UCLA Computer Science

STAR AI
RESEARCH LAB
UCLA

AI can learn from data. But can it learn to reason?

Guy Van den Broeck

Oregon State University - Feb 3 2023

Outline

1. The paradox of learning to reason from data

   ~~deep learning~~

2. Architectures for Learning and Reasoning

   logical reasoning + *deep learning*

   a. Constrained language generation
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

Outline

1. **The paradox of learning to reason from data**

   ~~deep learning~~

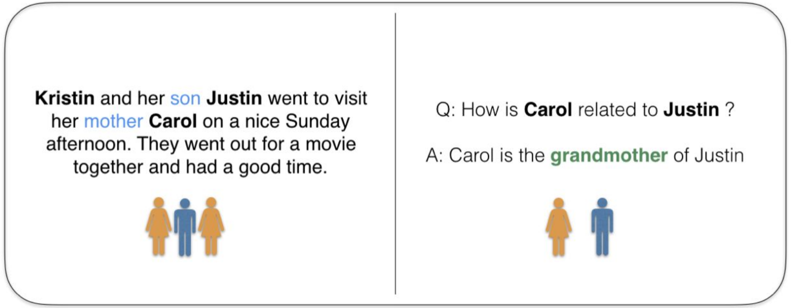
2. Architectures for Learning and Reasoning

   logical reasoning + *deep learning*

   a. Constrained language generation
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

Can Language Models Perform Logical Reasoning?

Language Models achieve high performance on various “reasoning” benchmarks in NLP.



Reasoning Example from the CLUTRR dataset

It is unclear whether they solve the tasks following the rules of logical deduction.

<u>Language Models</u>:
input → ? → Carol is the grandmother of Justin.

<u>Logical Reasoning</u>:
input → Justin in Kristin’s son; Carol is Kristin’s mother; → Carol is Justin’s mother’s mother; if X is Y’s mother’s mother then X is Y’s grandmother → Carol is the grandmother of Justin.

SimpleLogic

Generate textual train and test examples of the form:

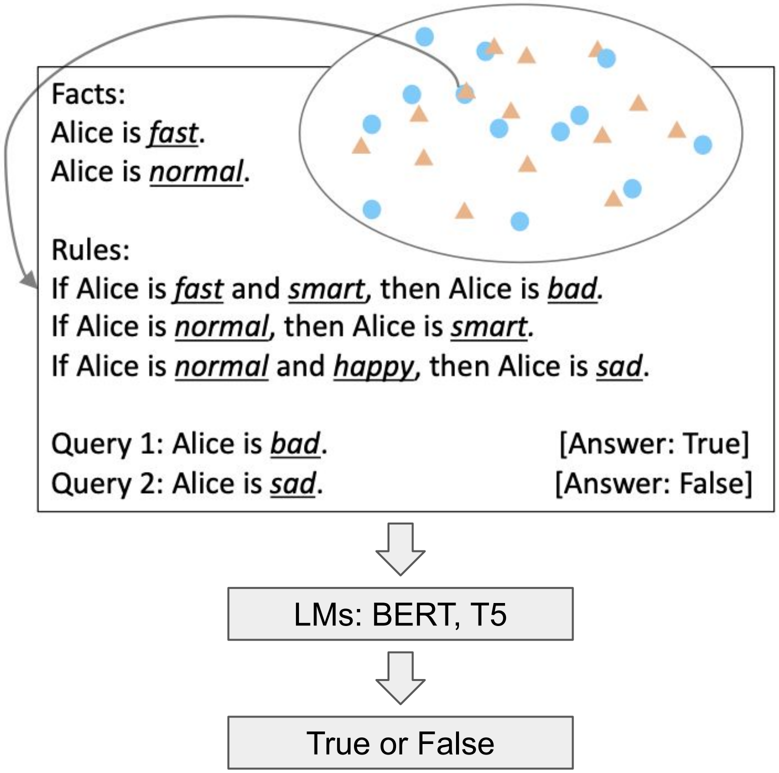
Rules: If witty, then diplomatic. If careless and condemned and attractive, then blushing. If dishonest and inquisitive and average, then shy. If average, then stormy. If popular, then blushing. If talented, then hurt. If popular and attractive, then thoughtless. If blushing and shy and stormy, then inquisitive. If adorable, then popular. If cooperative and wrong and stormy, then thoughtless. If popular, then sensible. If cooperative, then wrong. If shy and cooperative, then witty. If polite and shy and thoughtless, then talented. If polite, then condemned. If polite and wrong, then inquisitive. If dishonest and inquisitive, then talented. If blushing and dishonest, then careless. If inquisitive and dishonest, then troubled. If blushing and stormy, then shy. If diplomatic and talented, then careless. If wrong and beautiful, then popular. If ugly and shy and beautiful, then stormy. If shy and inquisitive and attractive, then diplomatic. If witty and beautiful and frightened, then adorable. If diplomatic and cooperative, then sensible. If thoughtless and inquisitive, then diplomatic. If careless and dishonest and troubled, then cooperative. If hurt and witty and troubled, then dishonest. If scared and diplomatic and troubled, then average. If ugly and wrong and careless, then average. If dishonest and scared, then polite. If talented, then dishonest. If condemned, then wrong. If wrong and troubled and blushing, then scared. If attractive and condemned, then frightened. If hurt and condemned and shy, then witty. If cooperative, then attractive. If careless, then polite. If adorable and wrong and careless, then diplomatic. Facts: Alice sensible Alice condemned Alice thoughtless Alice polite Alice scared Alice average
Query: Alice is shy ?

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Problem Setting: SimpleLogic

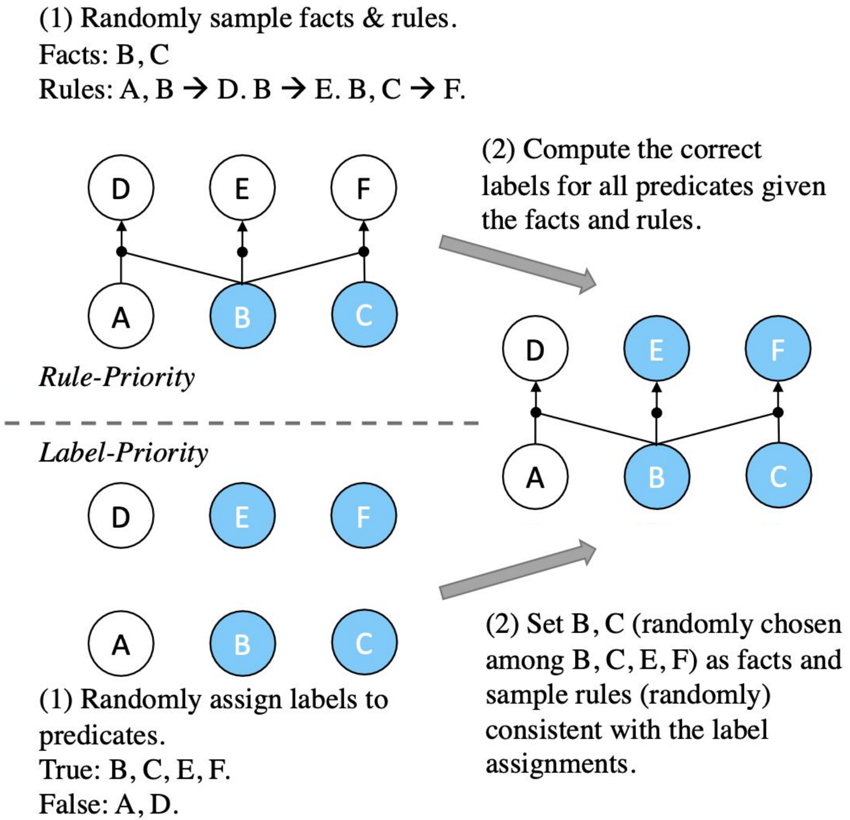
The easiest of reasoning problems:

1. **Propositional logic** fragment
   a. bounded vocabulary & number of rules
   b. bounded reasoning depth (≤ 6)
   c. finite space (≈ 10^360)
2. **No language variance**: templated language
3. **Self-contained**
   No prior knowledge
4. **Purely symbolic** predicates
   No shortcuts from word meaning
5. **Tractable** logic (definite clauses)
   Can always be solved efficiently



Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Training a BERT model on SimpleLogic



Test accuracy for different reasoning depths

Test	0	1	2	3	4	5	6
RP	99.9	99.8	99.7	99.3	98.3	97.5	95.5

Test	0	1	2	3	4	5	6
LP	100.0	100.0	99.9	99.9	99.7	99.7	99.0

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Has BERT learned to reason from data?

1. Easiest of reasoning problems (no variance, self-contained, purely symbolic, tractable)
2. RP/LP data covers the whole problem space
3. The learned model has almost 100% test accuracy
4. There exist BERT parameters that compute the ground-truth reasoning function:

<u>Theorem 1:</u> *For a BERT model with* n *layers and 12 attention heads, by construction, there exists a set of parameters such that the model can correctly solve any reasoning problem in SimpleLogic that requires at most* $n - 2$ *steps of reasoning.*

Surely, under these conditions,
BERT has learned the ground-truth 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, 2022

The Paradox of Learning to Reason from Data

Train	Test	0	1	2	3	4	5	6
RP	RP	99.9	99.8	99.7	99.3	98.3	97.5	95.5
	LP	99.8	99.8	99.3	96.0	90.4	75.0	57.3
LP	RP	97.3	66.9	53.0	54.2	59.5	65.6	69.2
	LP	100.0	100.0	99.9	99.9	99.7	99.7	99.0

The BERT model trained on one distribution fails to generalize to the other distribution within the same problem space.

