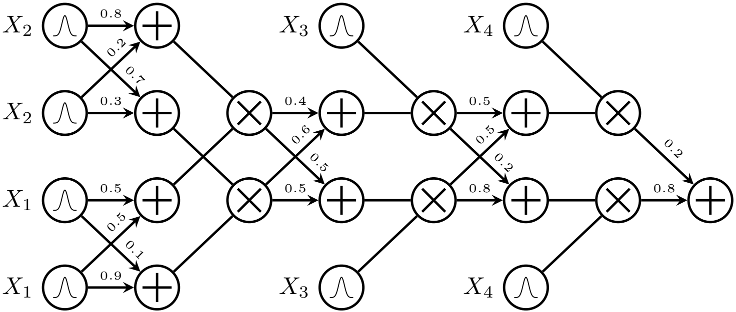
Probabilistic Circuits (PCs)

A grammar for tractable computational graphs



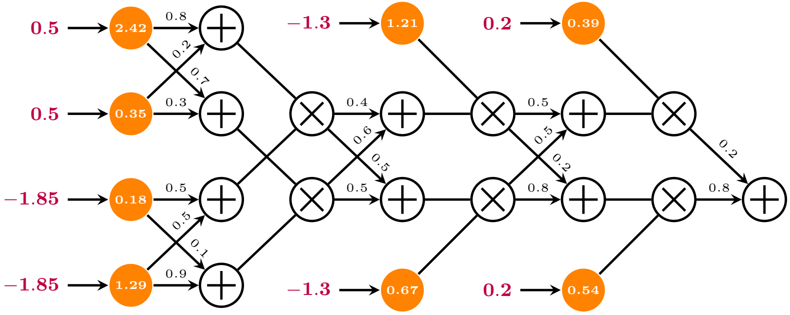
Probabilistic queries = feedforward evaluation

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



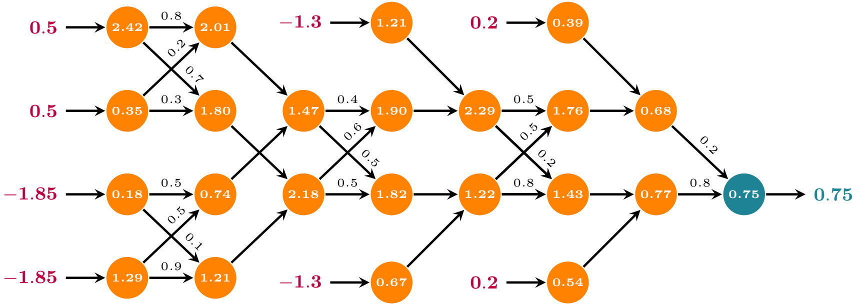
Probabilistic queries = feedforward evaluation

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



Probabilistic queries = *feedforward* evaluation

$$p(X_1 = -1.85, X_2 = 0.5, X_3 = -1.3, X_4 = 0.2) = 0.75$$



...why PCs?

1. A grammar for tractable models

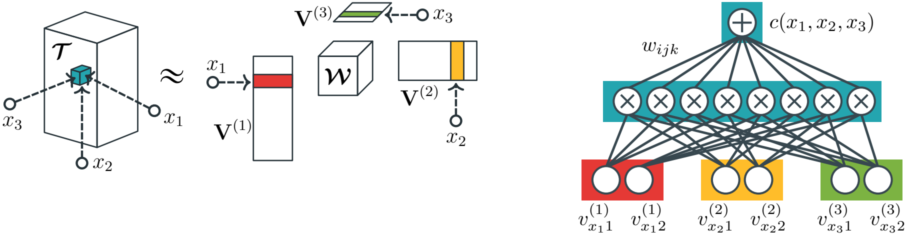
One formalism to represent many probabilistic and logical models

$\Longrightarrow$ *#HMMs #Trees #XGBoost, Tensor Networks, ...*

and other *PGMs...*

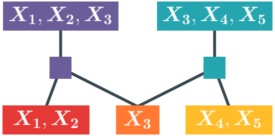
tensor factorizations

as circuits



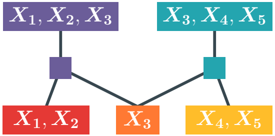
Loconte et al., What is the Relationship between Tensor Factorizations and Circuits (and How Can We Exploit it)?, , 2024

Learning recipe

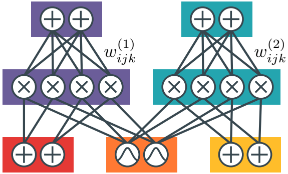


1) Build a ***region graph***

Learning recipe



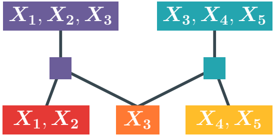
1) Build a ***region graph***

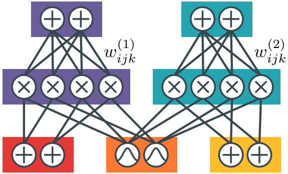


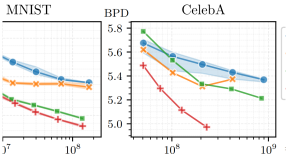
2) Overparameterize

2.1) pick a (composite) layer type
2.2) choose how many units per layer

Learning recipe







1) Build a ***region graph***

2) Overparameterize

3) Learn parameters

cirkit

learning & reasoning with circuits in pytorch

...why PCs?

1. A grammar for tractable models

One formalism to represent many probabilistic and logical models

$\Longrightarrow$ *#HMMs #Trees #XGBoost, Tensor Networks, ...*

and other *PGMs...*

...why PCs?

