At the Confluence of Logic and Learning

Guy Van den Broeck

Dagstuhl

September 3, 2019

Outline

1. The AI dilemma: logic vs. learning
2. Deep learning with symbolic knowledge
3. Efficient reasoning during learning
4. New machine learning formalisms
5. Statistical relational learning (tutorial)

Outline

1. **The AI dilemma: logic vs. learning**
2. Deep learning with symbolic knowledge
3. Efficient reasoning during learning
4. New machine learning formalisms
5. Statistical relational learning (tutorial)

The AI Dilemma

Pure Logic **Pure Learning**

The AI Dilemma

Pure Logic **Pure Learning**

- Slow thinking: deliberative, cognitive, model-based, extrapolation
- Amazing achievements until this day
- "*Pure logic is brittle*"
  noise, uncertainty, incomplete knowledge, …

The AI Dilemma

Pure Logic | **Pure Learning**

- Fast thinking: instinctive, perceptive, model-free, interpolation
- Amazing achievements recently
- “*Pure learning is brittle*”

  bias, algorithmic fairness, interpretability, explainability, adversarial attacks, unknown unknowns, calibration, verification, missing features, missing labels, data efficiency, shift in distribution, general robustness and safety

  fails to incorporate a sensible model of the world

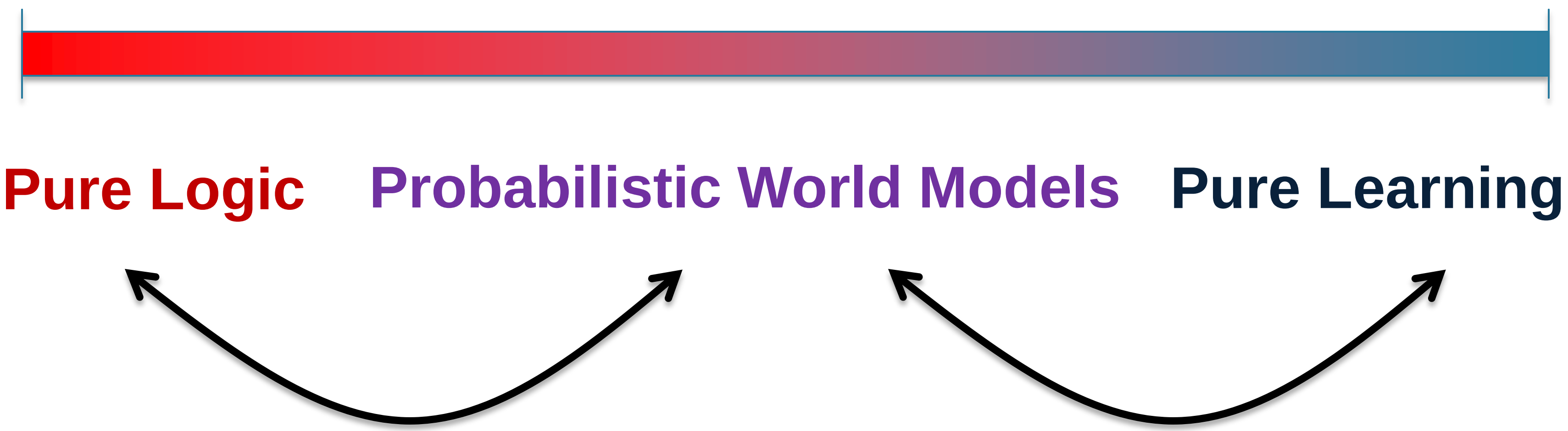
The **FALSE** AI Dilemma

So all hope is lost?

Probabilistic World Models

- Joint distribution P(X)
- Wealth of representations:
  can be causal, relational, etc.
- Knowledge + data
- Reasoning + learning

Then why isn't everything solved?



What did we gain?

What did we lose along the way?

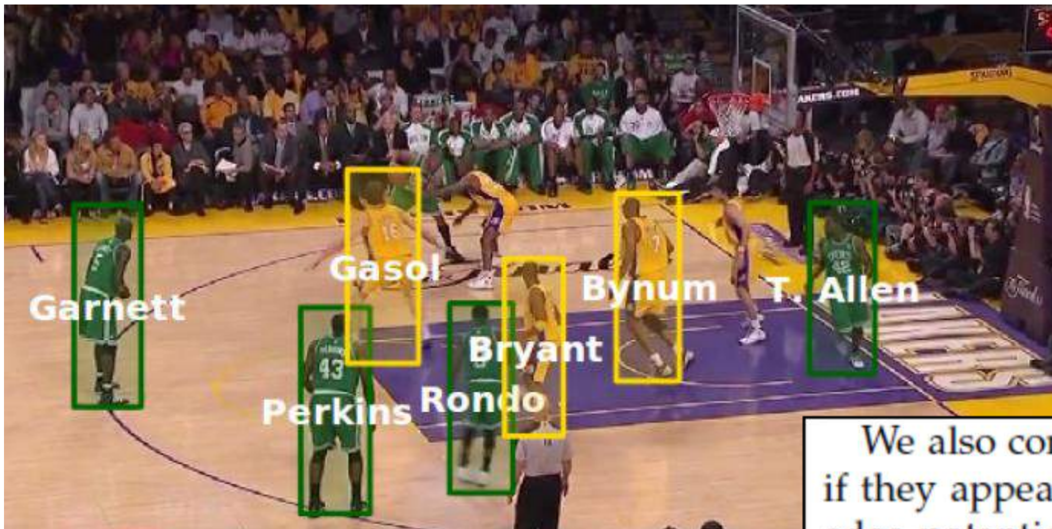
Pure Logic **Probabilistic World Models** **Pure Learning**

A New Synthesis of Learning and Reasoning

Outline

1. The AI dilemma: logic vs. learning
2. **Deep learning with symbolic knowledge**
3. Efficient reasoning during learning
4. New machine learning formalisms
5. Statistical relational learning (tutorial)
6. Lifted probabilistic inference

Motivation: Vision



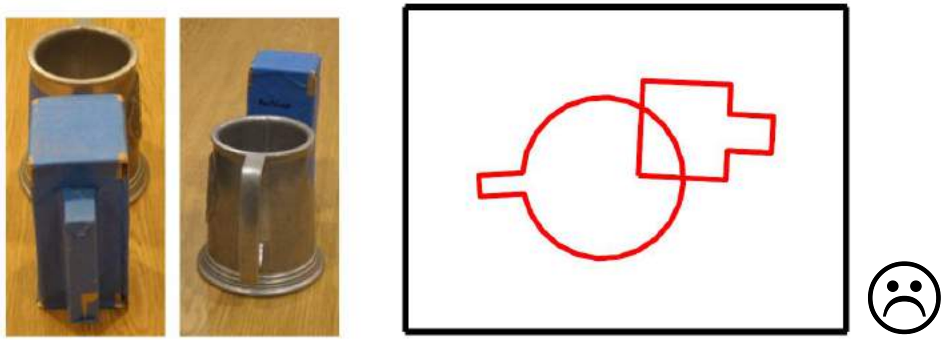
We also connect all pairs of identity nodes $y_{t,i}$ and $y_{t,j}$ if they appear in the same time t. We then introduce an edge potential that enforces mutual exclusion:

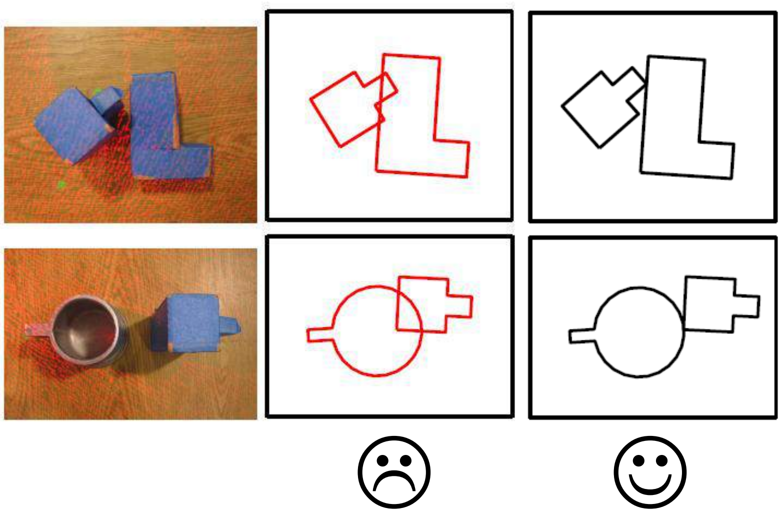
$$\psi_{\text{mutex}}(y_{t,i}, y_{t,j}) = \begin{cases} 1 & \text{if } y_{t,i} \neq y_{t,j} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

This potential specifies the constraint that a player can be appear only *once* in a frame. For example, if the i-th detection $y_{t,i}$ has been assign to Bryant, $y_{t,j}$ cannot have the same identity because Bryant is impossible to appear twice in a frame.

[Lu, W. L., Ting, J. A., Little, J. J., & Murphy, K. P. (2013). Learning to track and identify players from broadcast sports videos.]

Motivation: Robotics





The method developed in this paper can be used in a broad variety of semantic mapping and object manipulation tasks, providing an efficient and effective way to incorporate collision constraints into a recursive state estimator, obtaining optimal or near-optimal solutions.

[Wong, L. L., Kaelbling, L. P., & Lozano-Perez, T., Collision-free state estimation. ICRA 2012]

Motivation: Language

- Non-local dependencies:

  “At least one verb in each sentence”

- Sentence compression

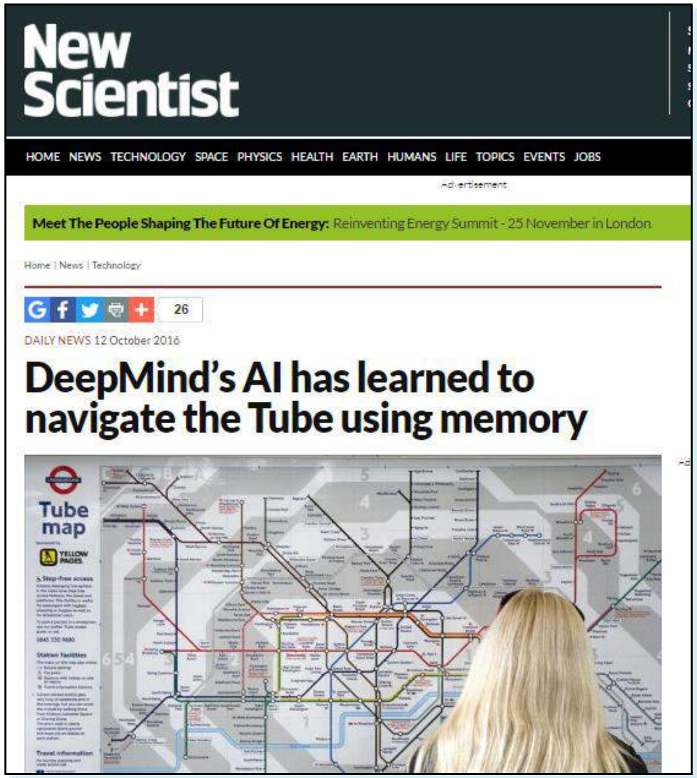
  “If a modifier is kept, its subject is also kept”

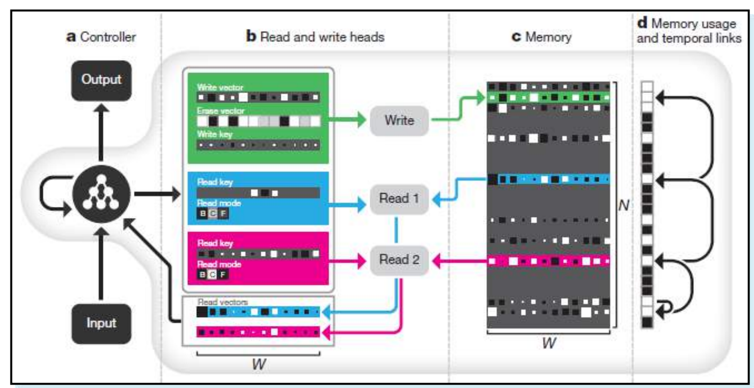
- NELL ontology and rules

… and much more!