1. If BERT **has learned** to reason,
   it should not exhibit such generalization failure.

2. If BERT **has not learned** to reason,
   it is baffling how it achieves near-perfect in-distribution test accuracy.

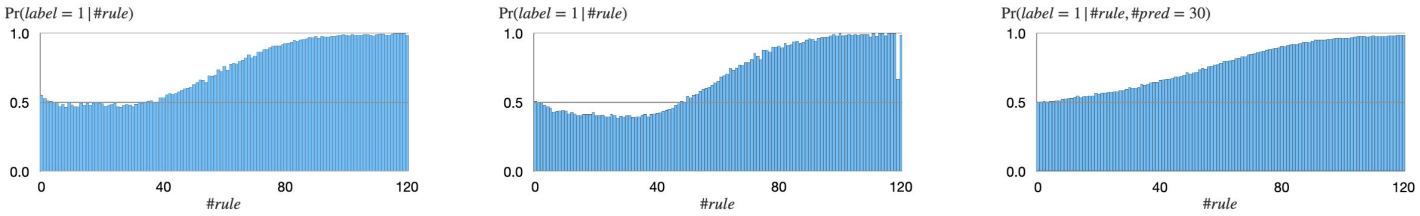
Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Why? Statistical Features

Monotonicity of entailment:
Any rules can be freely added to the hypothesis of any proven fact.

The more rules given, the more likely a predicate will be proved.

Pr(label = True | Rule # = x) should increase (roughly) monotonically with x



(a) Statistics for examples generated by Rule-Priority (RP). (b) Statistics for examples generated by Label-Priority (LP). (c) Statistics for examples generated by uniform sampling;

BERT leverages statistical features to make predictions

RP_b downsamples from RP such that Pr(label = True | rule# = x) = 0.5 for all x

Train	Test	0	1	2	3	4	5	6
RP	RP	99.9	99.8	99.7	99.3	98.3	97.5	95.5
	RP_b	99.0	99.3	98.5	97.5	96.7	93.5	88.3

1. Accuracy drop from RP to RP_b indicates that **the model is using rule# as a statistical feature to make predictions.**
2. Potentially countless statistical features
3. Such features are **inherent to the reasoning problem**, cannot make data "clean"

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

First Conclusion

Experiments unveil the fundamental difference between

1. learning to reason, and
2. learning to achieve high performance on benchmarks using statistical features.

Be careful deploying AI in applications where this difference matters.

Is more data going to solve this problem?

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Outline

1. The paradox of learning to reason from data

   ~~deep learning~~

2. **Architectures for Learning and Reasoning**

   logical reasoning + *deep learning*

   a. Constrained language generation
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

Outline

1. The paradox of learning to reason from data

   ~~deep learning~~

2. **Architectures for Learning and Reasoning**

   logical reasoning + *deep learning*

   a. **Constrained language generation**
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

Is ChatGPT all you need?

H generate a sentence using the following keywords: fish watch kid swim tank

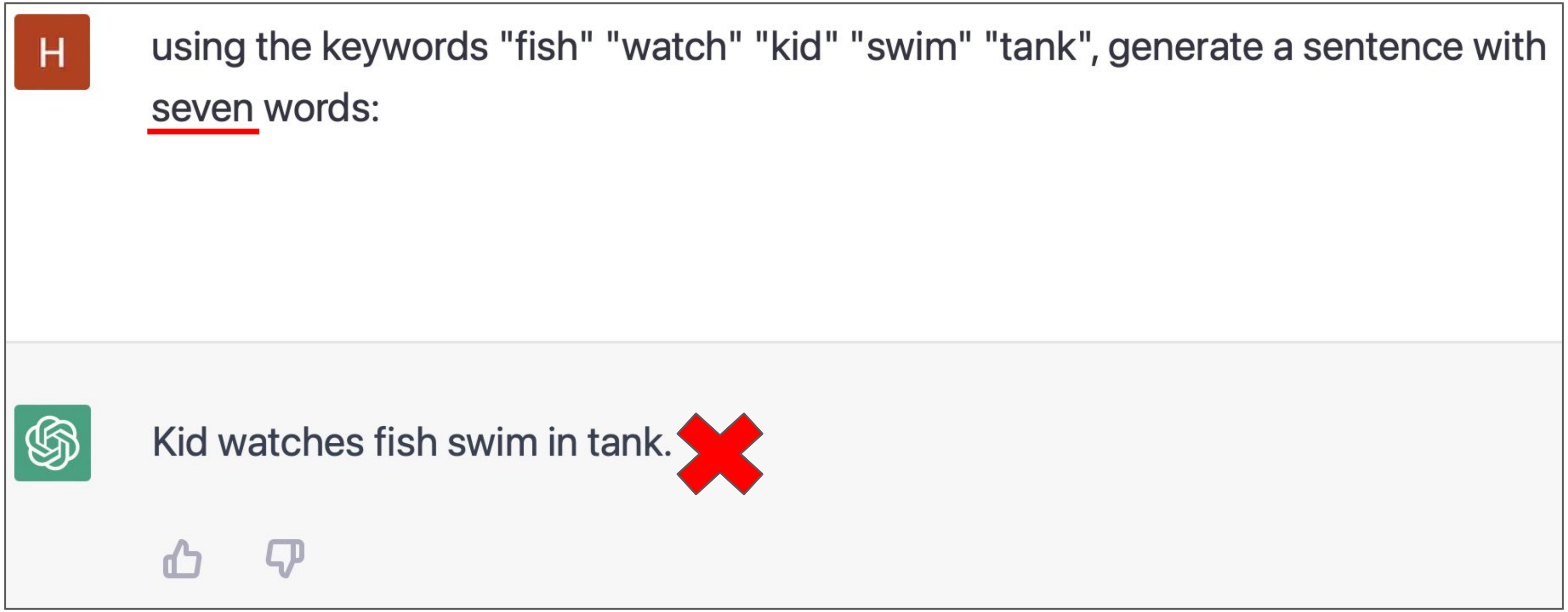
A kid watches a fish swim in the tank.

H generate a sentence using the following keywords: cook top pan stove vegetable

The cook used a pan on the stove top to prepare the vegetable.

Large language models are *not* all you need

fine-grained control (e.g., generation given lexical constraints) is challenging …



Large language models are *not* all you need

fine-grained control (e.g., generation given lexical constraints) is challenging …

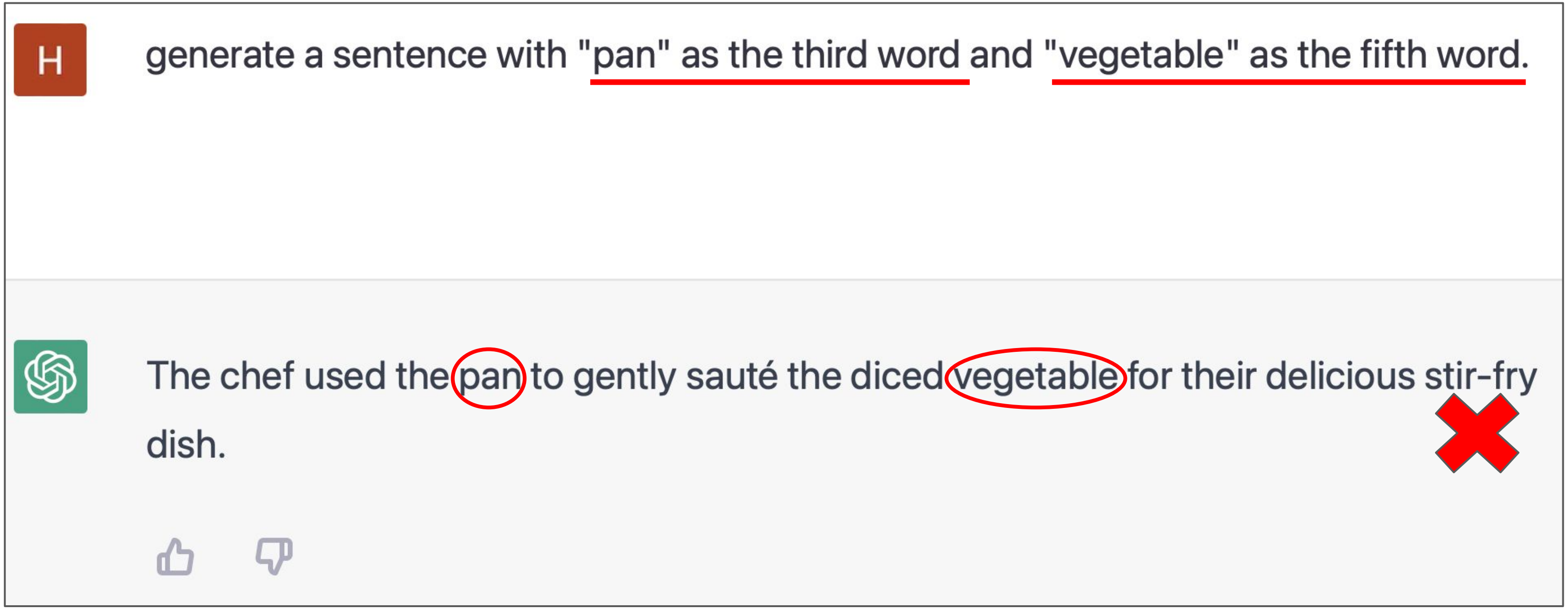
H

generate a sentence using the following keywords and the keywords should appear in the order as listed: pan vegetable cook stove top

"She placed a pan of chopped vegetables on the stove top to cook."