1. A grammar for tractable models

One formalism to represent many probabilistic and logical models

$\Longrightarrow$ *#HMMs #Trees #XGBoost, Tensor Networks, ...*

and other *PGMs...*

2. Expressiveness

Competitive with intractable models, VAEs, Flow...*#hierachical #mixtures #polynomials*

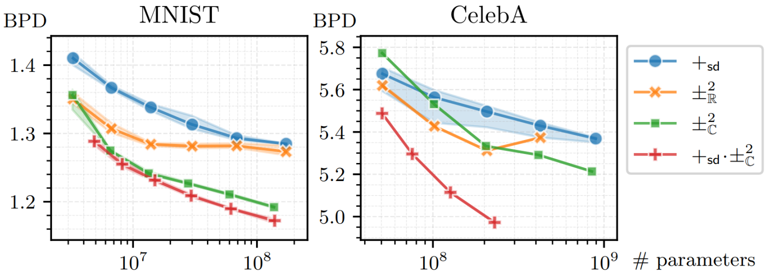
How expressive?

	QPC	PC	Sp-PC	HCLT	RAT	IDF	BitS	BBans	McB
MNIST	**1.11**	1.17	1.14	1.21	1.67	1.90	1.27	1.39	1.98
F-MNIST	**3.16**	3.32	3.27	3.34	4.29	3.47	3.28	3.66	3.72
EMN-MN	1.55	1.64	**1.52**	1.70	2.56	2.07	1.88	2.04	2.19
EMN-LE	**1.54**	1.62	1.58	1.75	2.73	1.95	1.84	2.26	3.12
EMN-BA	**1.59**	1.66	1.60	1.78	2.78	2.15	1.96	2.23	2.88
EMN-BY	1.53	**1.47**	1.54	1.73	2.72	1.98	1.87	2.23	3.14

competitive with Flows and VAEs!

Gala et al., Scaling Continuous Latent Variable Models as Probabilistic Integral Circuits, , 2024

How scalable?



up to billions of parameters

Loconte, Mengel, and Vergari, "Sum of Squares Circuits", arXiv, 2024

...why PCs?

1. A grammar for tractable models

One formalism to represent many probabilistic and logical models

$\Longrightarrow$ *#HMMs #Trees #XGBoost, Tensor Networks, ...*

and other *PGMs...*

2. Expressiveness

Competitive with intractable models, VAEs, Flow...*#hierachical #mixtures #polynomials*

3. Tractability == Structural Properties!!!

Exact computations of reasoning tasks are certified by guaranteeing certain structural properties. *#marginals #expectations #MAP, #product ...*

Structural properties

smoothness

decomposability

determinism

compatibility

Vergari *et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021*

Structural properties

property A

property B

property C

property D

__Vergari__ et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021

Structural properties

property A

property B

property C

property D

tractable computation of ***arbitrary integrals***

$$p(\mathbf{y}) = \int p(\mathbf{z}, \mathbf{y})\, d\mathbf{Z}, \quad \forall \mathbf{Y} \subseteq \mathbf{X}, \quad \mathbf{Z} = \mathbf{X} \setminus \mathbf{Y}$$

$\implies$ ***sufficient*** *and* ***necessary*** *conditions for a single feedforward evaluation*

$\implies$ *tractable partition function*

Vergari *et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021*

Structural properties

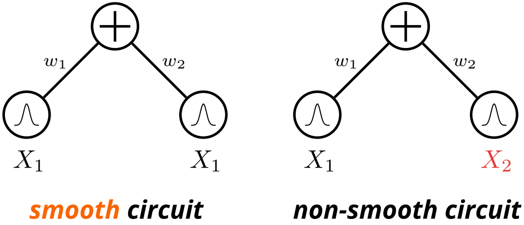
smoothness

decomposability

compatibility

determinism

the inputs of sum units are defined over the same variables



__Vergari__ et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021

Structural properties

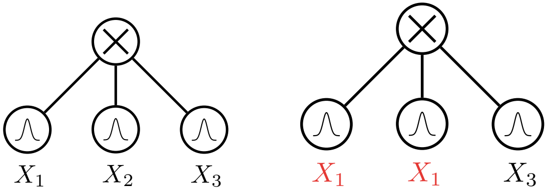
smoothness

decomposability

compatibility

determinism

the inputs of prod units are defined over disjoint variable sets

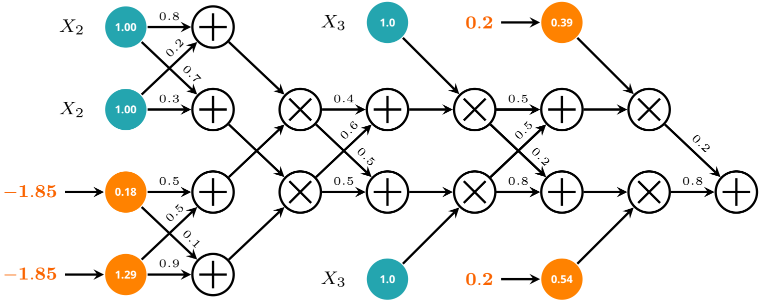


decomposable* circuit** ***non-decomposable circuit

Vergari *et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021*

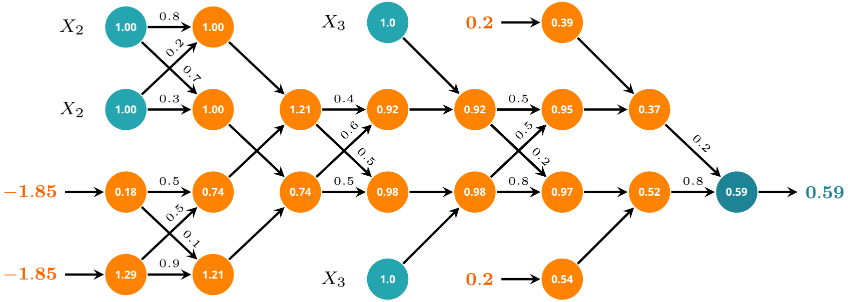
Probabilistic queries = feedforward evaluation