[Chang, M., Ratinov, L., & Roth, D. (2008). Constraints as prior knowledge],
[Ganchev, K., Gillenwater, J., & Taskar, B. (2010). Posterior regularization for structured latent variable models]
… and many many more!

Motivation: Deep Learning

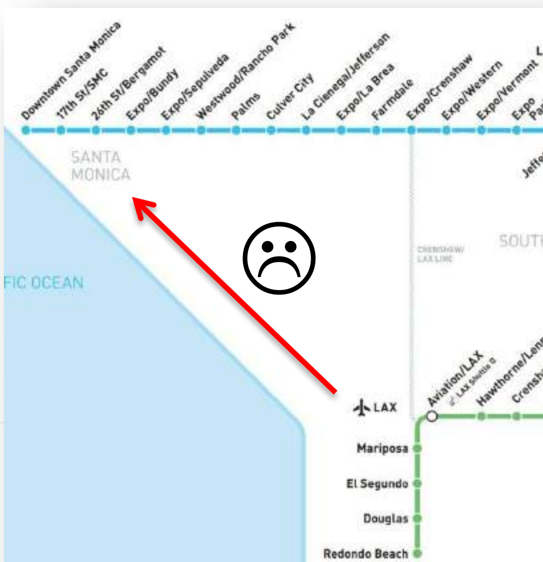




[Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., et al.. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, *538*(7626), 471-476.]

Motivation: Deep Learning

DeepMind's latest technique uses external memory to **solve tasks that require logic and reasoning — a step toward more human-like AI.**



… but …

optimal planner recalculating a shortest path to the end node. To ensure that the network always moved to a valid node, the output distribution was renormalized over the set of possible triples outgoing from the current node. The performance

it also received input triples during the answer phase, indicating the actions chosen on the previous time-step. This makes the problem a 'structured prediction'

[Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., et al.. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, *538*(7626), 471-476.]

Knowledge vs. Data

- Where did the world knowledge go?
  - Python scripts
    - Decode/encode cleverly
    - Fix inconsistent beliefs
  - Rule-based decision systems
  - Dataset design
  - "a big hack" (with author's permission)
- In some sense we went backwards

  Less principled, scientific, and intellectually satisfying ways of incorporating knowledge

Learning with Symbolic Knowledge

L	K	P	A	Students
0	0	1	0	6
0	0	1	1	54
0	1	1	1	10
1	0	0	0	5
1	0	1	0	1
1	0	1	1	0
1	1	0	0	17
1	1	1	0	4
1	1	1	1	3

Data + **Constraints**

(Background Knowledge)
(Physics)

$$P \vee L$$
$$A \Rightarrow P$$
$$K \Rightarrow (P \vee L)$$

1. Must take at least one of Probability (**P**) or Logic (**L**).
2. Probability (**P**) is a prerequisite for AI (**A**).
3. The prerequisites for KR (**K**) is either AI (**A**) or Logic (**L**).

Learning with Symbolic Knowledge

L	K	P	A	Students
0	0	1	0	6
0	0	1	1	54
0	1	1	1	10
1	0	0	0	5
1	0	1	0	1
1	0	1	1	0
1	1	0	0	17
1	1	1	0	4
1	1	1	1	3

Data + **Constraints**
(Background Knowledge)
(Physics)

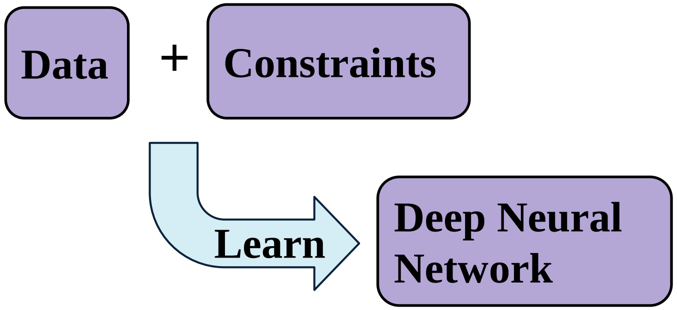
$$P \vee L$$
$$A \Rightarrow P$$
$$K \Rightarrow (P \vee L)$$

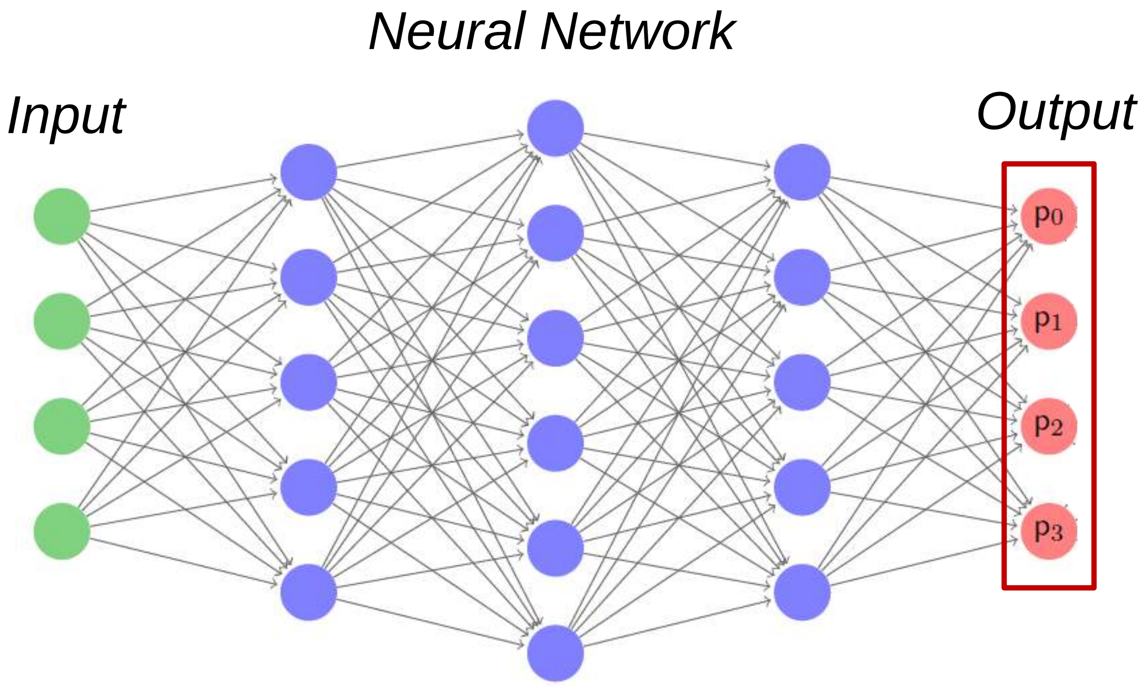
Learn

ML Model

Today's machine learning tools don't take knowledge as input! ☹

Deep Learning with Symbolic Knowledge





Output is probability vector **p**, not Boolean logic!

Semantic Loss

*Q: How close is output **p** to satisfying constraint α?*

*Answer: Semantic loss function L(α,**p**)*

- Axioms, for example:
  - If α constrains to one label, L(α,**p**) is cross-entropy
  - If α implies β then L(α,**p**) ≥ L(β,**p**) (*α more strict*)

- Implied Properties:
  - If α is equivalent to β then L(α,**p**) = L(β,**p**)
  - If **p** is Boolean and satisfies α then L(α,**p**) = 0

SEMANTIC Loss!

Semantic Loss: Definition

<u>Theorem</u>: Axioms imply unique semantic loss:

$$\mathrm{L}^{\mathrm{s}}(\alpha, \mathsf{p}) \propto -\log \underbrace{\sum_{\mathbf{x}\models\alpha} \underbrace{\prod_{i:\mathbf{x}\models X_i} \mathsf{p}_i \prod_{i:\mathbf{x}\models\neg X_i}(1-\mathsf{p}_i)}_{\text{Probability of getting state } \mathbf{x} \text{ after flipping coins with probabilities } \mathbf{p}}}_{\text{Probability of satisfying } \alpha \text{ after flipping coins with probabilities } \mathbf{p}}$$

Simple Example: Exactly-One

- Data must have some label

  We agree this must be one of the 10 digits:

- Exactly-one constraint

  → For 3 classes: $\begin{cases} x_1 \vee x_2 \vee x_3 \\ \neg x_1 \vee \neg x_2 \\ \neg x_2 \vee \neg x_3 \\ \neg x_1 \vee \neg x_3 \end{cases}$

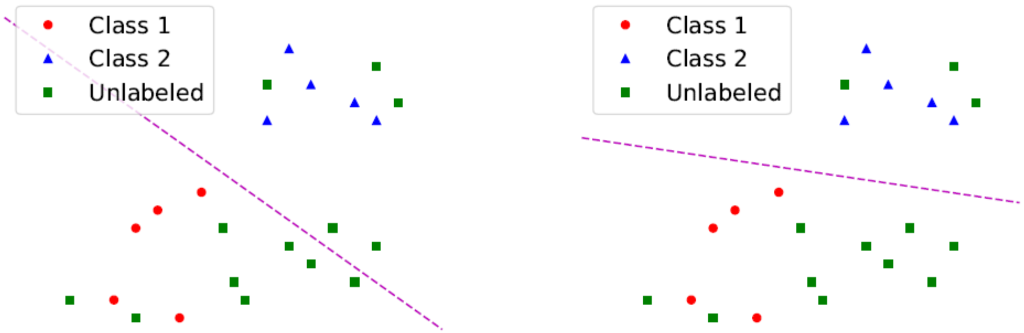
- Semantic loss:

$$\mathrm{L^s}(\text{exactly-one}, \mathsf{p}) \propto -\log \underbrace{\sum_{i=1}^{n} \underbrace{\mathsf{p}_i \prod_{j=1, j\neq i}^{n} (1 - \mathsf{p}_j)}_{\text{Only } x_i = 1 \text{ after flipping coins}}}_{\text{Exactly one true } x \text{ after flipping coins}}$$

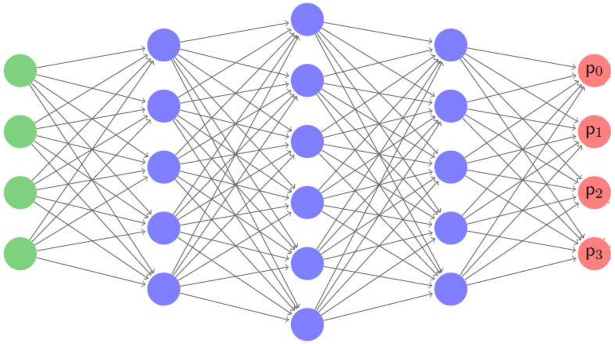
Semi-Supervised Learning

- Intuition: Unlabeled data must have some label

  Cf. entropy minimization, manifold learning



- Minimize exactly-one semantic loss on unlabeled data



Train with

$$existing\ loss + w \cdot semantic\ loss$$

Experimental Evaluation

Accuracy % with # of used labels	100	1000	ALL
AtlasRBF (Pitelis et al., 2014)	91.9 (±0.95)	96.32 (±0.12)	98.69
Deep Generative (Kingma et al., 2014)	96.67(±0.14)	97.60 (±0.02)	99.04
Virtual Adversarial (Miyato et al., 2016)	97.67	98.64	99.36
Ladder Net (Rasmus et al., 2015)	**98.94** (±0.37)	**99.16** (±0.08)	99.43 (±0.02)
Baseline: MLP, Gaussian Noise	78.46 (±1.94)	94.26 (±0.31)	99.34 (±0.08)
Baseline: Self-Training	72.55 (±4.21)	87.43 (±3.07)	
Baseline: MLP with Entropy Regularizer	96.27 (±0.64)	98.32 (±0.34)	99.37 (±0.12)
MLP with Semantic Loss	98.38 (±0.51)	98.78 (±0.17)	99.36 (±0.02)

Competitive with state of the art in semi-supervised deep learning

Accuracy % with # of used labels	100	500	1000	ALL
Ladder Net (Rasmus et al., 2015)	81.46 (±0.64)	85.18 (±0.27)	86.48 (±0.15)	90.46
Baseline: MLP, Gaussian Noise	69.45 (±2.03)	78.12 (±1.41)	80.94 (±0.84)	89.87
MLP with Semantic Loss	**86.74** (±0.71)	**89.49** (±0.24)	**89.67** (±0.09)	89.81

Outperforms SoA!