Large language models are *not* all you need

fine-grained control (e.g., generation given lexical constraints) is challenging …



Key Challenge: intractable conditioning

Given lexical constraint α:
"pan" "vegetable" "cook" "stove" "top" appears in the generated sentence

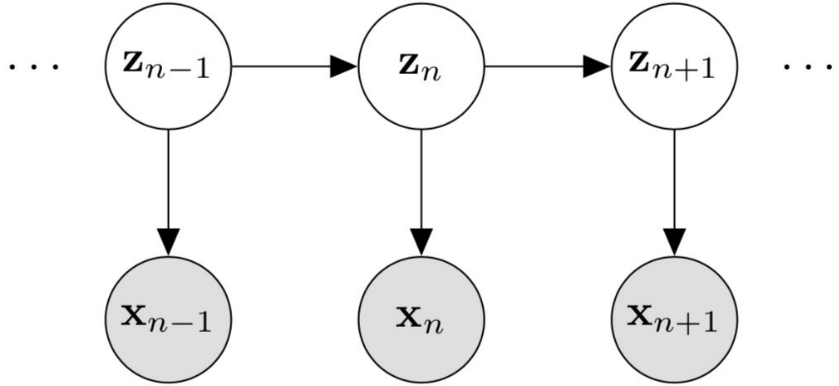
Goal: generate from the conditional distribution Pr(sentence | α)

Computing Pr(sentence | α) is intractable for auto-regressive large language models (LLMs); in particular, computing *Pr(next-token | α, prefix) is intractable.*

Solution: language models that allow efficient conditioning

Tractable Probabilistic Models

Step 1: distill a TPM that *approximates* the distribution of an LLM.



A simple *Hidden Markov Model* architecture will suffice!

- 50k emission tokens *x*
- 4096 hidden states *z*
- trained as a *Probabilistic Circuit* in Juice.jl from LLM samples
- scaled up using *Latent Variable Distillation* [ICLR 2023]

Anji Liu, Honghua Zhang, Guy Van den Broeck. Scaling Up Probabilistic Circuits by Latent Variable Distillation, ICLR, 2023

Step 2: compute Pr(next-token | prefix, α) via TPM

Dynamic programming algorithm in PyTorch that takes logical constraints α.

Can be quite complex: all inflections, all positions in seq, …

Step 3: control auto-regressive generation with the LLM.

Assume independence between fluency β and constraint α

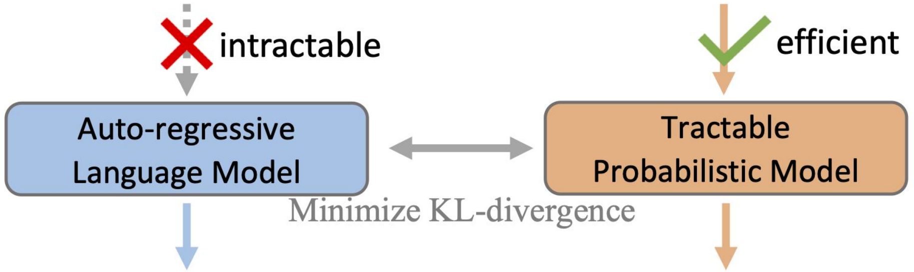
$$
\begin{aligned}
&\Pr_{\text{TPM}}(x_{t+1} \mid x_{1:t}, \alpha, \beta) \\
&\quad \propto \Pr_{\text{TPM}}(\alpha \mid x_{1:t+1}, \beta) \cdot \Pr_{\text{TPM}}(x_{t+1} \mid x_{1:t}, \beta) \\
&\quad \propto \Pr_{\text{TPM}}(\alpha \mid x_{1:t+1}) \cdot \Pr_{\text{LM}}(x_{t+1} \mid x_{1:t})
\end{aligned}
$$

↳ sleight of hand

Controlling Language Generation via TPM

Lexical Constraint α: the sentence contains keyword "**winter**"

Probabilistic Query: Pr(next-token | α, prefix = "**the weather is**")



next-token	Pr(next-token | prefix)
cold	0.05
warm	0.10

next-token	Pr(α | next-token, prefix)
cold	0.50
warm	0.01

next-token	p(next-token | α, prefix)
cold	0.025
warm	0.001

CommonGen: a challenging constrained generation task

Method	Generation Quality				Constraint Satisfaction			
	ROUGE-L		BLEU-4		Coverage		Success Rate	
Unsupervised	*dev*	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*
InsNet (Lu et al., 2022a)	-	-	18.7	-	**100.0**		**100.0**	-
NeuroLogic (Lu et al., 2021)	-	41.9	-	24.7	-	96.7	-	-
A*esque (Lu et al., 2022b)	-	**44.3**	-	28.6	-	97.1	-	-
NADO (Meng et al., 2022)	-	-	26.2	-	96.1	-	-	-
Ours	**44.6**	44.1	**29.9**	**29.4**	**100.0**	**100.0**	**100.0**	**100.0**
Supervised	*dev*	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*
NeuroLogic (Lu et al., 2021)	-	42.8	-	26.7	-	97.7	-	93.9†
A*esque (Lu et al., 2022b)	-	43.6	-	28.2	-	97.8	-	97.9†
NADO (Meng et al., 2022)	44.4†	-	30.8	-	97.1	-	88.8†	-
Ours	**46.0**	**45.6**	**34.1**	**32.9**	**100.0**	**100.0**	**100.0**	**100.0**

State-of-the-art performance on the CommonGen dataset, beating baselines from various families of constrained generation techniques with a large margin. All baselines use GPT2-large as the base model.

Outline

1. The paradox of learning to reason from data

~~*deep learning*~~

2. **Architectures for Learning and Reasoning**

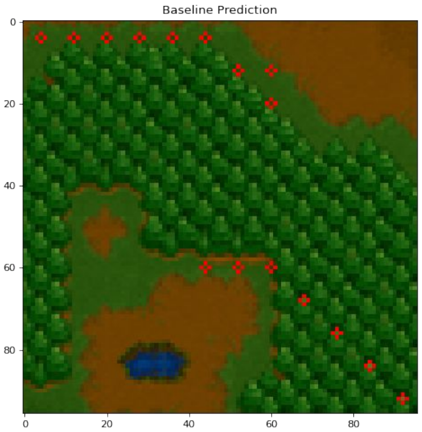
logical reasoning + *deep learning*

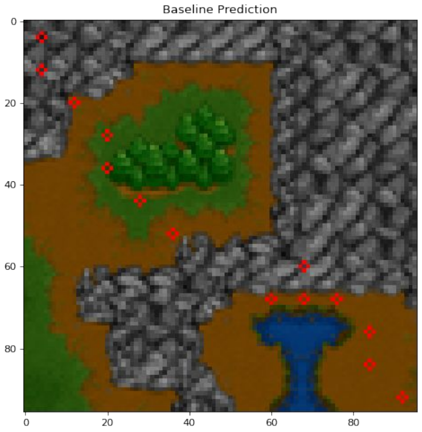
   a. Constrained language generation
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

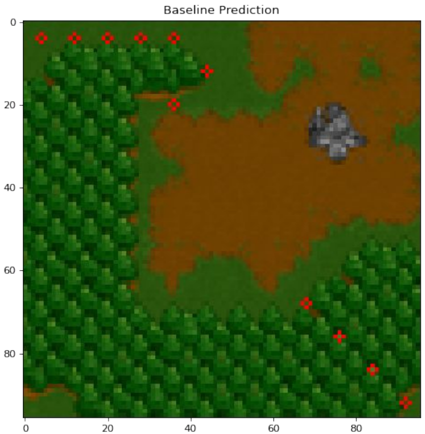
Warcraft Shortest Path

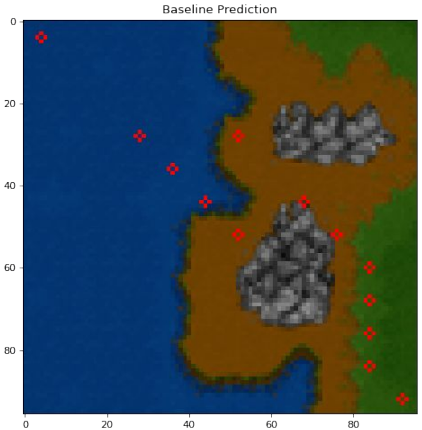
// for a 12×12 grid, 2^{144} states but only 10^{10} valid ones!

[Differentiation of Blackbox Combinatorial Solvers, Marin Vlastelica, Anselm Paulus, Vít Musil, Georg Martius, Michal Rolínek, 2019]


Baseline Prediction
0
20
40
60
80
0
20
40
60
80


Baseline Prediction
0
20
40
60
80
0
20
40
60
80


Baseline Prediction
0
20
40
60
80
0
20
40
60
80


Baseline Prediction
0
20
40
60
80
0
20
40
60
80

Architecture	Exact Match	Hamming Score	Consistency
ResNet-18+FIL	55.0	**97.7**	56.9

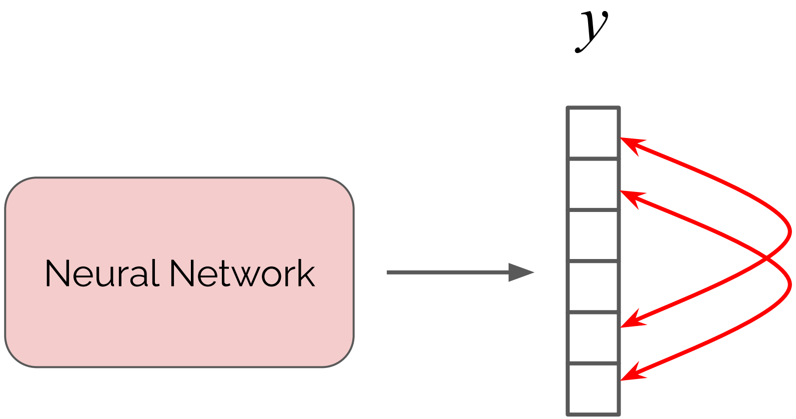
Is prediction the shortest path?
This is the real task!