$$p(X_1 = -1.85, X_4 = 0.2)$$



Probabilistic queries = feedforward evaluation

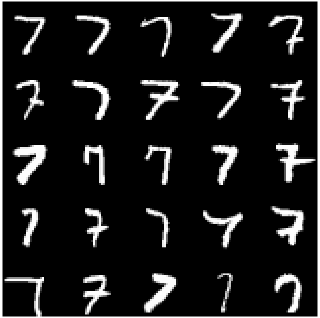
$$p(X_1 = -1.85, X_4 = 0.2)$$



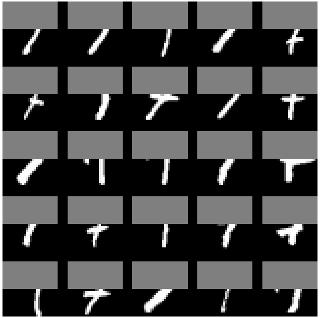
Tractable inference on PCs

Einsum networks

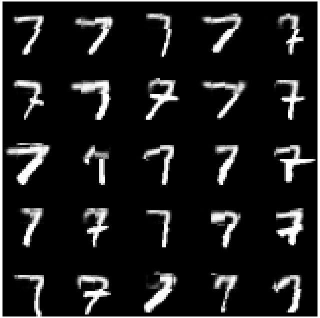
Original



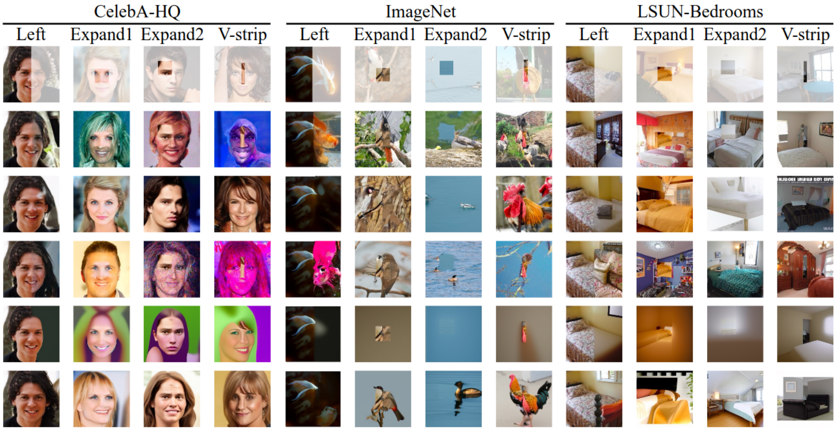
Missing



Conditional sample



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


CelebA-HQ
ImageNet
LSUN-Bedrooms
Left
Expand1
Expand2
V-strip
Left
Expand1
Expand2
V-strip
Left
Expand1
Expand2
V-strip

General expectations

Integrals involving two or more functions:

$$\int \boldsymbol{p}(\mathbf{x})\,\boldsymbol{f}(\mathbf{x})\,d\,\mathbf{X}$$

General expectations

Integrals involving two or more functions:

$$\int p(\mathbf{x}) f(\mathbf{x}) d\mathbf{X}$$

represent both p and f as circuits...but with which structural properties? E.g.,

General expectations

Integrals involving two or more functions:

$$\int p(\mathbf{x}) f(\mathbf{x}) d\mathbf{X}$$

represent both p and f as circuits...but with which structural properties? E.g.,

$$\mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 0)} \left[f_0(\mathbf{x}_c) \right] - \mathbb{E}_{\mathbf{x}_c \sim p(\mathbf{X}_c | X_s = 1)} \left[f_1(\mathbf{x}_c) \right]$$

Structural properties

smoothness

decomposability

compatibility

determinism

Vergari *et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021*

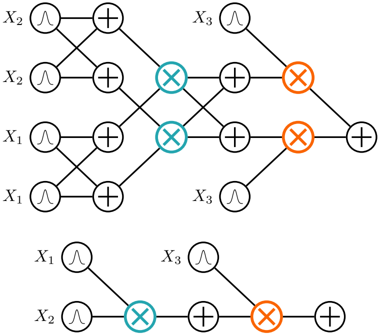
Structural properties

smoothness

decomposability

compatibility

determinism



compatible circuits

Vergari *et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021*

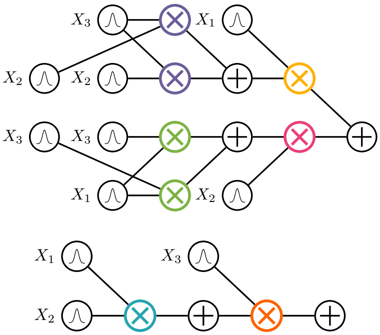
Structural properties

smoothness

decomposability

compatibility

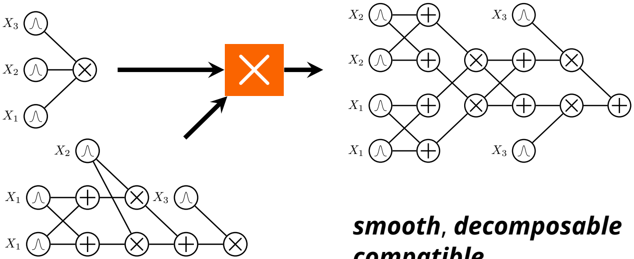
determinism



non-compatible circuits

__Vergari__ et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021

Tractable products



smooth**, **decomposable
compatible

exactly compute $\int \boldsymbol{p}(\mathbf{x})\boldsymbol{f}(\mathbf{x})dX$ **in time** $O(|\boldsymbol{p}||\boldsymbol{f}|)$

Vergari *et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021*

Semantic Probabilistic Layers for Neuro-Symbolic Learning

Kareem Ahmed
CS Department
UCLA
ahmedk@cs.ucla.edu

Stefano Teso
CIMeC and DISI
University of Trento
stefano.teso@unitn.it

Kai-Wei Chang
CS Department
UCLA
kwchang@cs.ucla.edu

Guy Van den Broeck
CS Department
UCLA
guyvdb@cs.ucla.edu

Antonio Vergari
School of Informatics
University of Edinburgh
avergari@ed.ac.uk

circuit products for reliable NeSy

When?