Same conclusion on CIFAR10

Accuracy % with # of used labels	4000	ALL
CNN Baseline in Ladder Net	76.67 (± 0.61)	90.73
Ladder Net (Rasmus et al., 2015)	79.60 (±0.47)	
Baseline: CNN, Whitening, Cropping	77.13	90.96
CNN with Semantic Loss	**81.79**	90.92

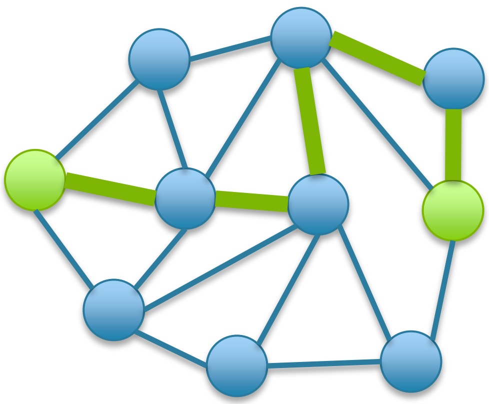
Outline

1. The AI dilemma: logic vs. learning
2. Deep learning with symbolic knowledge
3. **Efficient reasoning during learning**
4. New machine learning formalisms
5. Statistical relational learning (tutorial)

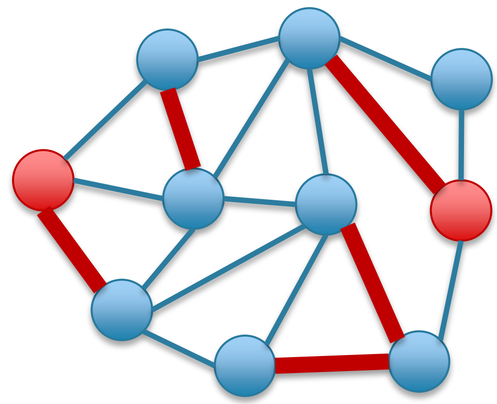
But what about *real* constraints?

- Path constraint

cf. Nature paper



vs.



- Example: 4x4 grids

  2^{24} = 184 paths + 16,777,032 non-paths

- Easily encoded as logical constraints ☺

[Nishino et al., Choi et al.]

How to Compute Semantic Loss?

- In general: #P-hard ☹

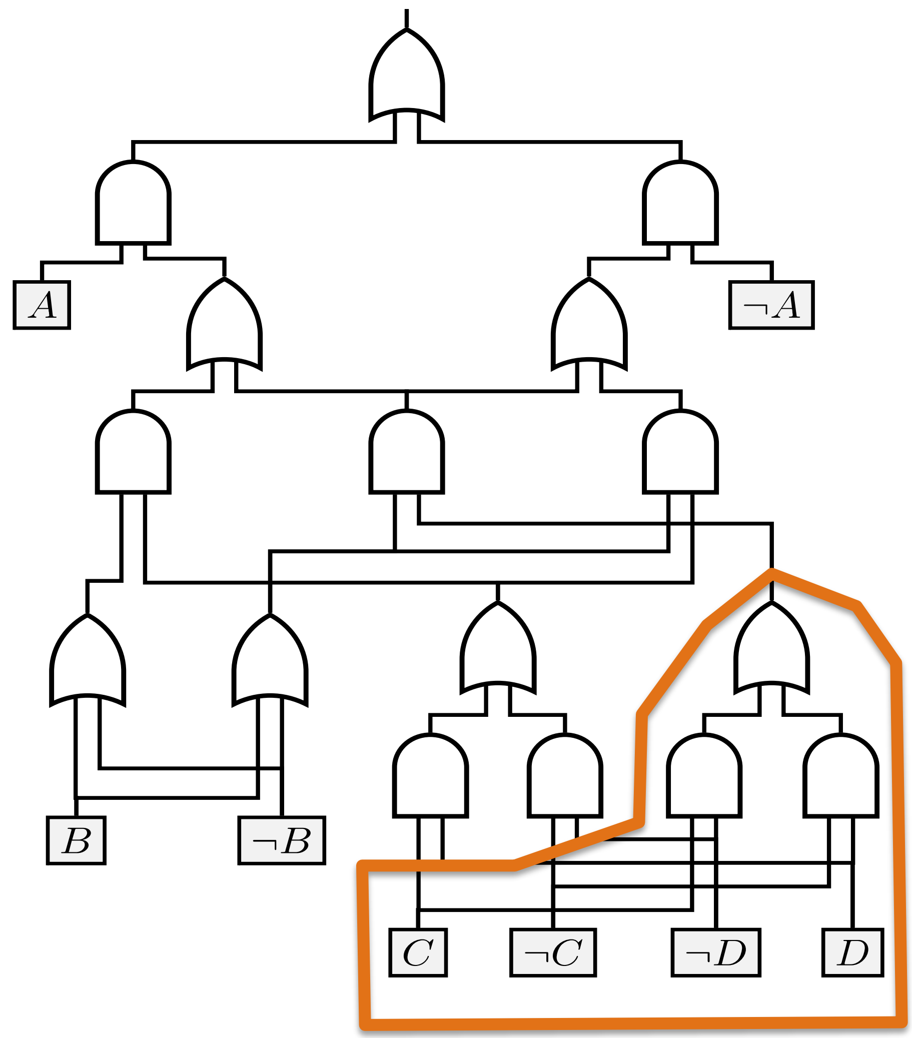
$$\mathrm{L^s}(\alpha, \mathsf{p}) \propto -\log \sum_{\mathbf{x} \models \alpha} \prod_{i:\mathbf{x} \models X_i} \mathsf{p}_i \prod_{i:\mathbf{x} \models \neg X_i} (1 - \mathsf{p}_i)$$

Reasoning Tool: Logical Circuits

Representation of logical sentences:

$(C \wedge \neg D) \vee (\neg C \wedge D)$

C XOR D

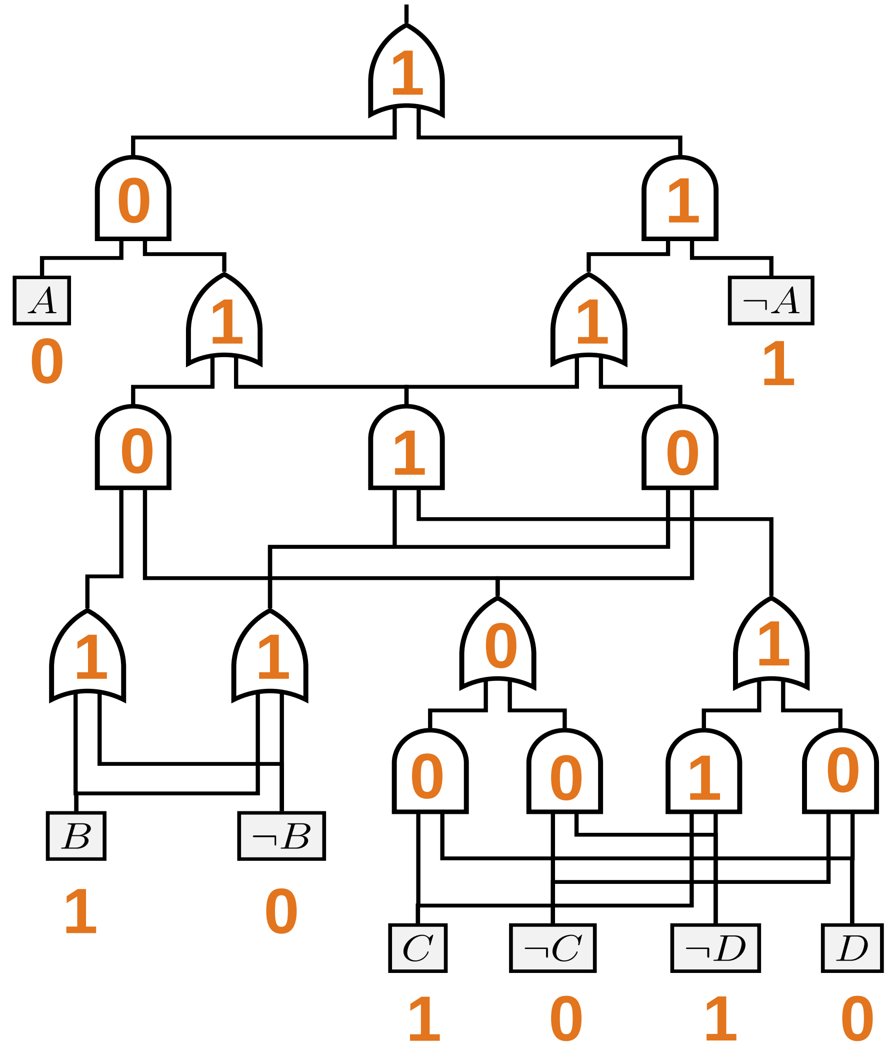


Reasoning Tool: Logical Circuits

Representation of logical sentences:

Input:

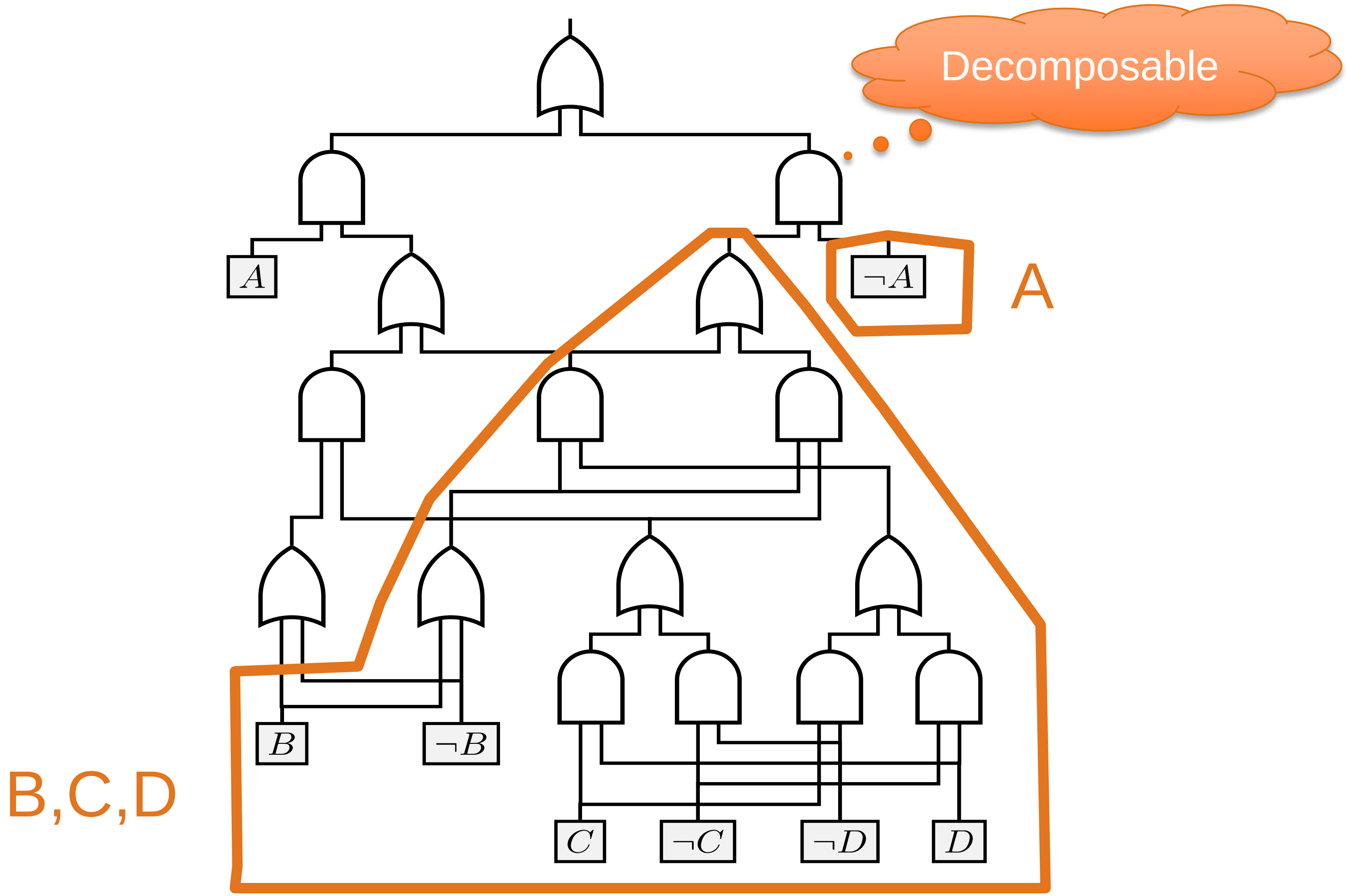
A	B	C	D
0	1	1	0



Tractable for Logical Inference

- Is there a solution? (SAT)
  - SAT($\alpha \lor \beta$) iff SAT(α) or SAT(β)   (*always*)
  - SAT($\alpha \land \beta$) iff ***???***

Decomposable Circuits

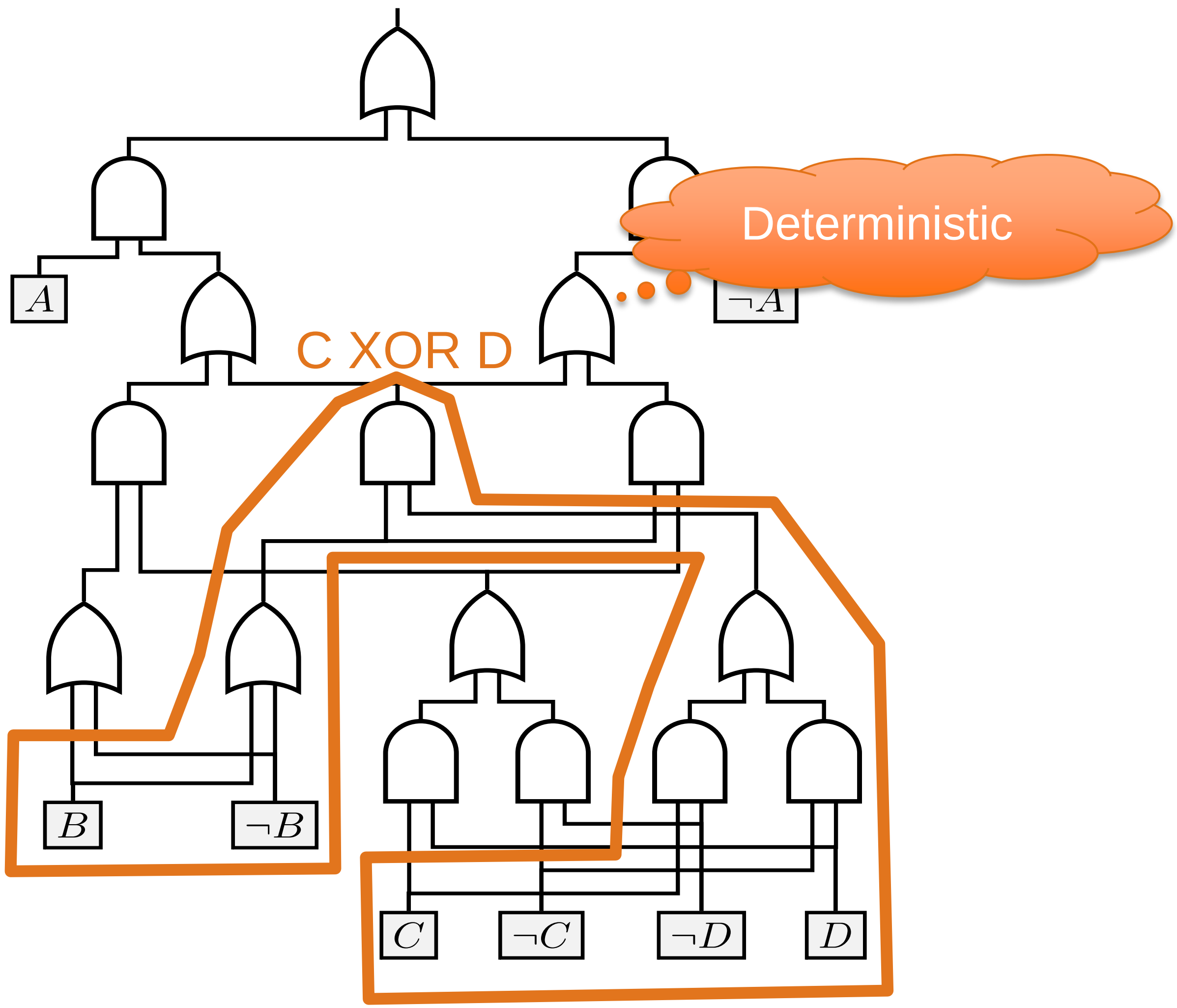


Tractable for Logical Inference

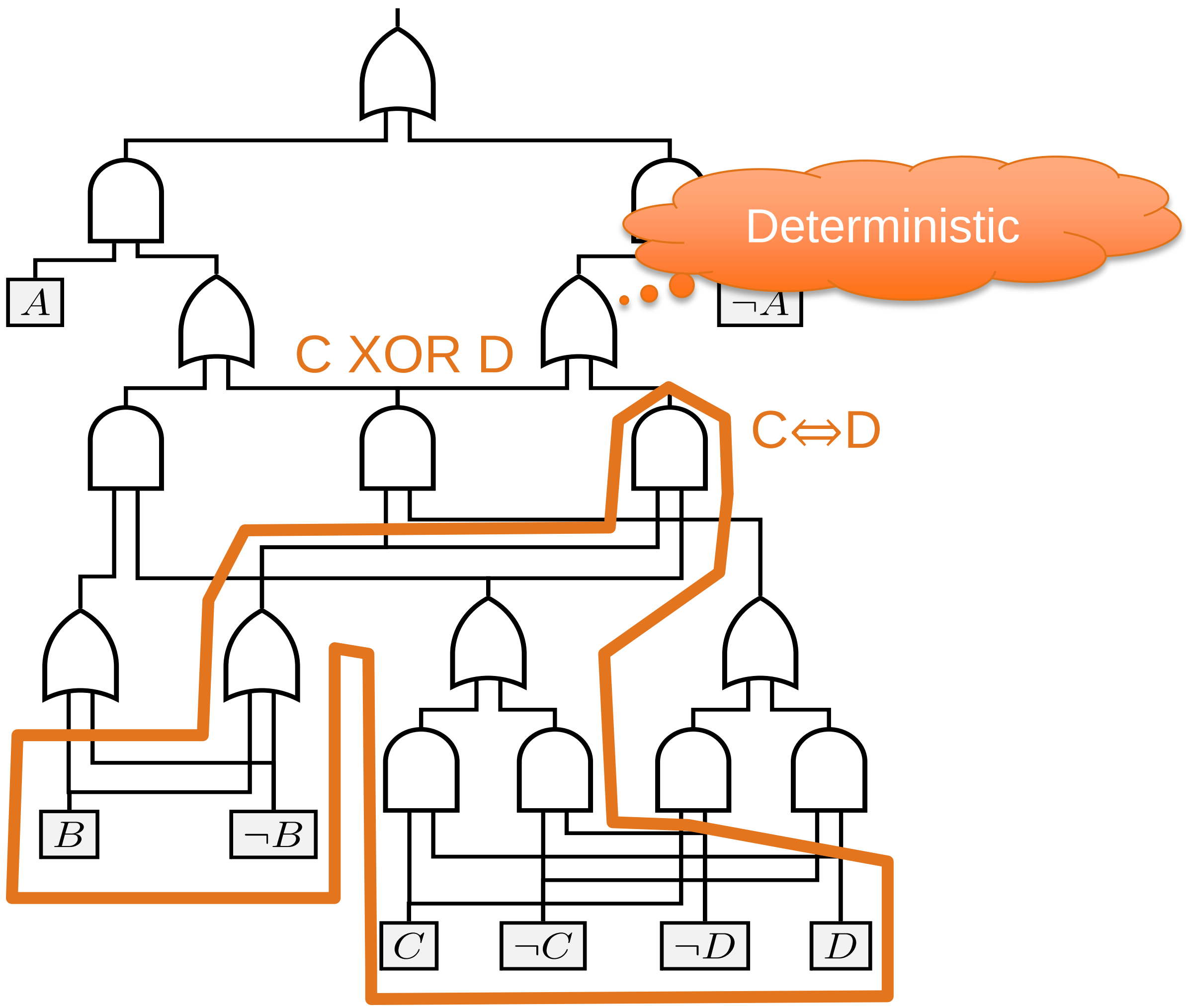
- Is there a solution? (SAT) ✓
  - SAT($\alpha \vee \beta$) iff SAT(α) or SAT(β) (*always*)
  - SAT($\alpha \wedge \beta$) iff SAT(α) and SAT(β) (*decomposable*)
- How many solutions are there? (#SAT)