Are individual edge predictions correct?

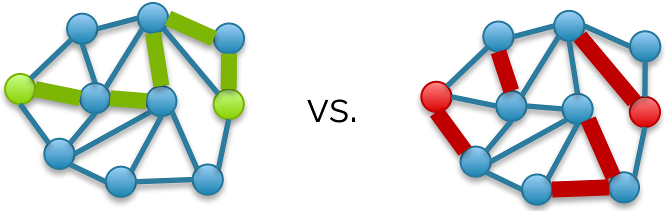
Is output a path?

Kareem Ahmed, Eric Wang, Kai-Wei Chang and Guy Van den Broeck. Neuro-Symbolic Entropy Regularization, 2021.

Declarative Knowledge of the Output



How is the output structured?
Are all possible outputs valid?



How are the outputs related to each other?

Learning this from data is inefficient
Much easier to express this declaratively

pylon

PyTorch Code

```
for i in range(train_iters):
    ...
    py = model(x)
    ...
    loss = CrossEntropy(py,...)
```

1 Specify knowledge as a predicate

```
def check(y):
    ...
    return isValid
```

pylon

PyTorch Code

```
for i in range(train_iters):
    ...
    py = model(x)
    ...
    loss = CrossEntropy(py,...)

    loss += constraint_loss(check)(py)
```

1 Specify knowledge as a predicate

```
def check(y):
    ...
    return isValid
```

2 Add as loss to training

```
loss += constraint_loss(check)
```

pylon

PyTorch Code

```
for i in range(train_iters):
    ...
    py = model(x)
    ...
    loss = CrossEntropy(py,...)

    loss += constraint_loss(check)(py)
```

1 Specify knowledge as a predicate

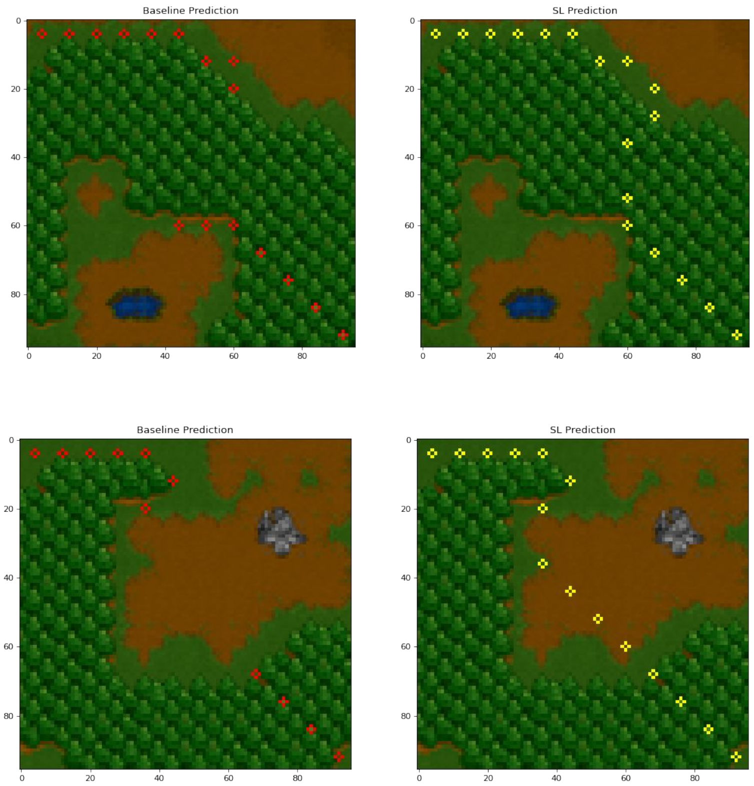
```
def check(y):
    ...
    return isValid
```

2 Add as loss to training

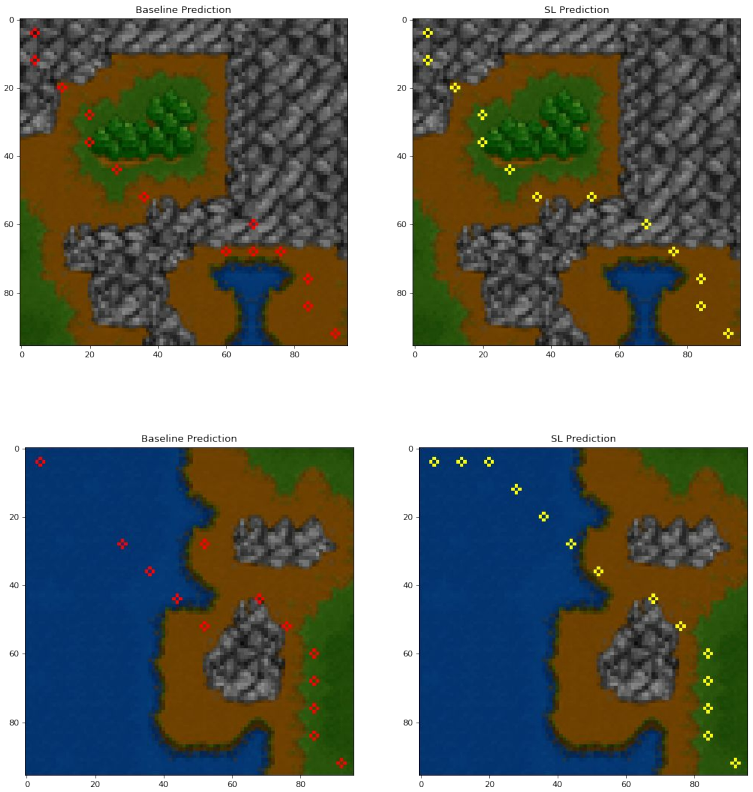
```
loss += constraint_loss(check)
```

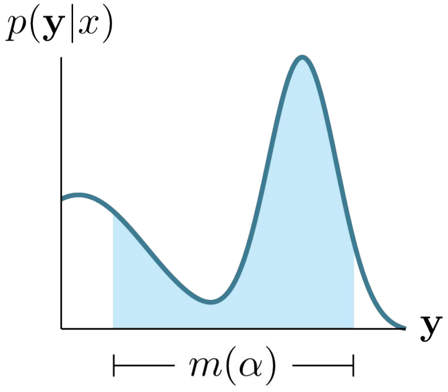
3 pylon derives the gradients
(solves a combinatorial problem)