Ground Truth

e.g. predict shortest path in a map

When?

Ground Truth

given $\mathbf{x}$ *// e.g. a tile map*

nesy structured output prediction (SOP) tasks

Vlastelica et al., "Differentiation of blackbox combinatorial solvers", , 2020

When?

Ground Truth

given $\mathbf{x}$ *// e.g. a tile map*

find $\mathbf{y}^* = \operatorname{argmax}_{\mathbf{y}} p_\theta(\mathbf{y} \mid \mathbf{x})$ *// e.g. a configurations of edges in a grid*

nesy structured output prediction (SOP) tasks

Vlastelica et al., "Differentiation of blackbox combinatorial solvers", , 2020

When?

Ground Truth

given $\mathbf{x}$ *// e.g. a tile map*

find $\mathbf{y}^* = \text{argmax}_{\mathbf{y}}\, p_\theta(\mathbf{y} \mid \mathbf{x})$ *// e.g. a configurations of edges in a grid*

s.t. $\mathbf{y} \models \mathsf{K}$ *// e.g., that form a valid path*

nesy structured output prediction (SOP) tasks

Vlastelica et al., "Differentiation of blackbox combinatorial solvers", , 2020

When?

Ground Truth

given $\mathbf{x}$ *// e.g. a tile map*

find $\mathbf{y}^* = \text{argmax}_{\mathbf{y}}\, p_\theta(\mathbf{y} \mid \mathbf{x})$ *// e.g. a configurations of edges in a grid*

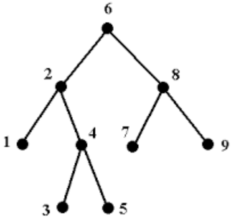
s.t. $\mathbf{y} \models \mathsf{K}$ *// e.g., that form a valid path*

// for a 12×12 grid, 2^{144} states but only 10^{10} valid ones!

nesy structured output prediction (SOP) tasks

Vlastelica et al., "Differentiation of blackbox combinatorial solvers", , 2020

When?



given $\mathbf{x}$ *// e.g. a feature map*

find $\mathbf{y}^* = \text{argmax}_{\mathbf{y}}\, p_\theta(\mathbf{y} \mid \mathbf{x})$ *// e.g. labels of classes*

s.t. $\mathbf{y} \models \mathsf{K}$ *// e.g., constraints over superclasses*

$$\mathsf{K} : (Y_{\text{cat}} \implies Y_{\text{animal}}) \wedge (Y_{\text{dog}} \implies Y_{\text{animal}})$$

hierarchical multi-label classification

Giunchiglia and Lukasiewicz, "Coherent hierarchical multi-label classification networks", , 2020

When?

given $\mathbf{x}$ *// e.g. a user preference over $K - N$ sushi types*

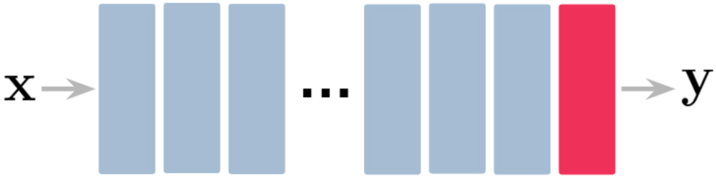
find $\mathbf{y}^* = \text{argmax}_{\mathbf{y}}\, p_\theta(\mathbf{y} \mid \mathbf{x})$ *// e.g. prefs over N more types*

s.t. $\mathbf{y} \models \mathsf{K}$ *// e.g., output valid rankings*

user preference learning

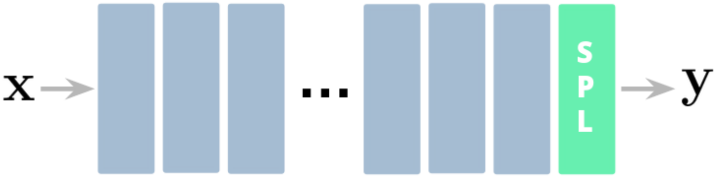
Choi, Van den Broeck, and Darwiche, "Tractable learning for structured probability spaces: A case study in learning preference distributions", Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), 2015

How?



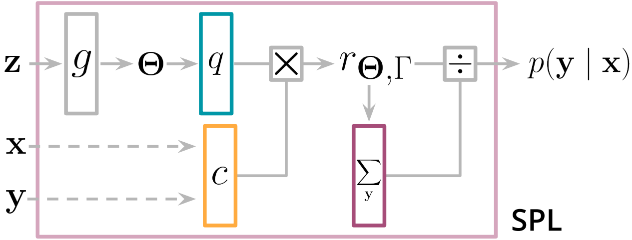
take an unreliable neural network architecture...

How?