- Complexity linear in circuit size ☺

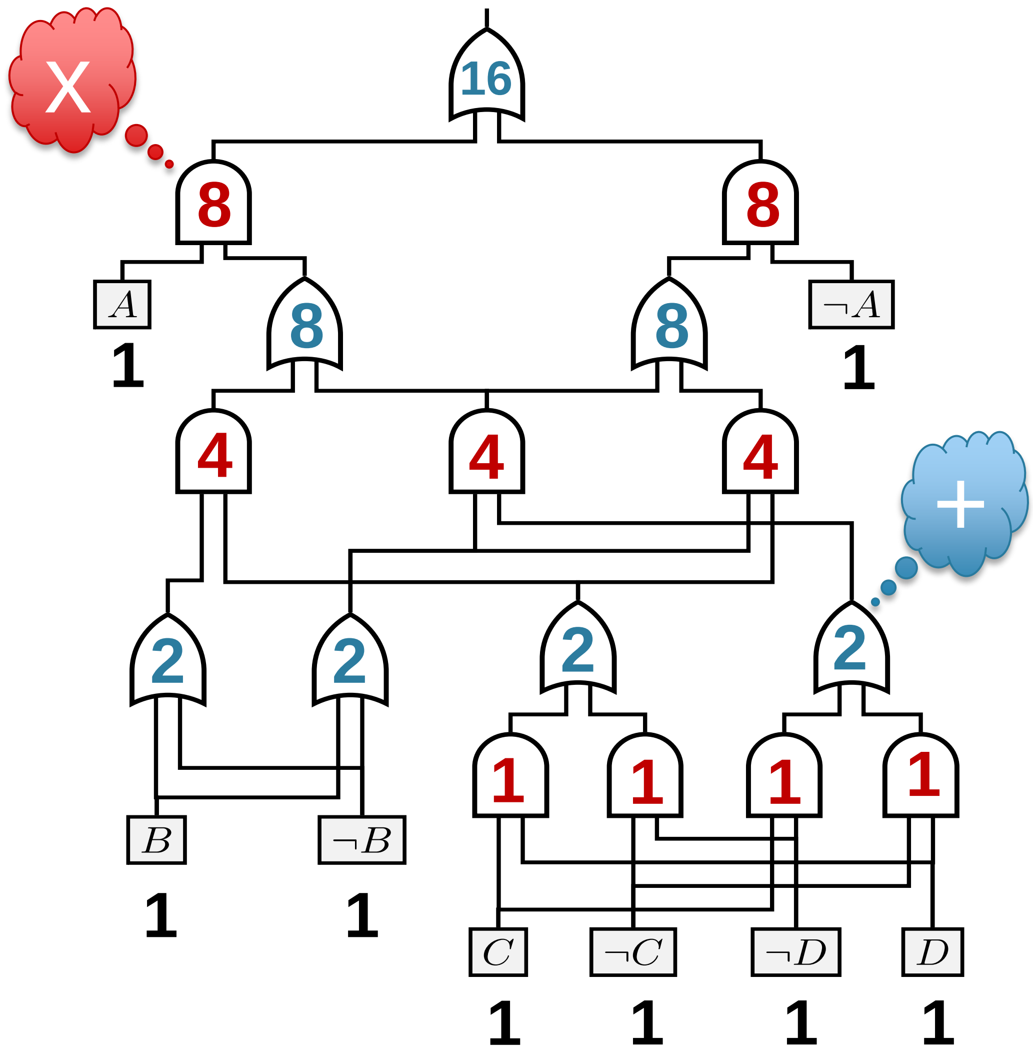
Deterministic Circuits



Deterministic Circuits



How many solutions are there? (#SAT)



Tractable for Logical Inference

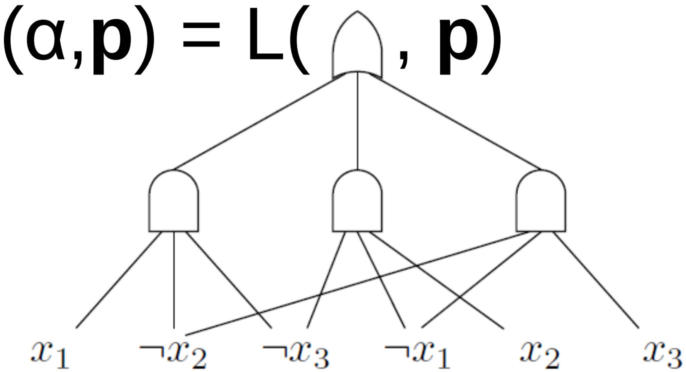
- Is there a solution? (SAT) ✓
- How many solutions are there? (#SAT) ✓
- Conjoin, disjoin, equivalence checking, etc. ✓
- Complexity linear in circuit size ☺

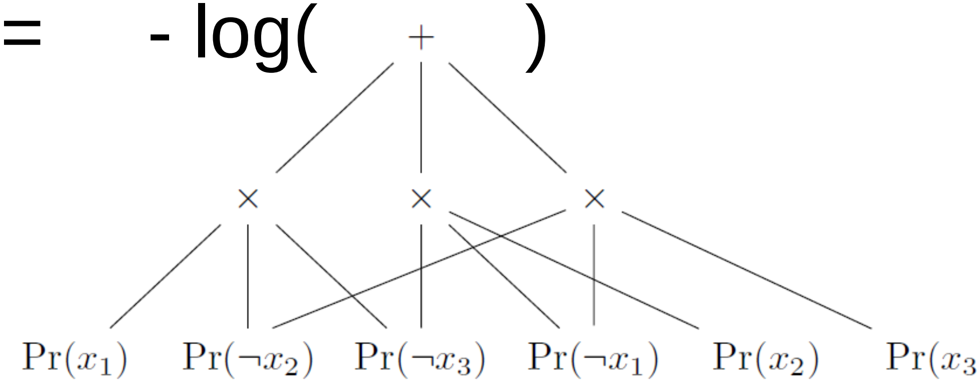
- Compilation into circuit by
  - ↓ exhaustive SAT solver
  - ↑ conjoin/disjoin/negate

[Darwiche and Marquis, JAIR 2002]

How to Compute Semantic Loss?

- In general: #P-hard ☹
- With a logical circuit for α: Linear ☺
- Example: exactly-one constraint:

L(α,**p**) = L(

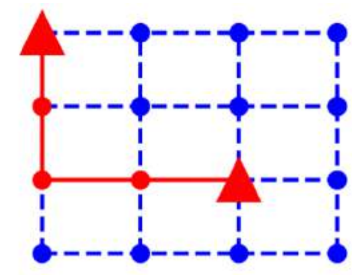
, **p**) = - log(

)

- *Why?* Decomposability and determinism!

Predict Shortest Paths

Add semantic loss
for path constraint



Test accuracy %	Coherent	Incoherent	Constraint
5-layer MLP	5.62	**85.91**	6.99
Semantic loss	**28.51**	83.14	**69.89**

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

Are individual edge predictions correct?

Is output a path?

(same conclusion for predicting sushi preferences, see paper)

Conclusions 1

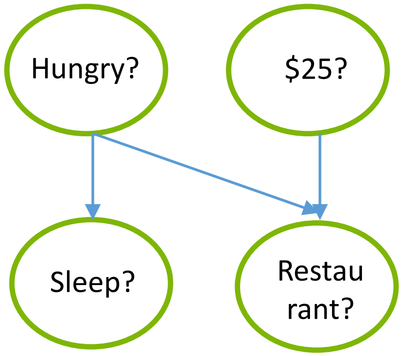
- Knowledge is (hidden) everywhere in ML
- Semantic loss makes logic differentiable
- Performs well semi-supervised
- Requires hard reasoning in general
  - Reasoning can be encapsulated in a circuit
  - No overhead during learning
- Performs well on structured prediction
- A little bit of reasoning goes a long way!

Outline

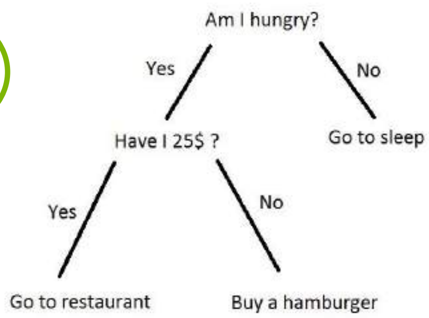
1. The AI dilemma: logic vs. learning
2. Deep learning with symbolic knowledge
3. Efficient reasoning during learning
4. **New machine learning formalisms**
5. Statistical relational learning (tutorial)

Another False Dilemma?

Classical AI Methods

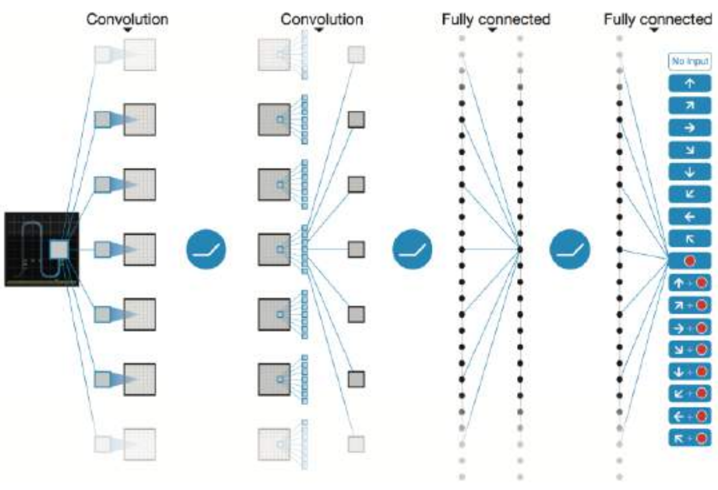


...



Clear Modeling Assumption
Well-understood

Neural Networks

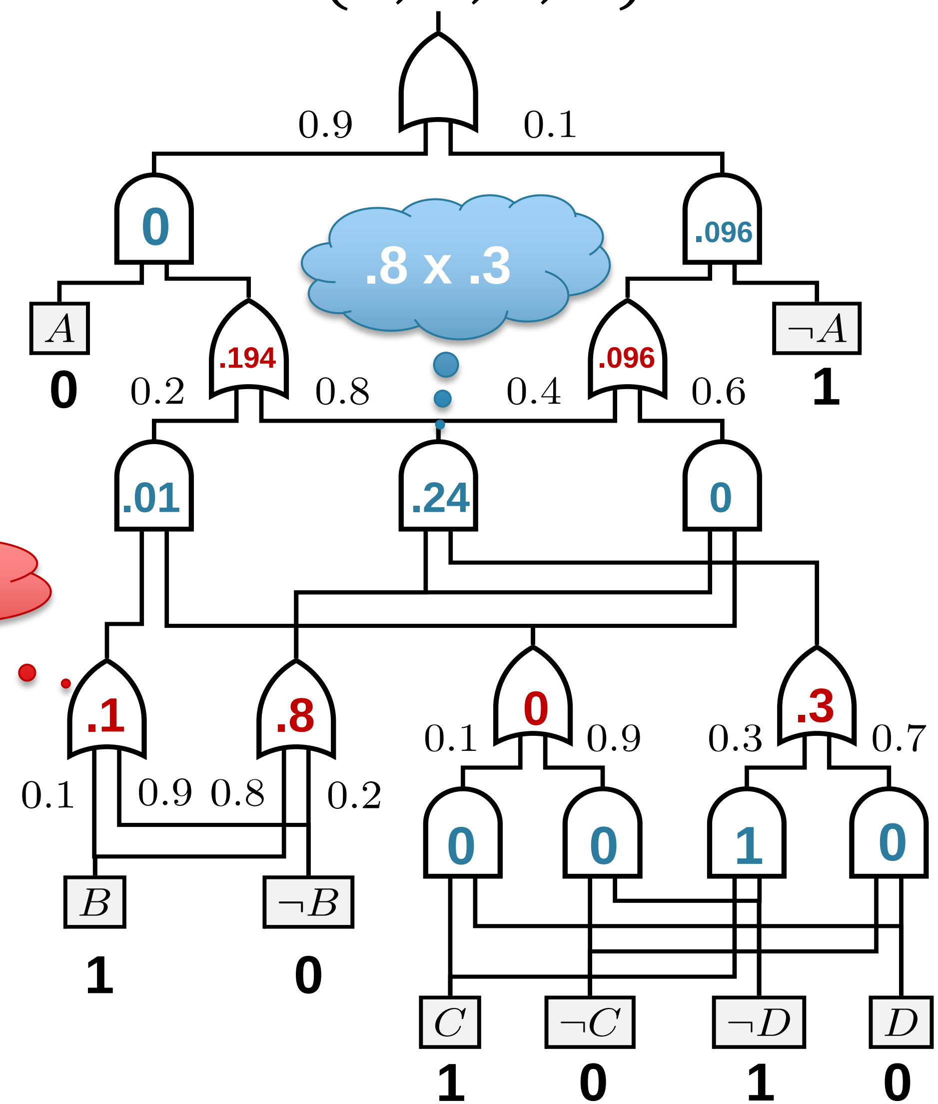


“Black Box”
Empirical performance

Probabilistic Circuits

SPNs, ACs
PSDDs, CNs

$$\mathbf{Pr}(A, B, C, D) = \mathbf{0.096}$$



Input:

A	B	C	D
0	1	1	0

Properties, Properties, Properties!