without constraint *with constraint*

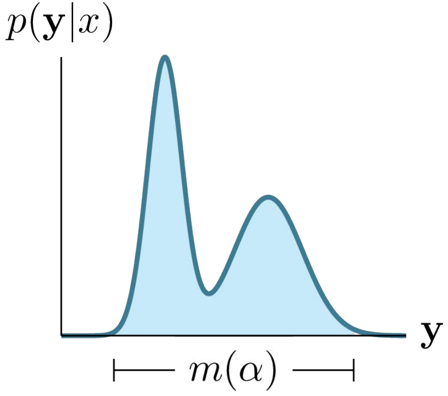


without constraint *with constraint*





a) A network uncertain over both valid & invalid predictions



c) A network allocating most of its mass to models of constraint

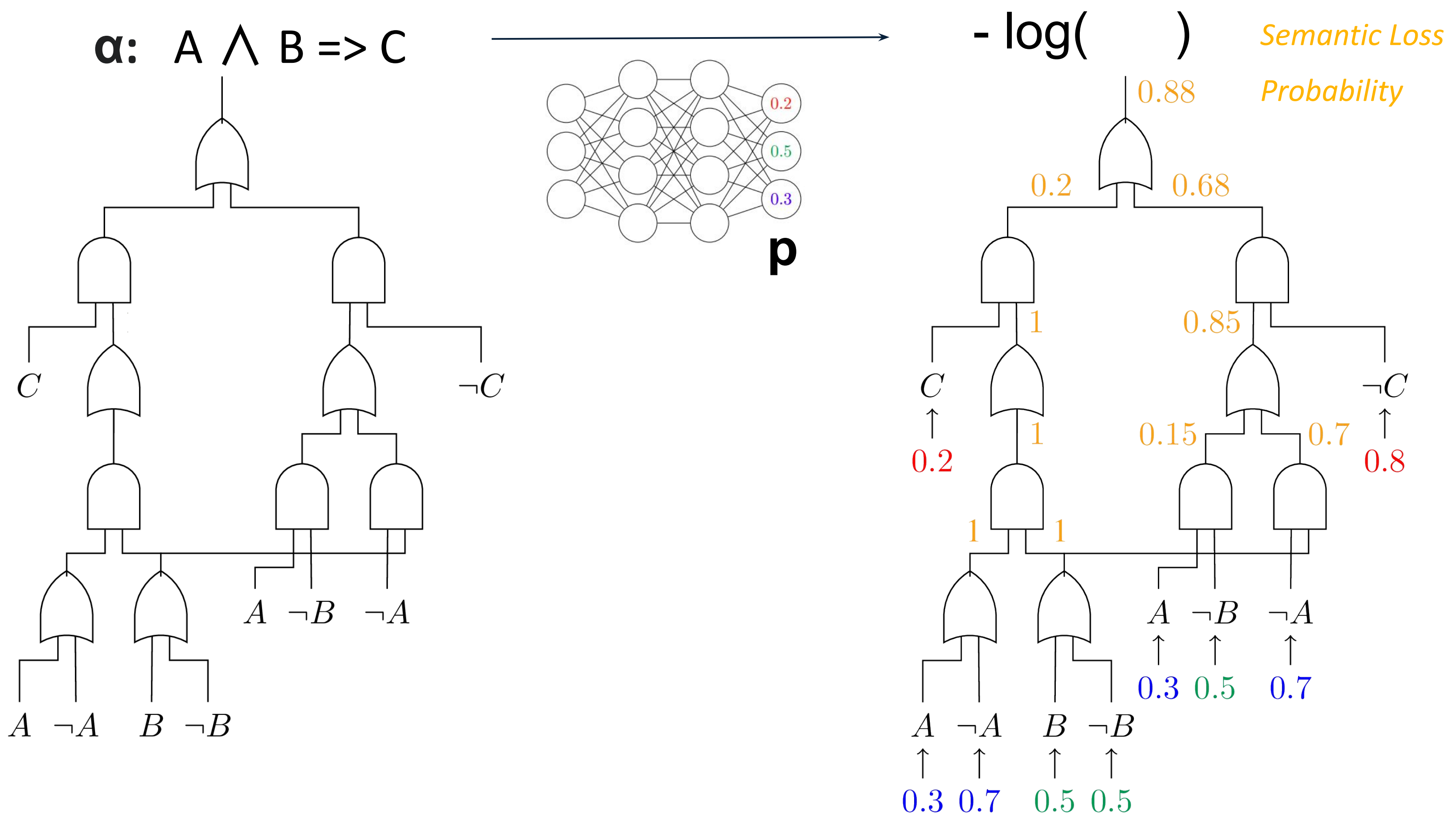
Semantic Loss

$$\mathrm{L^s}(\alpha, \mathsf{p}) \propto -\log \underbrace{\sum_{\mathbf{x} \models \alpha} \prod_{i:\mathbf{x} \models X_i} \mathsf{p}_i \prod_{i:\mathbf{x} \models \neg X_i} (1 - \mathsf{p}_i)}$$

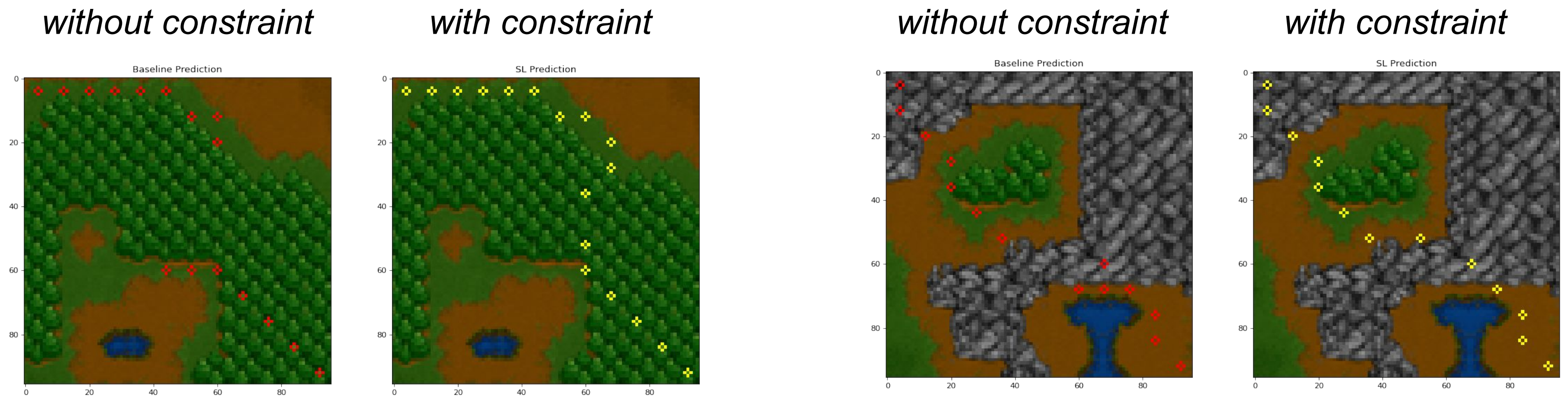
Probability of satisfying constraint α after sampling from neural net output layer **p**

In general: #P-hard ☹️

Do this probabilistic-logical reasoning during learning in a computation graph



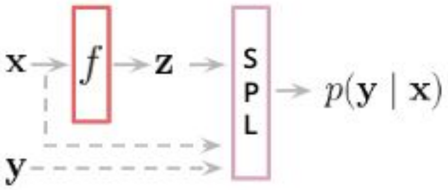
α: A ∧ B => C
C
¬C
A
¬B
¬A
A
¬A
B
¬B
0.2
0.5
0.3
p
- log( )
Semantic Loss
Probability
0.88
0.2
0.68
1
0.85
1
0.15
0.7
0.2
0.8
1
1
0.3
0.5
0.7
0.3
0.7
0.5
0.5

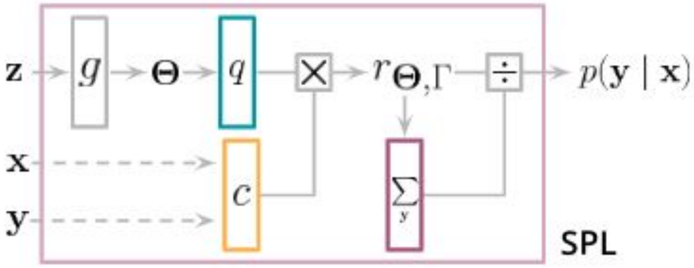


Architecture	Exact Match	Hamming Score	Consistency
ResNet-18+FIL	55.0	**97.7**	56.9
ResNet-18+$\mathcal{L}_{\text{SL}}$	59.4	**97.7**	61.2

Semantic Probabilistic Layers

- How to give a 100% guarantee that Boolean constraints will be satisfied?
- Bake the constraint into the neural network as a special layer





- Secret sauce is tractable circuits – computation graphs for reasoning

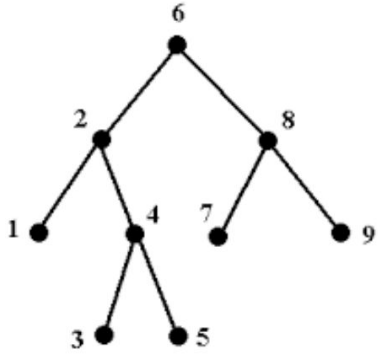
Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck and Antonio Vergari. Semantic Probabilistic Layers for Neuro-Symbolic Learning, 2022.

GROUND TRUTH | RESNET-18 | SEMANTIC LOSS | SPL (ours)

ARCHITECTURE	EXACT MATCH	HAMMING SCORE	CONSISTENCY
RESNET-18+FIL	55.0	**97.7**	56.9
RESNET-18+$\mathcal{L}_{\text{SL}}$	59.4	**97.7**	61.2
RESNET-18+SPL	75.1	97.6	**100.0**
OVERPARAM. SDD	**78.2**	96.3	**100.0**

Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck and Antonio Vergari. Semantic Probabilistic Layers for Neuro-Symbolic Learning, 2022.

Hierarchical Multi-Label Classification

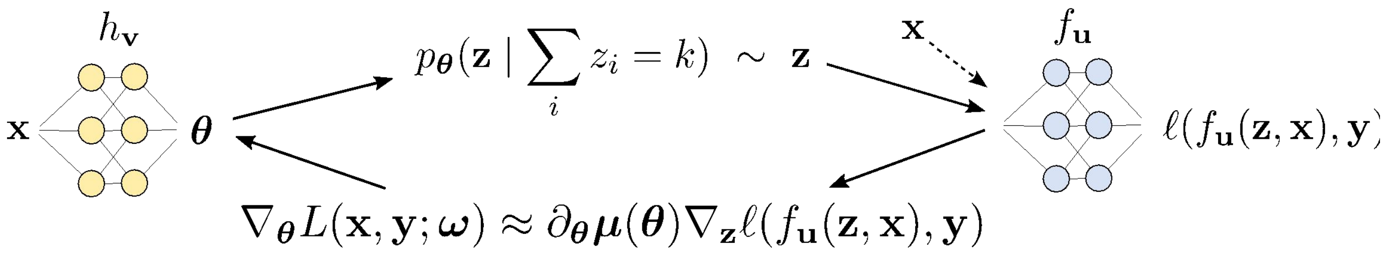


"if the image is classified as a dog, it must also be classified as an animal"

"if the image is classified as an animal, it must be classified as either cat or dog"

Dataset	Exact Match	
	HMCNN	MLP+SPL
CellCycle	3.05 ± 0.11	$\mathbf{3.79 \pm 0.18}$
Derisi	1.39 ± 0.47	$\mathbf{2.28 \pm 0.23}$
Eisen	5.40 ± 0.15	$\mathbf{6.18 \pm 0.33}$
Expr	4.20 ± 0.21	$\mathbf{5.54 \pm 0.36}$
Gasch1	3.48 ± 0.96	$\mathbf{4.65 \pm 0.30}$
Gasch2	3.11 ± 0.08	$\mathbf{3.95 \pm 0.28}$
Seq	5.24 ± 0.27	$\mathbf{7.98 \pm 0.28}$
Spo	$\mathbf{1.97 \pm 0.06}$	1.92 ± 0.11
Diatoms	48.21 ± 0.57	$\mathbf{58.71 \pm 0.68}$
Enron	5.97 ± 0.56	$\mathbf{8.18 \pm 0.68}$
Imclef07a	79.75 ± 0.38	$\mathbf{86.08 \pm 0.45}$
Imclef07d	76.47 ± 0.35	$\mathbf{81.06 \pm 0.68}$