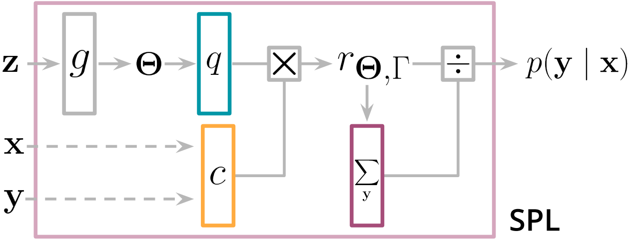


......and replace the last layer with a semantic probabilistic layer

SPL



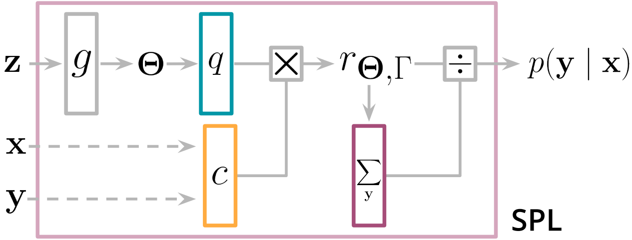
SPL



$$p(\mathbf{y} \mid \mathbf{x}) = q_{\Theta}(\mathbf{y} \mid g(\mathbf{z}))$$

$q_{\Theta}(\mathbf{y} \mid g(\mathbf{z}))$ is an expressive distribution over labels

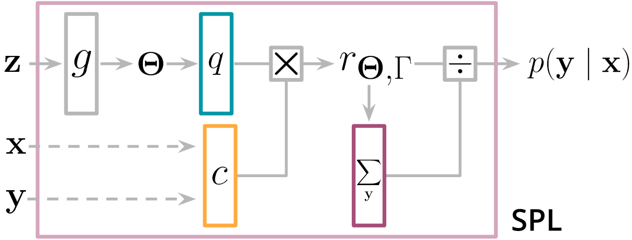
SPL



$$p(\mathbf{y} \mid \mathbf{x}) = q_{\boldsymbol{\Theta}}(\mathbf{y} \mid g(\mathbf{z})) \cdot c_{\mathsf{K}}(\mathbf{x}, \mathbf{y})$$

$c_{\mathsf{K}}(\mathbf{x}, \mathbf{y})$ encodes the constraint $\mathbb{1}\{\mathbf{x}, \mathbf{y} \models \mathsf{K}\}$

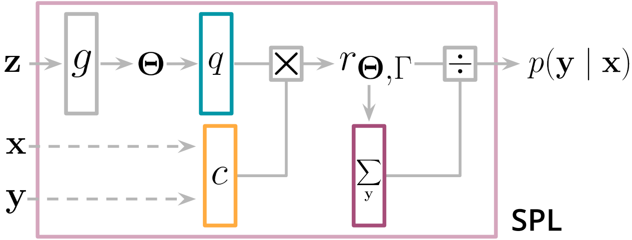
SPL



$$p(\mathbf{y} \mid \mathbf{x}) = q_{\Theta}(\mathbf{y} \mid g(\mathbf{z})) \cdot c_{\mathsf{K}}(\mathbf{x}, \mathbf{y})$$

a product of experts : (

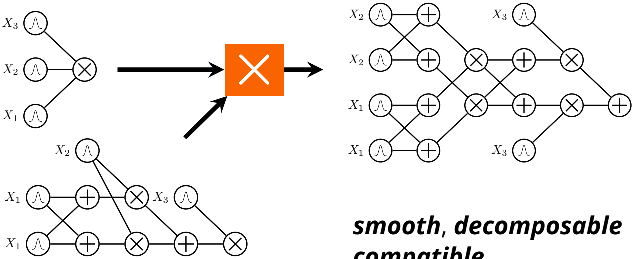
SPL



$$p(\mathbf{y} \mid \mathbf{x}) = q_{\Theta}(\mathbf{y} \mid g(\mathbf{z})) \cdot c_{\mathsf{K}}(\mathbf{x}, \mathbf{y}) / \mathcal{Z}(\mathbf{x})$$

$$\mathcal{Z}(\mathbf{x}) = \sum\nolimits_{\mathbf{y}} q_{\Theta}(\mathbf{y} \mid \mathbf{x}) \cdot c_{\mathsf{K}}(\mathbf{x}, \mathbf{y})$$

Tractable products



smooth, ***decomposable***
compatible

exactly compute $\mathcal{Z}$ in time $O(|\boldsymbol{q}||\boldsymbol{c}|)$

__Vergari__ et al., "A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference", NeurIPS, 2021

How to Turn Your Knowledge Graph Embeddings into Generative Models

Lorenzo Loconte
University of Edinburgh, UK
l.loconte@sms.ed.ac.uk

Nicola Di Mauro
University of Bari, Italy
nicola.dimauro@uniba.it

Robert Peharz
TU Graz, Austria
robert.peharz@tugraz.at

Antonio Vergari
University of Edinburgh, UK
avergari@ed.ac.uk

PCs meet knowledge graph embedding models
oral at NeurIPS 2023

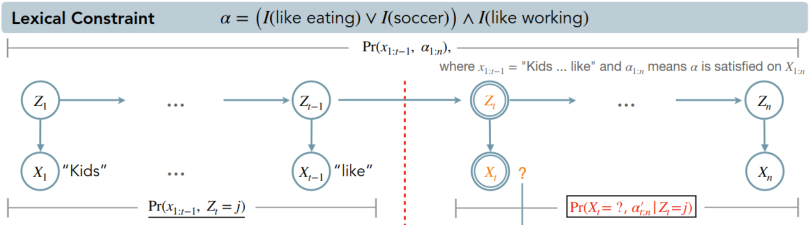
SPLs

(and more circuits)

everywhere

Tractable Control for Autoregressive Language Generation

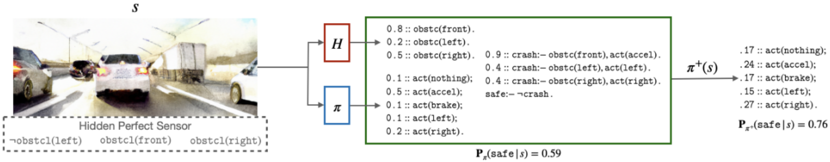
Honghua Zhang [*1] Meihua Dang [*1] Nanyun Peng [1] Guy Van den Broeck [1]



constrained text generation with LLMs (ICML 2023)