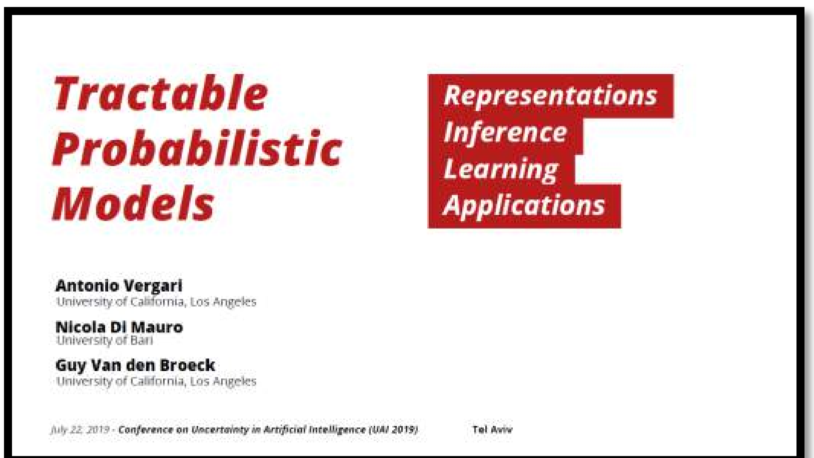
- Read conditional independencies from structure
- Interpretable parameters (XAI)
  (conditional probabilities of logical sentences)
- Closed-form parameter learning
- Efficient reasoning (linear ☺)
  - Computing **conditional probabilities** Pr(x|y)
  - **MAP inference**: most-likely assignment to x given y
  - Even much harder tasks: expectations, KLD, entropy, logical queries, decision making queries, etc.

Probabilistic Circuits: Performance

Density estimation benchmarks: tractable vs. intractable

Dataset	best circuit	BN	MADE	VAE
nltcs	**-5.99**	-6.02	-6.04	**-5.99**
msnbc	**-6.04**	**-6.04**	-6.06	-6.09
kdd2000	-2.12	-2.19	**-2.07**	-2.12
plants	**-11.84**	-12.65	12.32	-12.34
audio	-39.39	-40.50	-38.95	**-38.67**
jester	-51.29	**-51.07**	-52.23	-51.54
netflix	-55.71	-57.02	-55.16	**-54.73**
accidents	-26.89	**-26.32**	-26.42	-29.11
retail	**-10.72**	-10.87	-10.81	-10.83
pumbs*	-22.15	**-21.72**	-22.3	-25.16
dna	**-79.88**	-80.65	-82.77	-94.56
Kosarek	**-10.52**	-10.83	-	-10.64
Msweb	-9.62	-9.70	**-9.59**	-9.73

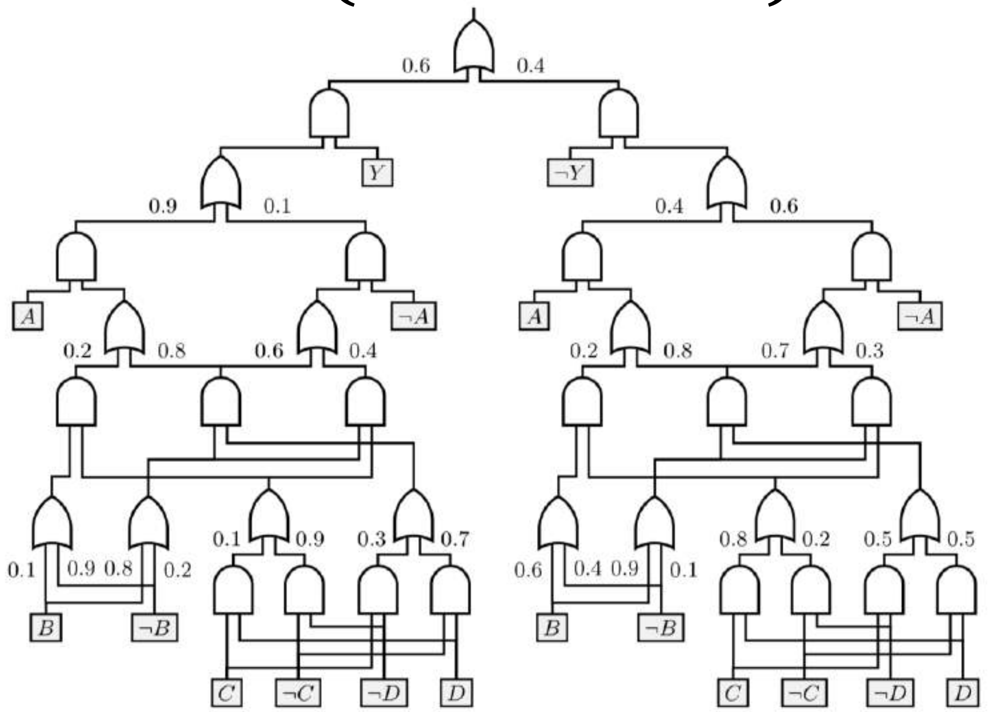
Dataset	best circuit	BN	MADE	VAE
Book	-33.82	-36.41	-33.95	**-33.19**
movie	-50.34	-54.37	-48.7	**-47.43**
webkb	-149.20	-157.43	-149.59	**-146.9**
cr52	-81.87	-87.56	-82.80	**-81.33**
c20ng	-151.02	-158.95	-153.18	**-146.90**
bbc	**-229.21**	-257.86	-242.40	-240.94
ad	-14.00	-18.35	**-13.65**	-18.81



But what if I only want to classify?

$$\Pr(Y|A, B, C, D)$$

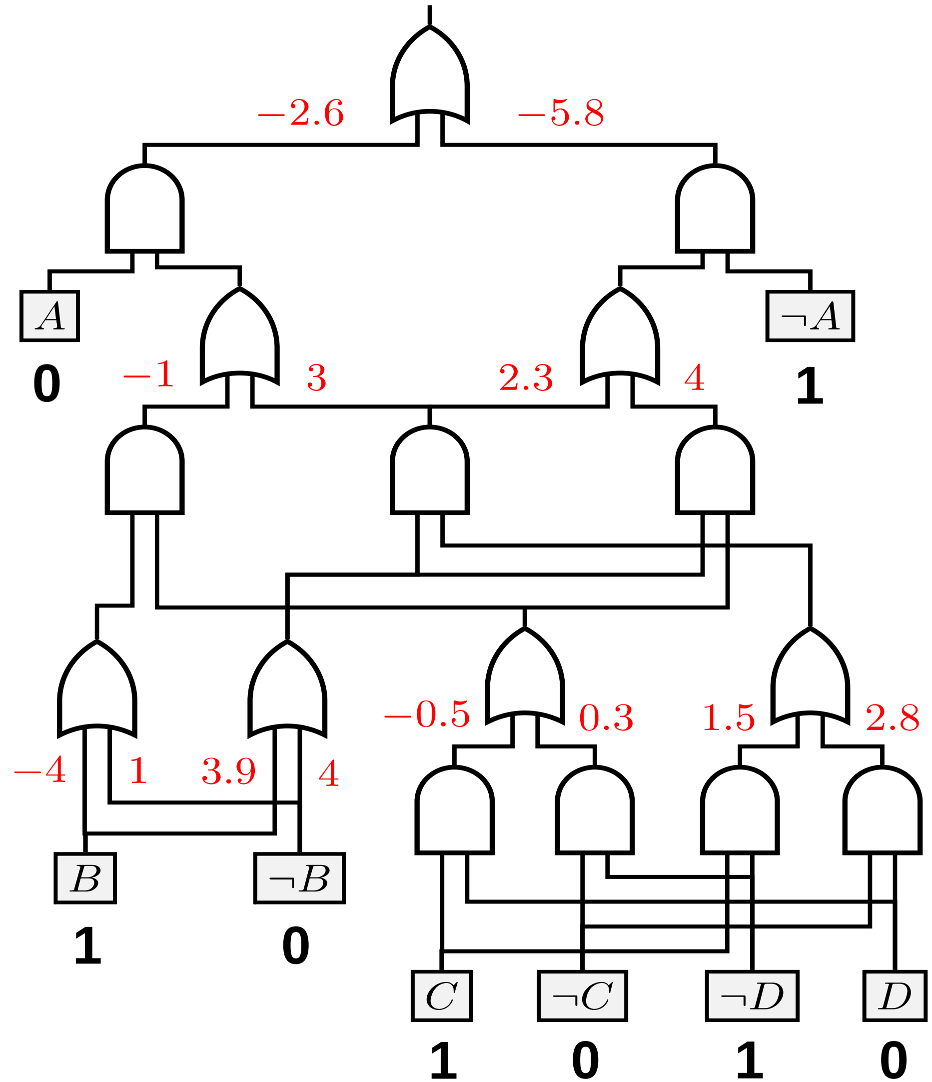
$$\cancel{\Pr(Y, A, B, C, D)}$$



Logistic Circuits

$$\Pr(Y = 1 \mid A, B, C, D)$$

$$= \frac{1}{1 + exp(-1.9)} = 0.869$$



Input:

A	B	C	D	$\Pr(Y \mid A, B, C, D)$
0	1	1	0	?

Learning Logistic Circuits

Parameter learning reduces to logistic regression:

$$\Pr(Y = 1 \mid \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x} \cdot \boldsymbol{\theta})}$$

Features associated with each wire
"Global Circuit Flow" features

Learning parameters θ is convex optimization!