SIMPLE: Gradient Estimator for *k*-Subset Sampling

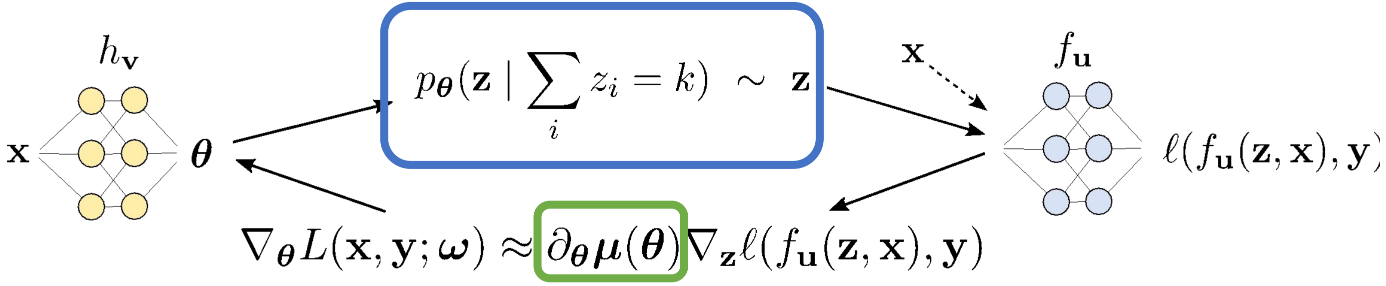


Example.
Learning to Explain (L2X)

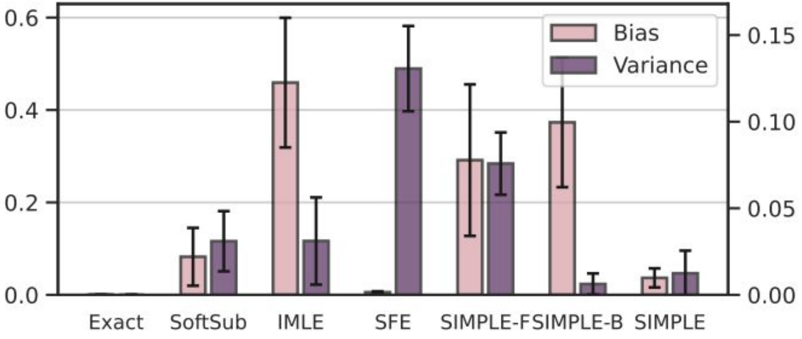
Taste Score	**Key Words (*k* = 10)**
0.7	a lite bodied beer with a pleasant taste. was like a reddish color. a little like wood and caramel with a hop finish. has a sort of fruity flavor like grapes or cherry that is sort of buried in there. mouth feel was lite, sort of bubbly. not hard to down, though a bit harder then one would expect given the taste.

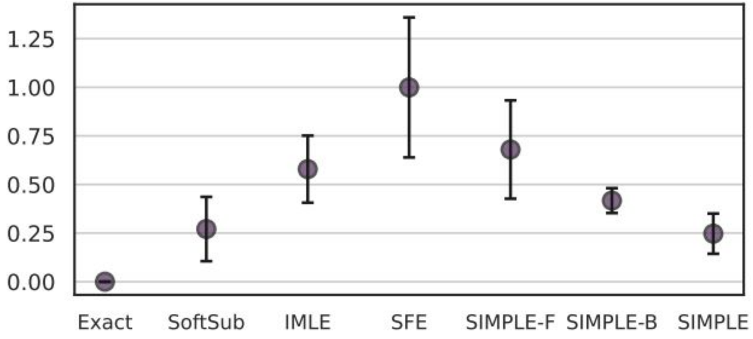
Kareem Ahmed, Zhe Zeng, Mathias Niepert, Guy Van den Broeck. [SIMPLE: A Gradient Estimator for k-Subset Sampling](), ICLR 2023

SIMPLE: Gradient Estimator for *k*-Subset Sampling



We achieve ***lower bias and variance*** by exact, discrete samples and exact derivative of conditional marginals.





Kareem Ahmed, Zhe Zeng, Mathias Niepert, Guy Van den Broeck. [SIMPLE: A Gradient Estimator for k-Subset Sampling](), ICLR 2023

Experiment: Learn to Explain (L2X)

Taste Score	**Key Words (*k* = 10)**
0.7	a lite bodied beer with a pleasant taste. was like a reddish color. a little like wood and caramel with a hop finish. has a sort of fruity flavor like grapes or cherry that is sort of buried in there. mouth feel was lite, sort of bubbly. not hard to down, though a bit harder then one would expect given the taste.

Results for three aspects with *k* = 10

Method	Appearance		Palate		Taste	
	Test MSE	Precision	Test MSE	Precision	Test MSE	Precision
SIMPLE (Ours)	**2.35 ± 0.28**	**66.81 ± 7.56**	**2.68 ± 0.06**	**44.78 ± 2.75**	**2.11 ± 0.02**	**42.31 ± 0.61**
L2X (t = 0.1)	10.70 ± 4.82	30.02 ± 15.82	6.70 ± 0.63	**50.39 ± 13.58**	6.92 ± 1.61	32.23 ± 4.92
SoftSub (t = 0.5)	**2.48 ± 0.10**	52.86 ± 7.08	2.94 ± 0.08	39.17 ± 3.17	2.18 ± 0.10	**41.98 ± 1.42**
I-MLE (τ = 30)	**2.51 ± 0.05**	**65.47 ± 4.95**	2.96 ± 0.04	40.73 ± 3.15	2.38 ± 0.04	**41.38 ± 1.55**

Results for aspect Aroma, for *k* in {5, 10, 15}

Method	$k = 5$		$k = 10$		$k = 15$	
	Test MSE	Precision	Test MSE	Precision	Test MSE	Precision
SIMPLE (Ours)	**2.27 ± 0.05**	**57.30 ± 3.04**	**2.23 ± 0.03**	**47.17 ± 2.11**	3.20 ± 0.04	**53.18 ± 1.09**
L2X (t = 0.1)	5.75 ± 0.30	33.63 ± 6.91	6.68 ± 1.08	26.65 ± 9.39	7.71 ± 0.64	23.49 ± 10.93
SoftSub (t = 0.5)	2.57 ± 0.12	**54.06 ± 6.29**	2.67 ± 0.14	44.44 ± 2.27	**2.52 ± 0.07**	37.78 ± 1.71
I-MLE (τ = 30)	2.62 ± 0.05	**54.76 ± 2.50**	2.71 ± 0.10	**47.98 ± 2.26**	2.91 ± 0.18	39.56 ± 2.07

Kareem Ahmed, Zhe Zeng, Mathias Niepert, Guy Van den Broeck. SIMPLE: A Gradient Estimator for k-Subset Sampling, ICLR 2023

Outline

1. The paradox of learning to reason from data

   ~~deep learning~~

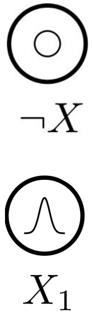
2. **Architectures for Learning and Reasoning**

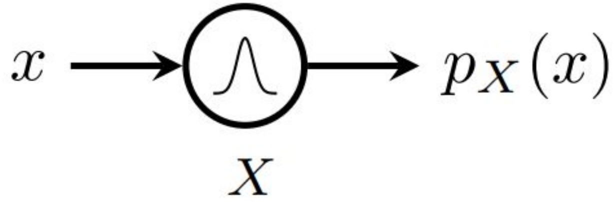
   logical reasoning + *deep learning*

   a. Constrained language generation
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

Probabilistic circuits

computational graphs that recursively define distributions



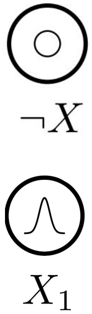


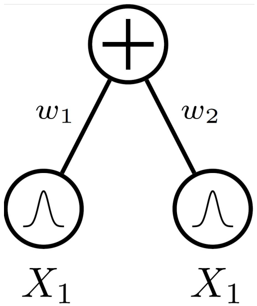
Simple distributions are tractable "black boxes" for:

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

Probabilistic circuits

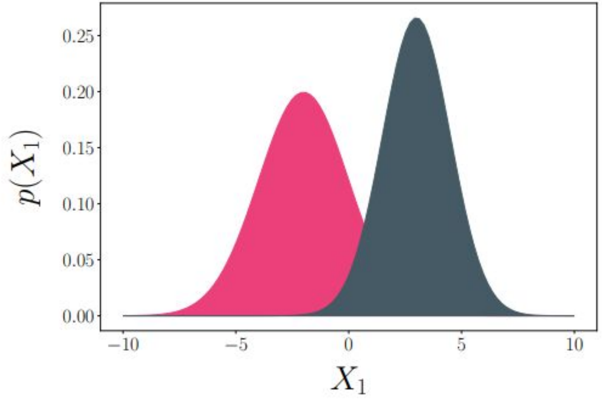
computational graphs that recursively define distributions





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

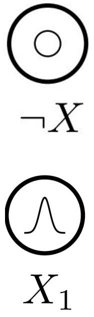
$\Rightarrow$ *mixtures*

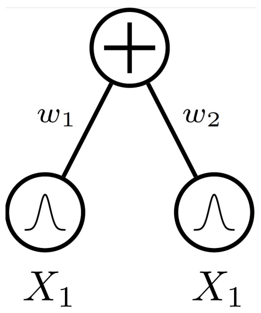


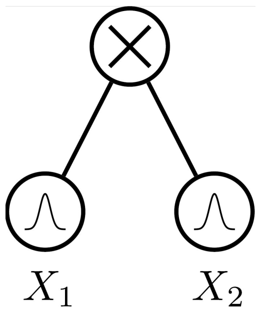
$$p(X) = p(Z = \boxed{1}) \cdot p_1(X|Z = \boxed{1}) + p(Z = \boxed{2}) \cdot p_2(X|Z = \boxed{2})$$