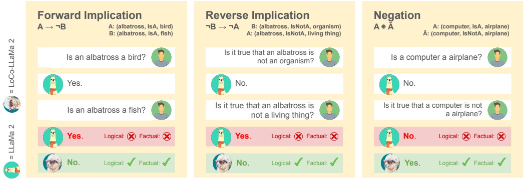
Safe Reinforcement Learning via Probabilistic Logic Shields

Wen-Chi Yang[1], Giuseppe Marra[1], Gavin Rens and Luc De Raedt[1,2]



reliable reinforcement learning (AAAI 23)

Logically Consistent Language Models via Neuro-Symbolic Integration



improving logical (self-)consistency in LLMs (under submission)

How to Turn Your Knowledge Graph Embeddings into Generative Models

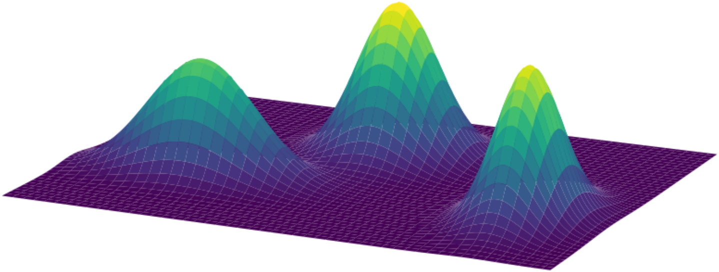
Lorenzo Loconte
University of Edinburgh, UK
l.loconte@sms.ed.ac.uk

Nicola Di Mauro
University of Bari, Italy
nicola.dimauro@uniba.it

Robert Peharz
TU Graz, Austria
robert.peharz@tugraz.at

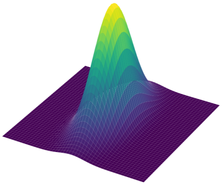
Antonio Vergari
University of Edinburgh, UK
avergari@ed.ac.uk

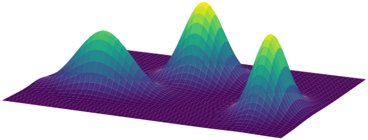
PCs meet knowledge graph embedding models
oral at NeurIPS 2023



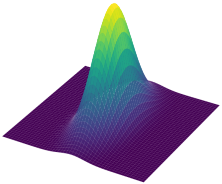
oh mixtures, you're so fine you blow my mind!

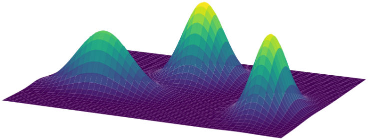
image taken from Hao Tang's course on ASR





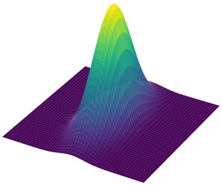
$$p(\mathbf{X}) \rightarrow \sum_{i=1}^{K} p_i(\mathbf{X})$$

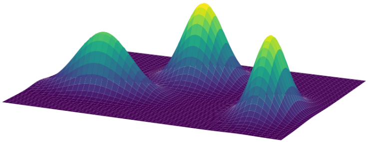


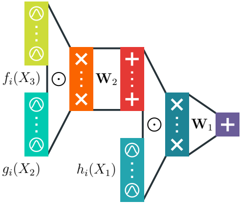


$$p(\mathbf{X}) \rightarrow \sum_{i=1}^{K} p_i(\mathbf{X})$$

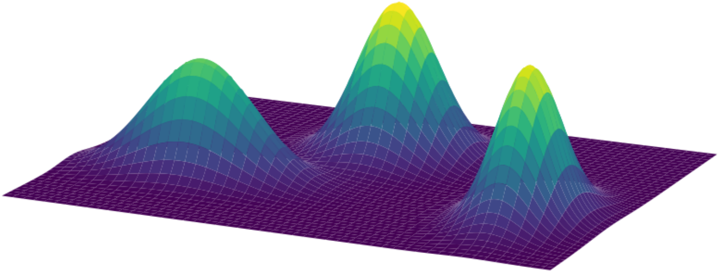
"if someone publishes a paper on model A, there will be a paper about mixtures of A soon with high probability" **A. Vergari**





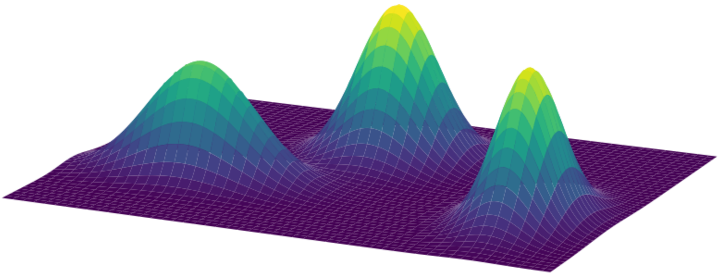


$$p(\mathbf{X}) \rightarrow \sum_{i=1}^{K} p_i(\mathbf{X}) \rightarrow \sum_{i=1}^{2^D} p_i(\mathbf{X}) = \mathsf{PC}(\mathbf{X})$$



$$c(\mathbf{X}) = \sum_{i=1}^{K} w_i c_i(\mathbf{X}), \quad \text{with} \quad w_i \geq 0, \quad \sum_{i=1}^{K} w_i = 1$$

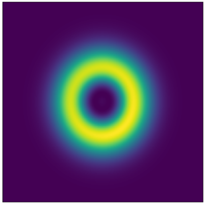
image taken from Hao Tang's course on ASR



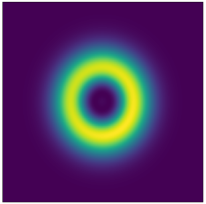
$$c(\mathbf{X}) = \sum_{i=1}^{K} w_i c_i(\mathbf{X}), \quad \text{with} \quad w_i \geq 0, \quad \sum_{i=1}^{K} w_i = 1$$

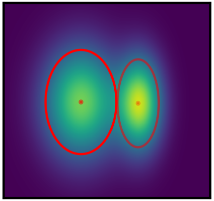
image taken from Hao Tang's course on ASR

however...