Greedy structure learning (cf. decision trees)

Comparable Accuracy with Neural Nets

Accuracy % on Dataset	Mnist	Fashion
Baseline: Logistic Regression	85.3	79.3
Baseline: Kernel Logistic Regression	97.7	88.3
Random Forest	97.3	81.6
3-layer MLP	97.5	84.8
RAT-SPN (Peharz et al. 2018)	98.1	89.5
SVM with RBF Kernel	98.5	87.8
5-Layer MLP	99.3	89.8
Logistic Circuit (binary)	97.4	87.6
Logistic Circuit (real-valued)	99.4	91.3
CNN with 3 conv layers	99.1	90.7
Resnet (He et al. 2016)	99.5	93.6

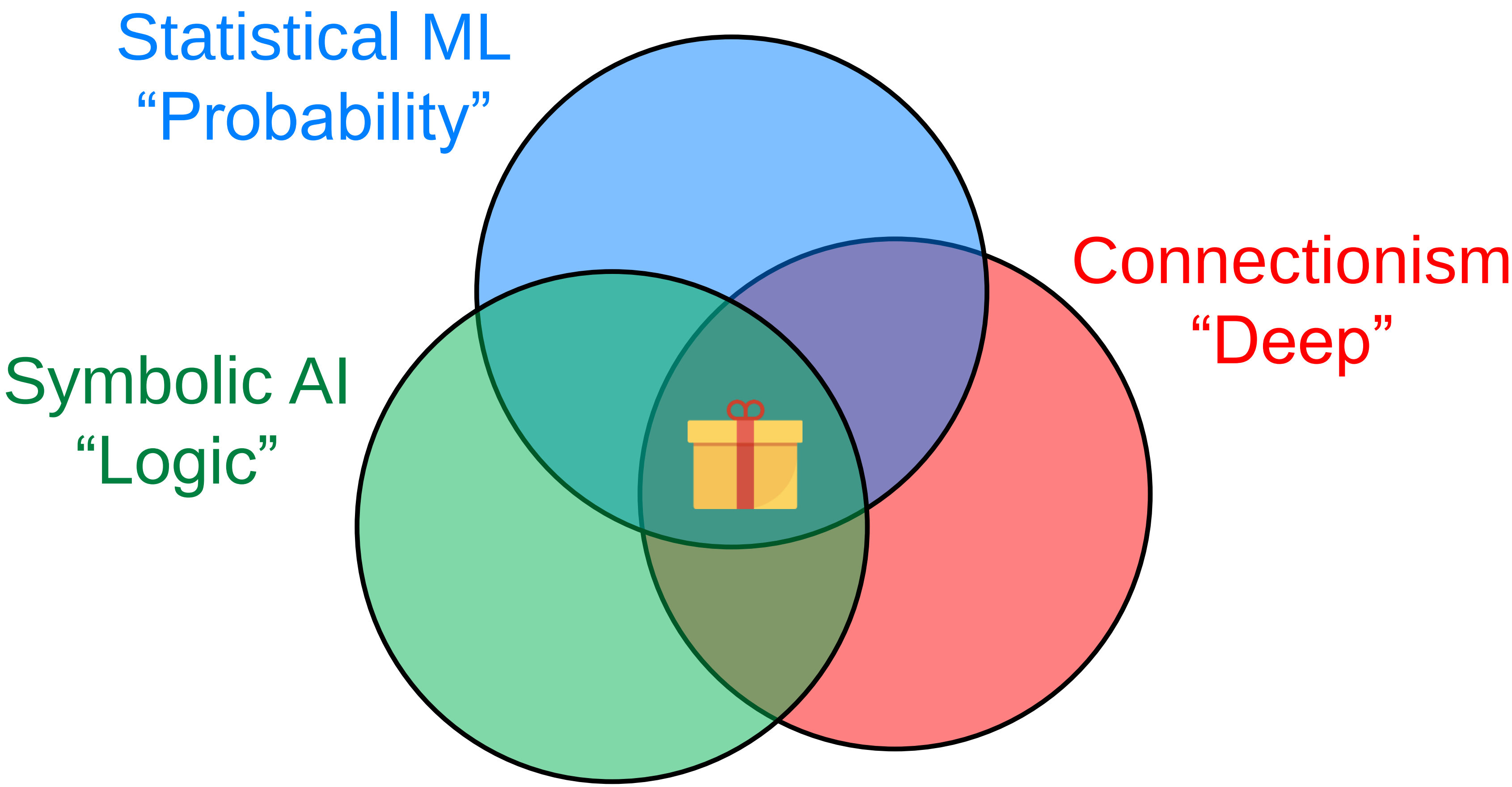
Significantly Smaller in Size

Number of Parameters	Mnist	Fashion
Baseline: Logistic Regression	<1K	<1K
Baseline: Kernel Logistic Regression	1,521 K	3,930K
Logistic Circuit (real-valued)	182K	467K
Logistic Circuit (binary)	268K	614K
3-layer MLP	1,411K	1,411K
RAT-SPN (Peharz et al. 2018)	8,500K	650K
CNN with 3 conv layers	2,196K	2,196K
5-layer MLP	2,411K	2,411K
Resnet (He et al. 2016)	4,838K	4,838K

Better Data Efficiency

Accuracy % with % of Training Data	MNIST			Fashion		
	100%	10%	2%	100%	10%	2%
5-layer MLP	99.3	**98.2**	94.3	89.8	86.5	80.9
CNN with 3 Conv Layers	99.1	98.1	95.3	90.7	87.6	83.8
Logistic Circuit (Binary)	97.4	96.9	94.1	87.6	86.7	83.2
Logistic Circuit (Real-Valued)	**99.4**	97.6	**96.1**	**91.3**	**87.8**	**86.0**

Probabilistic & Logistic Circuits



Reasoning about World Model + Classifier

“*Pure learning is brittle*”

bias, algorithmic fairness, interpretability, explainability, adversarial attacks, unknown unknowns, calibration, verification, missing features, missing labels, data efficiency, shift in distribution, general robustness and safety

fails to incorporate a sensible model of the world

- Given a learned predictor F(x)
- Given a probabilistic world model P(x)
- How does the world act on learned predictors?

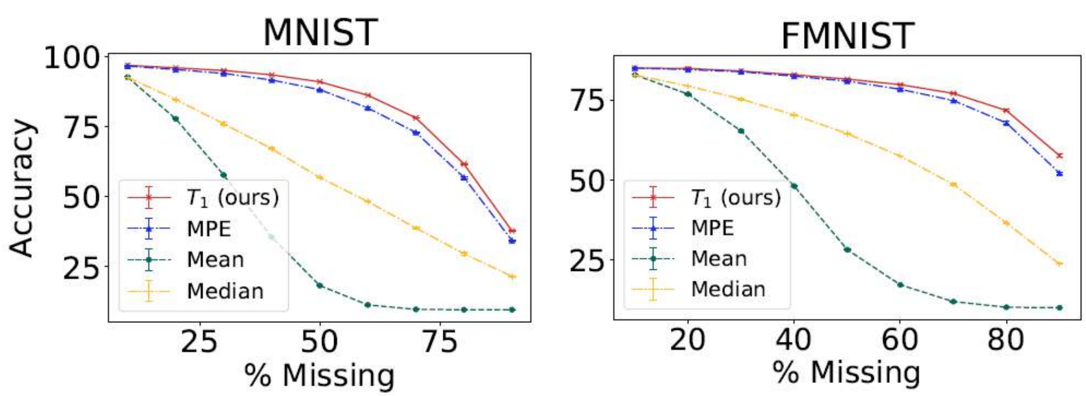
Can we solve these hard problems?

What to expect of classifiers?

- Missing features at prediction time
- What is expected prediction of F(x) in P(x)?

$$E_{\mathcal{F},P}(\mathbf{y}) = \mathop{\mathbb{E}}_{\mathbf{m}\sim P(\mathbf{M}|\mathbf{y})}\left[\mathcal{F}(\mathbf{ym})\right]$$

M: Missing features
y: Observed Features

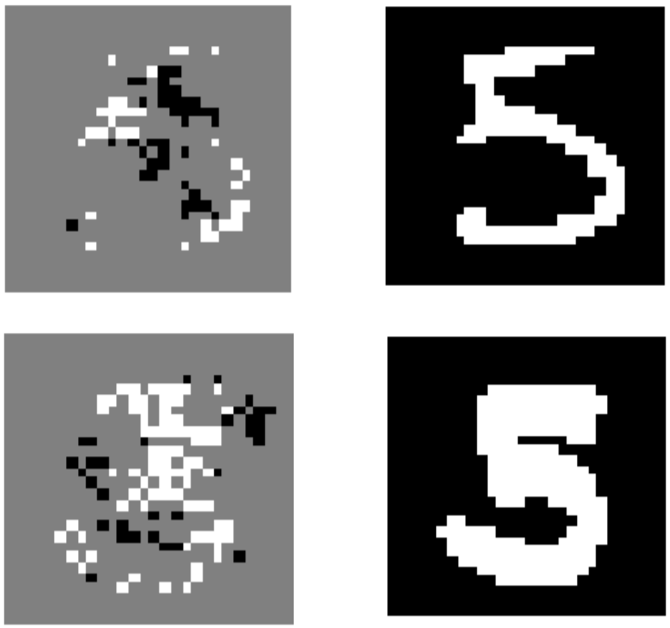


Explaining classifiers on the world

If the world looks like P(x),

then what part of the data is *sufficient* for

F(x) to make the prediction it makes?



Outline

1. The AI dilemma: logic vs. learning
2. Deep learning with symbolic knowledge
3. Efficient reasoning during learning
4. New machine learning formalisms
5. **Statistical relational learning (tutorial)**

Pure Logic | Probabilistic World Models | Pure Learning

High-Level Probabilistic Representations Reasoning, and Learning

Graphical Model Learning [Pearl 1988]

Medical Records

Bayesian Network

Name	Cough	Asthma	Smokes
Alice	1	1	0
Bob	0	0	0
Charlie	0	1	0
Dave	1	0	1
Eve	1	0	0

Brothers

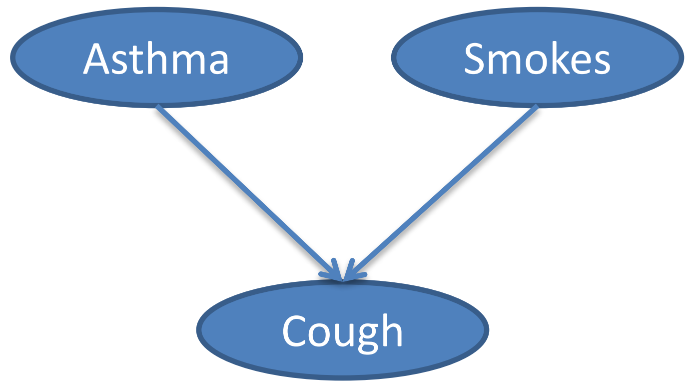
Friends

Frank	1	?	?

Big data

Frank	1	0.3	0.2

Frank	1	0.2	0.6

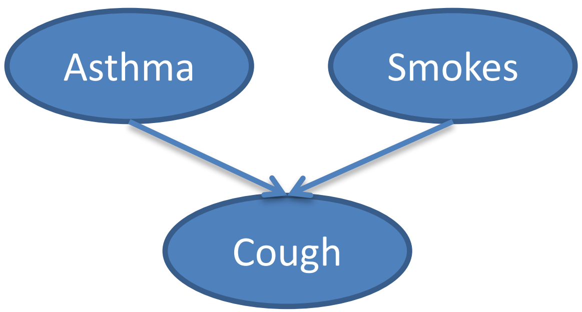


Rows are **independent** during learning and inference!

Statistical Relational Representations

Augment graphical model with relations between entities (rows).