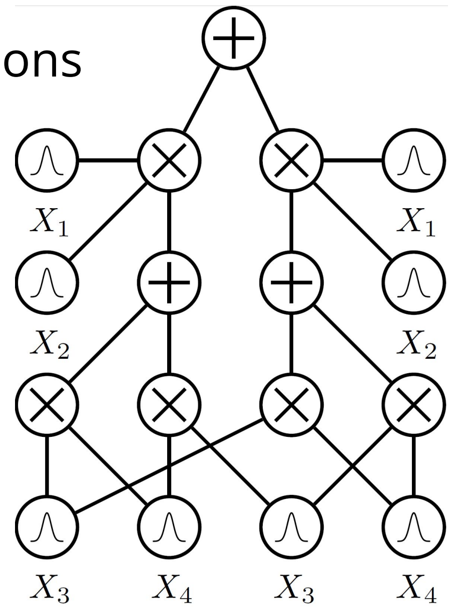
Probabilistic circuits

computational graphs that recursively define distributions









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

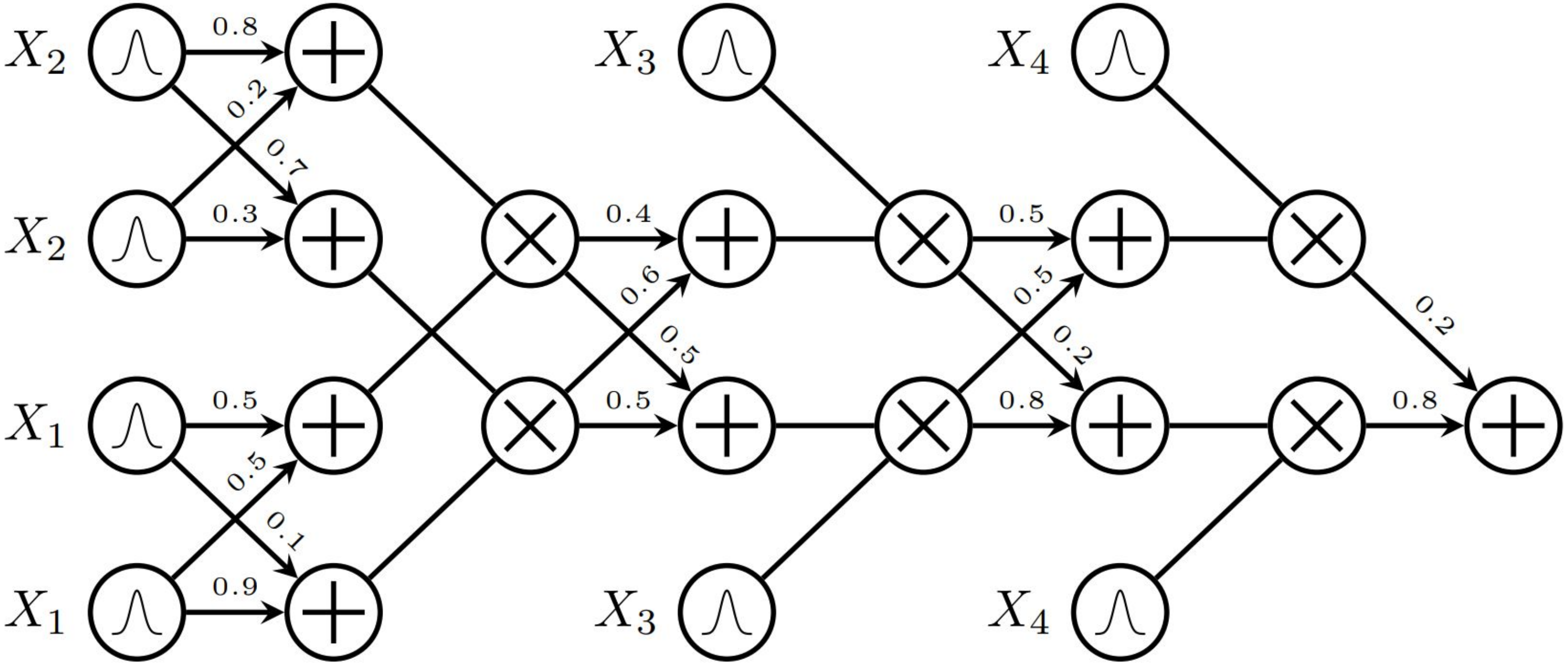
$\Rightarrow$ *mixtures*

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

$\Rightarrow$ *factorizations*

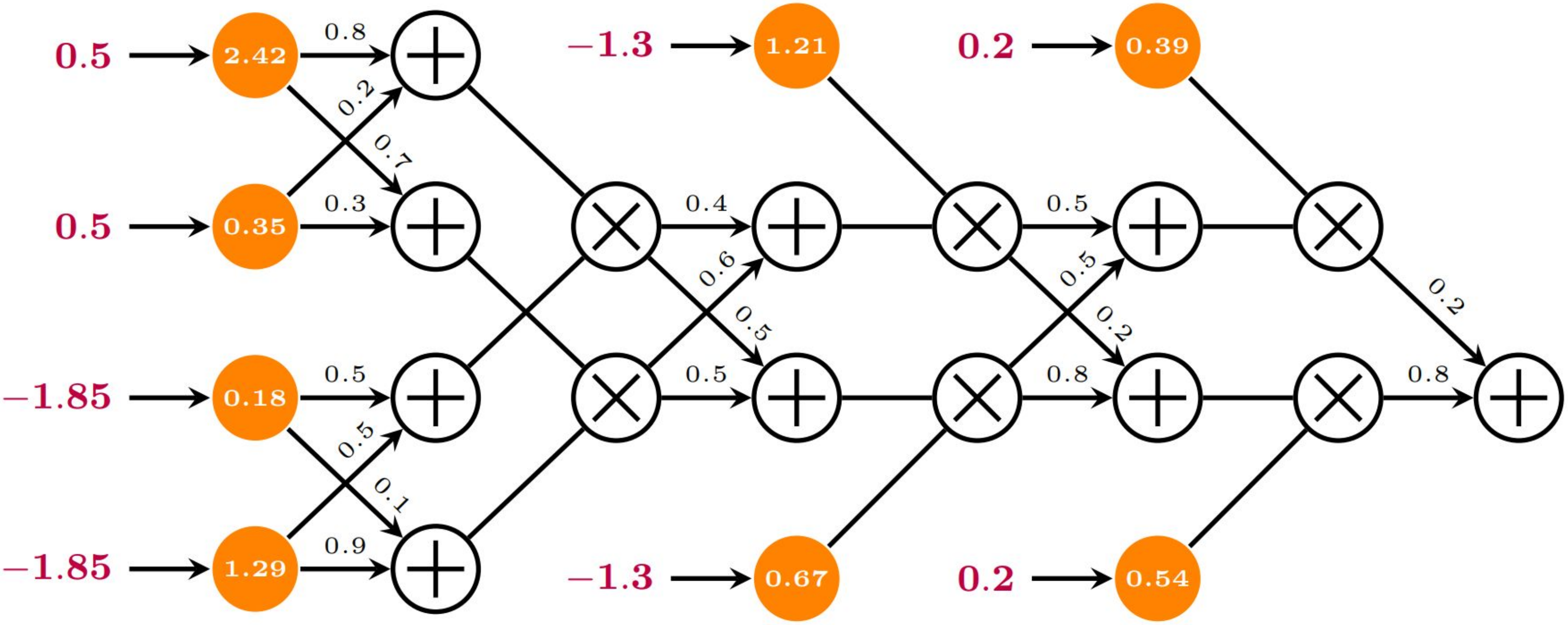
Likelihood

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



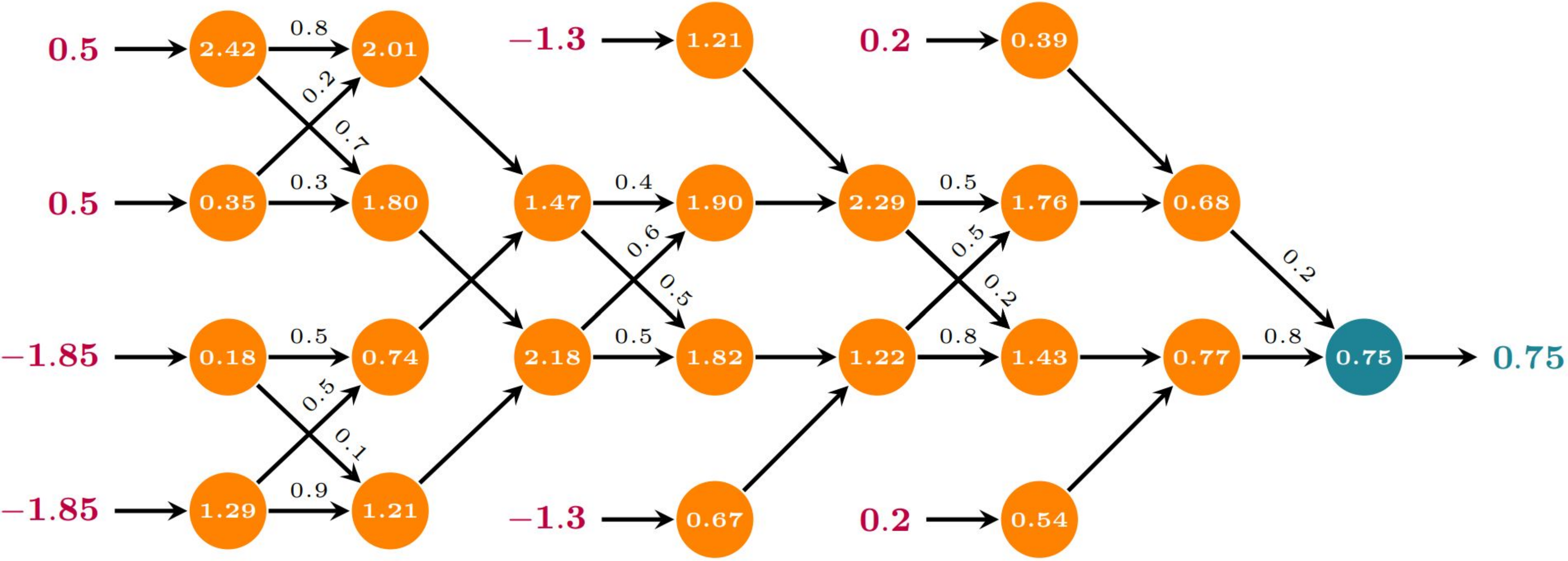
Likelihood

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



Likelihood

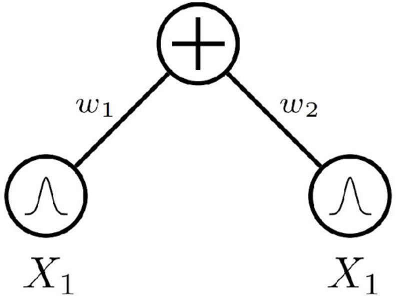
$$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$$