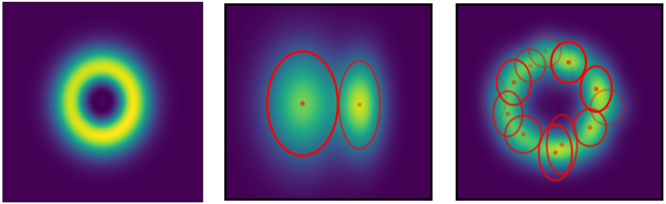
however...





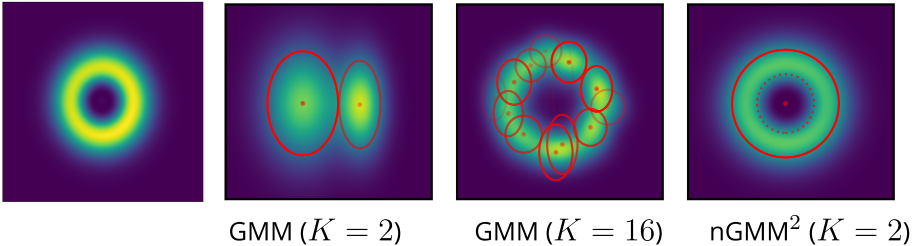
GMM ($K = 2$)

however...



GMM ($K = 2$) GMM ($K = 16$)

however...

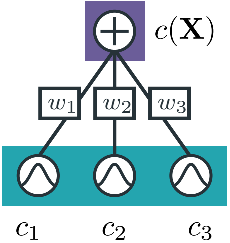


theorem

shallow mixtures
with negative parameters
can be *exponentially more compact* than
deep ones with positive ones.

Loconte et al., "Subtractive Mixture Models via Squaring: Representation and Learning", ICLR, 2024

subtractive MMs as circuits

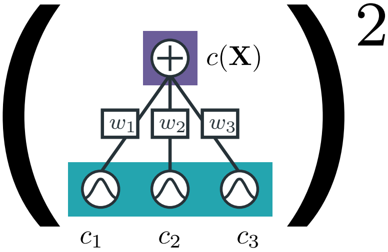


a **non-monotonic** smooth and (structured) decomposable circuit

$\Rightarrow$ *possibly with negative outputs*

$$c(\mathbf{X}) = \sum_{i=1}^{K} w_i c_i(\mathbf{X}), \qquad w_i \in \mathbb{R},$$

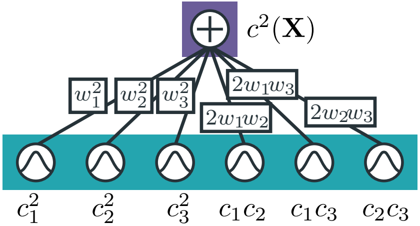
squaring shallow MMs



$$c^2(\mathbf{X}) = \left(\sum_{i=1}^{K} w_i c_i(\mathbf{X})\right)^2$$

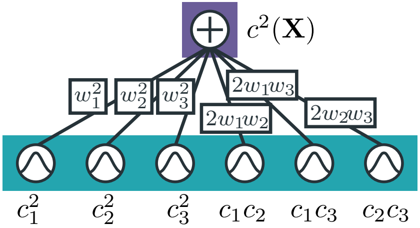
$\Longrightarrow$ *ensure non-negative output*

squaring shallow MMs



$$c^2(\mathbf{X}) = \left(\sum_{i=1}^{K} w_i c_i(\mathbf{X})\right)^2 = \sum_{i=1}^{K}\sum_{j=1}^{K} w_i w_j c_i(\mathbf{X}) c_j(\mathbf{X})$$

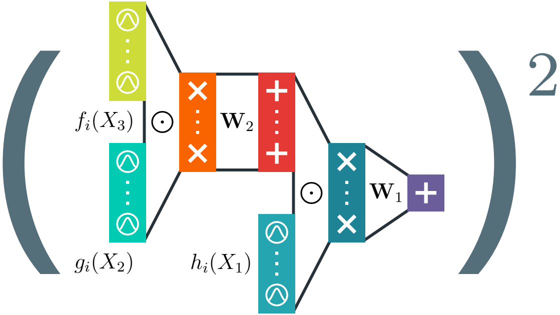
squaring shallow MMs



$$c^2(\mathbf{X}) = \left(\sum_{i=1}^{K} w_i c_i(\mathbf{X})\right)^2$$
$$= \sum_{i=1}^{K} \sum_{j=1}^{K} w_i w_j c_i(\mathbf{X}) c_j(\mathbf{X})$$

still a smooth and (str) decomposable PC with $\mathcal{O}(K^2)$ components!

$\Rightarrow$ *but still* $\mathcal{O}(K)$ *parameters*

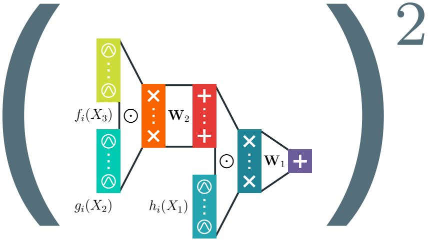


how to efficiently square (and renormalize) a deep PC?