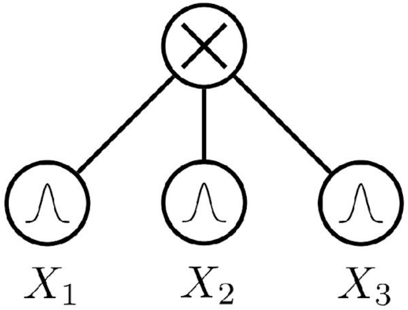
Tractable marginals

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

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



smooth circuit



decomposable circuit

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

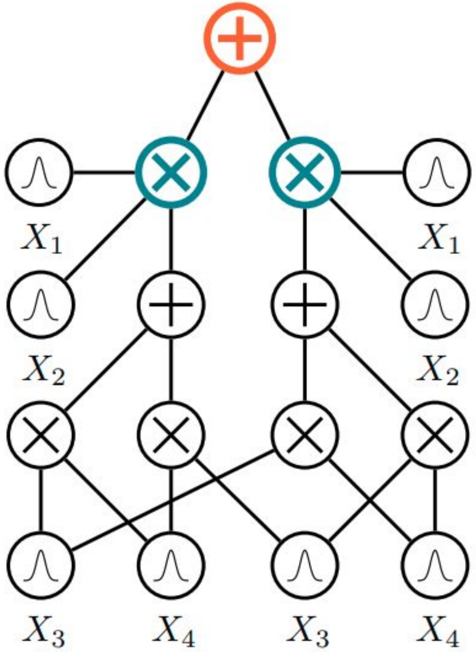
Smoothness + decomposability = tractable MAR

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

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

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

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



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

Smoothness + decomposability = tractable MAR

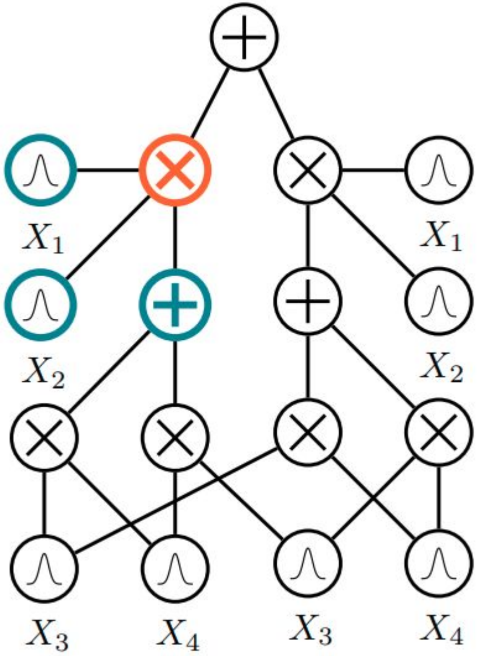
If $p(\mathbf{x},\mathbf{y},\mathbf{z}) = p(\mathbf{x})p(\mathbf{y})p(\mathbf{z})$, (***decomposability***):

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

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

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

$\Rightarrow$ *integrals decompose into easier ones*



Smoothness + decomposability = tractable MAR

Forward pass evaluation for MAR

$\Rightarrow$ *linear in circuit size!*

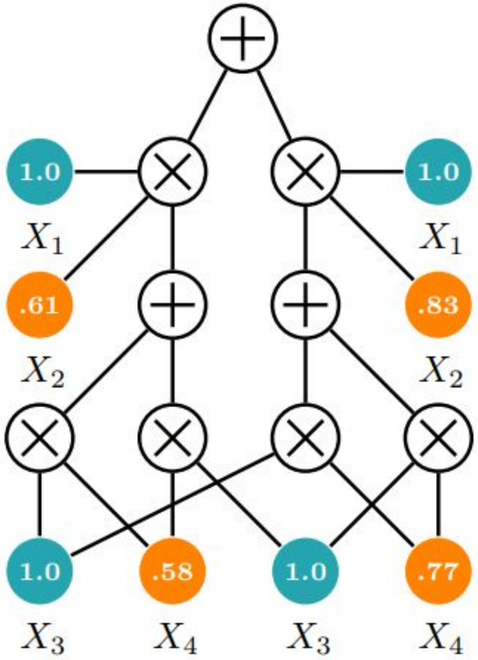
E.g. to compute $p(x_2, x_4)$:

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

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

- leafs over X_2 and X_4 output ***EVI***

- feedforward evaluation (bottom-up)



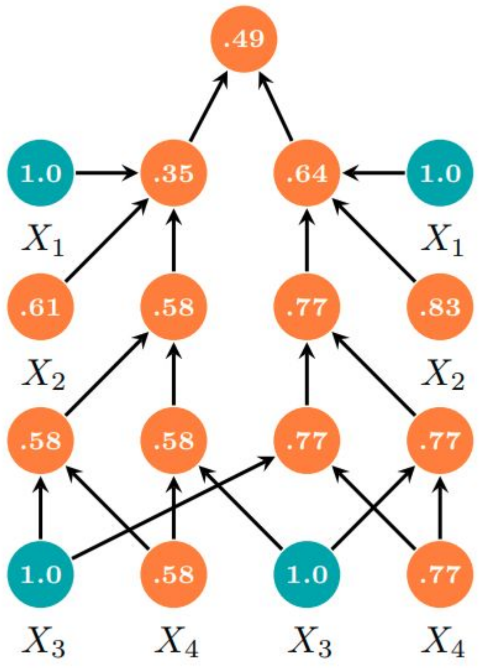
Smoothness + decomposability = tractable MAR

Forward pass evaluation for MAR

$\Rightarrow$ *linear in circuit size!*

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

- leafs over X_1 and X_3 output $\boldsymbol{Z}_i = \int p(x_i)dx_i$
  $\Rightarrow$ *for normalized leaf distributions:* 1.0
- leafs over X_2 and X_4 output ***EVI***
- feedforward evaluation (bottom-up)

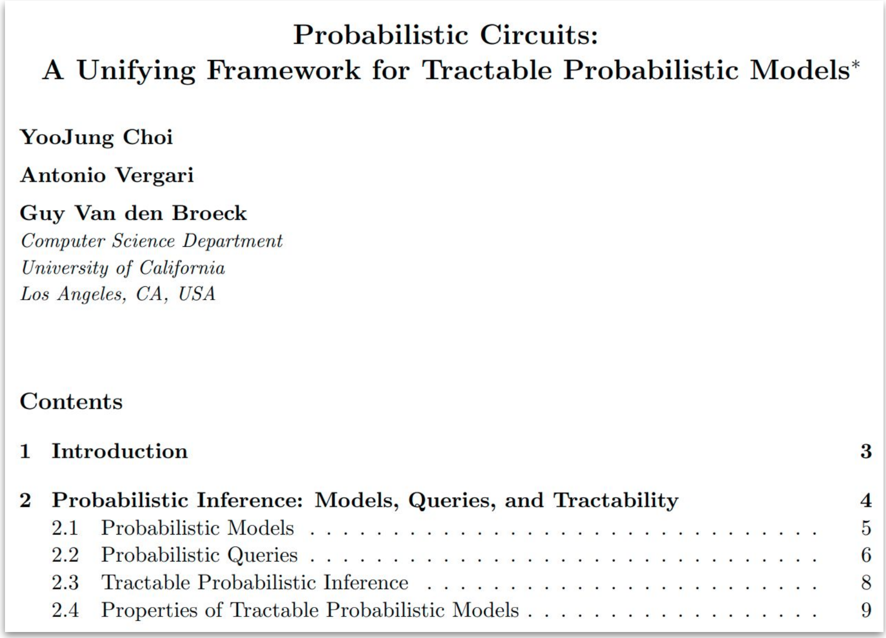


Learn more about probabilistic circuits?

Tutorial (3h)

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

Overview Paper (80p)



Probabilistic Circuits:
A Unifying Framework for Tractable Probabilistic Models*

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

Contents

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

Outline

1. The paradox of learning to reason from data

   ~~deep learning~~

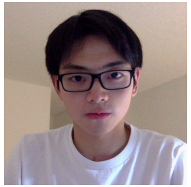
2. Architectures for Learning and Reasoning

   logical (and probabilistic) reasoning + *deep learning*

   a. Constrained language generation
   b. Constrained structured prediction
   c. Secret sauce: tractable circuits

Thanks

This was the work of many wonderful students/postdocs/collaborators!



Honghua

Kareem

Zhe

Meihua

Anji

References: http://starai.cs.ucla.edu/publications/