Loconte et al., "Subtractive Mixture Models via Squaring: Representation and Learning", ICLR, 2024

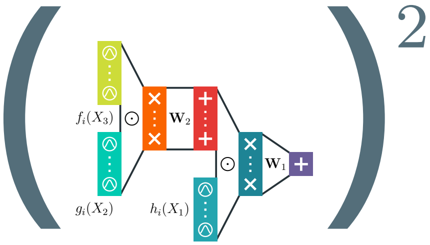
squaring deep PCs

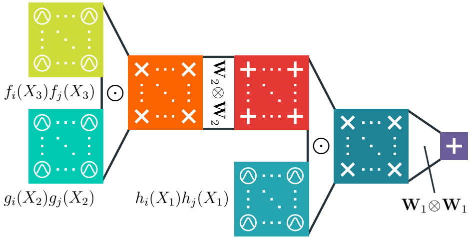
the tensorized way



squaring deep PCs

the tensorized way

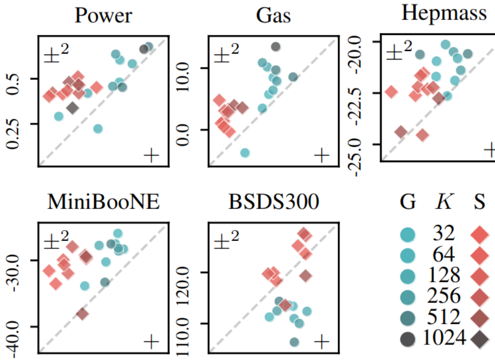


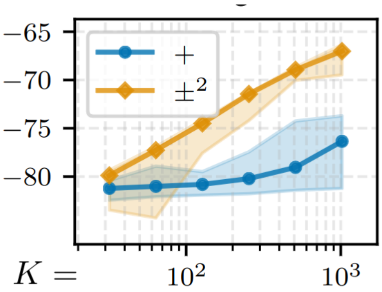


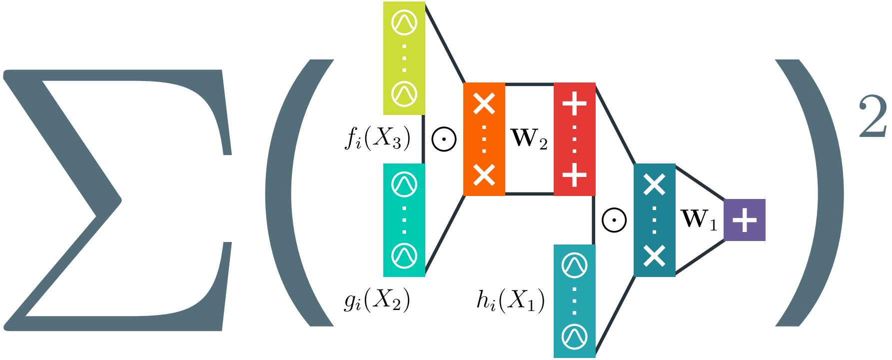
squaring a circuit = to squaring layers

how more expressive?

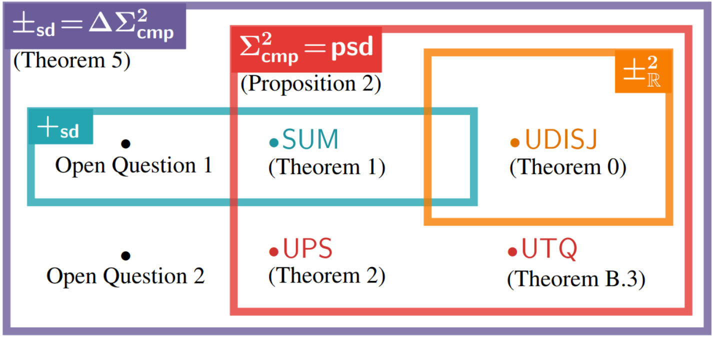
for the ML crowd





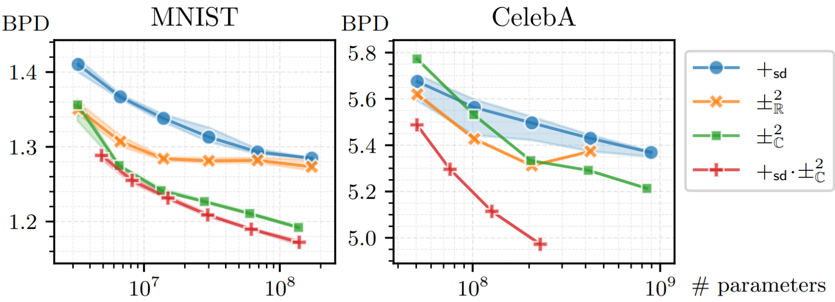


more that a single square?



SOS circuits are more expressive

Loconte, Mengel, and Vergari, "Sum of Squares Circuits", arXiv, 2024

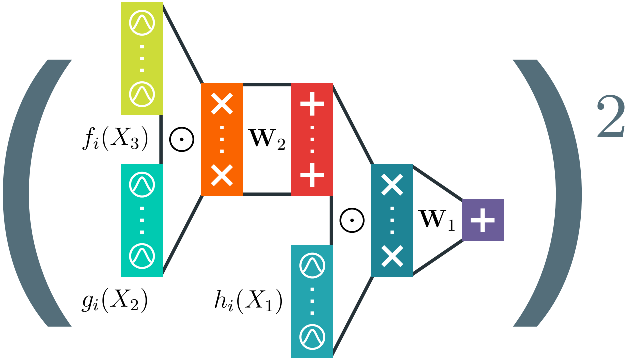


complex circuits are SOS (and scale better!)

Loconte, Mengel, and Vergari, “Sum of Squares Circuits”, arXiv, 2024

cirkit

learning & reasoning with circuits in pytorch



questions?