Intuition



+ Friends have similar smoking habits

+ Asthma can be hereditary

Markov Logic

2.1 Asthma ⇒ Cough

3.5 Smokes ⇒ Cough

1.9 Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y)

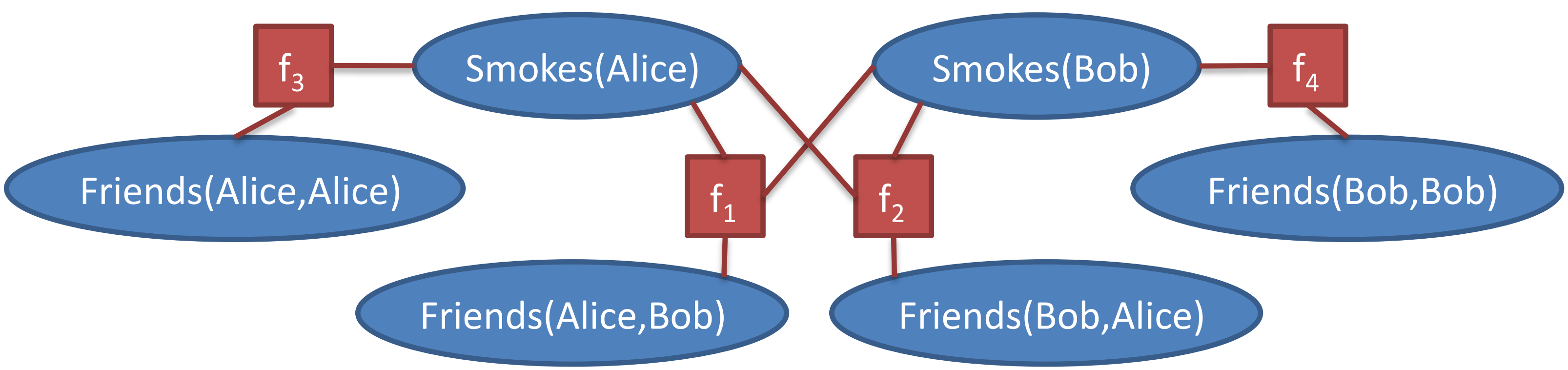
1.5 Asthma (x) ∧ Family(x,y) ⇒ Asthma (y)

Equivalent Graphical Model

- Statistical relational model (e.g., MLN)

1.9 Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y)

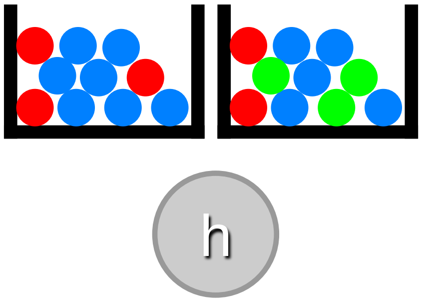
- Ground atom/tuple = **random variable** in {true,false}
  e.g., Smokes(Alice), Friends(Alice,Bob), etc.
- Ground formula = **factor** in propositional factor graph



Relational PGMs

- Markov logic
- Probabilistic soft logic (relaxation)
  - Random variables become continuous degrees of truth
  - Inference by convex optimization
  - *Talk to Angelika*
- Relational dependency networks
  - Learn local relational models that define a sampler
  - *Talk to Sriraam*
- Light on logic, heavy on PGMs

Probabilistic Logic Programming

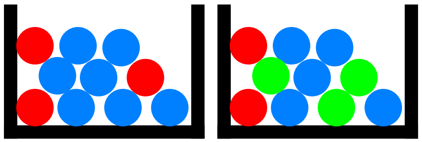


- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
0.4 :: heads.
```

probabilistic fact: heads is true with probability 0.4 (and false with 0.6)

Probabilistic Logic Programming



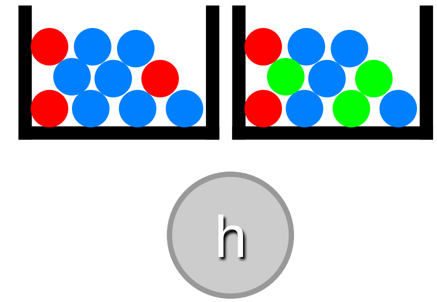
- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
0.4 :: heads.
```

annotated disjunction: first ball is red with probability 0.3 and blue with 0.7

```
0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
```

Probabilistic Logic Programming



- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

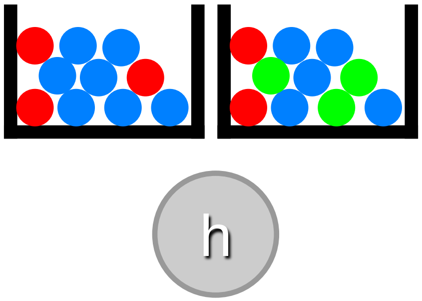
```
0.4 :: heads.
```

annotated disjunction: first ball is red with probability 0.3 and blue with 0.7

```
0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green);
                   0.5 :: col(2,blue) <- true.
```

annotated disjunction: second ball is red with probability 0.2, green with 0.3, and blue with 0.5

Probabilistic Logic Programming

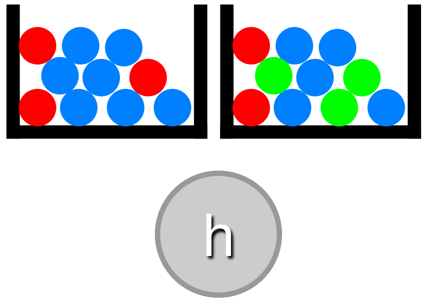


- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green);
                   0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
```

logical rule encoding background knowledge

Probabilistic Logic Programming

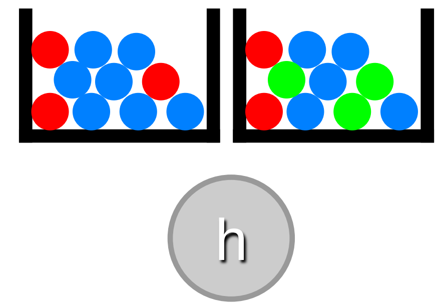


- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green);
                   0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

logical rule encoding background knowledge

Probabilistic Logic Programming



- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green);
                   0.5 :: col(2,blue) <- true
```

probabilistic choices

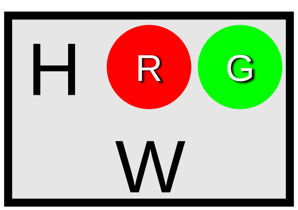
```
win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

consequences

Possible Worlds

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

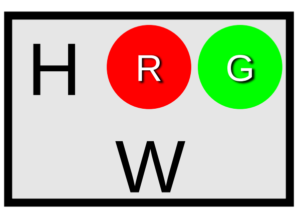
$0.4 \times 0.3 \times 0.3$



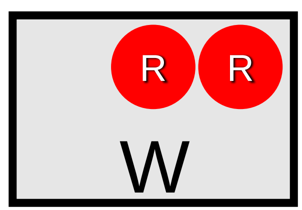
Possible Worlds

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

$0.4 \times 0.3 \times 0.3$



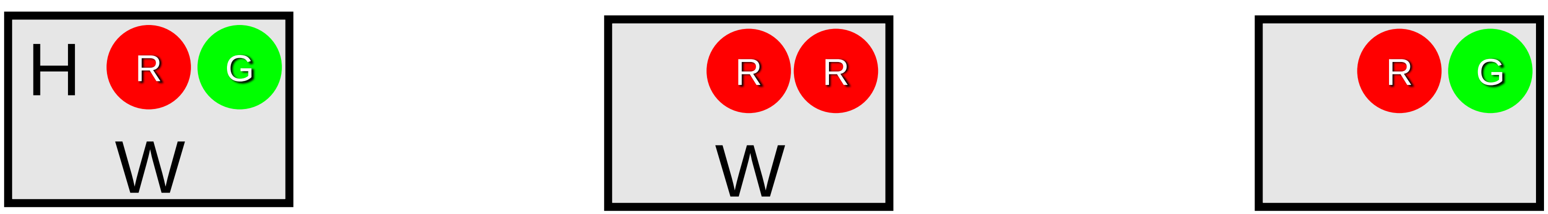
$(1-0.4) \times 0.3 \times 0.2$



Possible Worlds

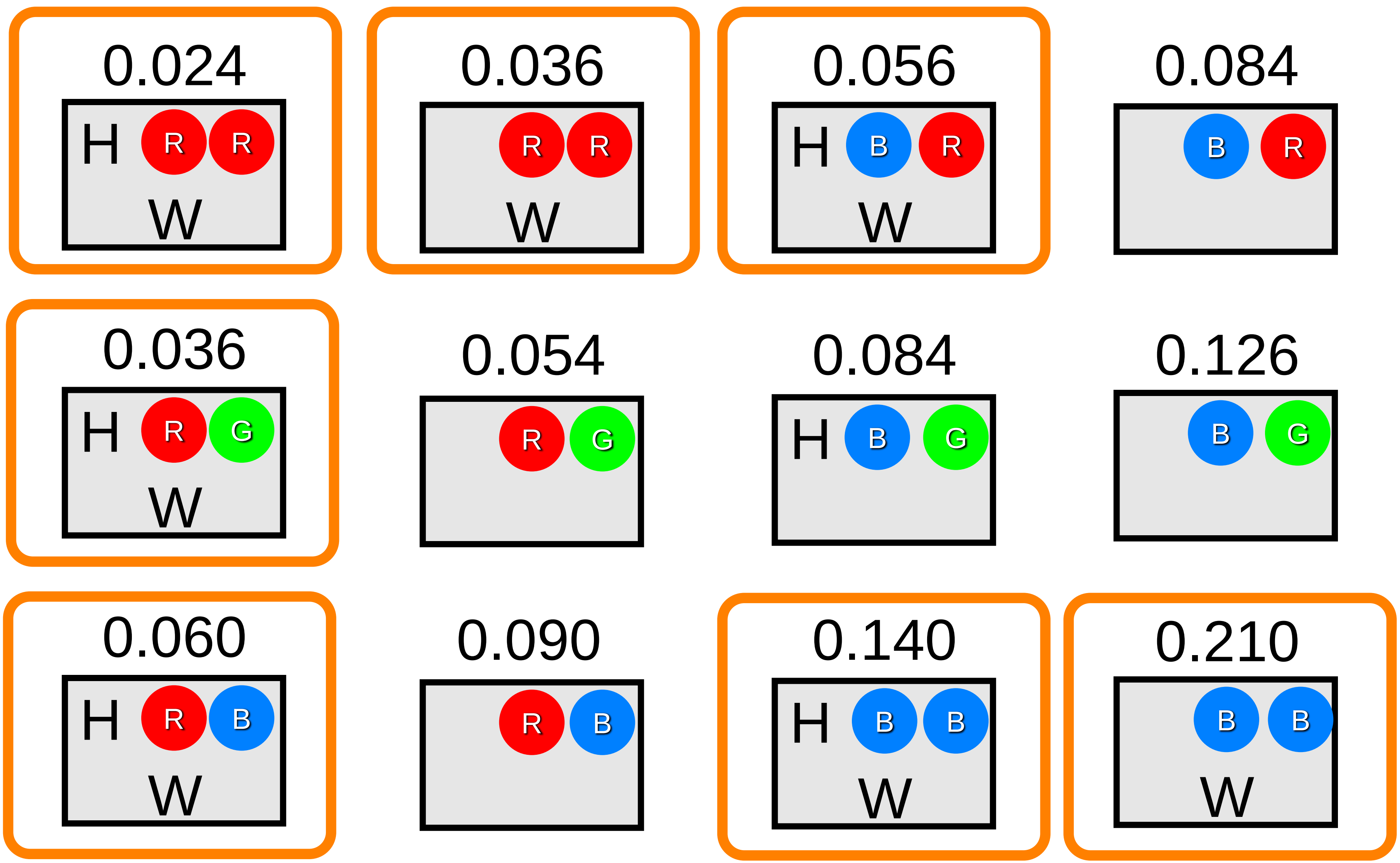
```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

$0.4 \times 0.3 \times 0.3$ $\quad$ $(1-0.4) \times 0.3 \times 0.2$ $\quad$ $(1-0.4) \times 0.3 \times 0.3$



$$P(\underline{\text{win}}) = \sum ? = 0.562$$

Marginal Probability



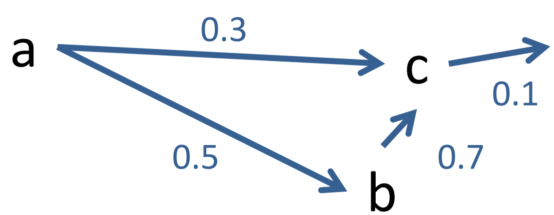
Probabilistic (Logic) Programming

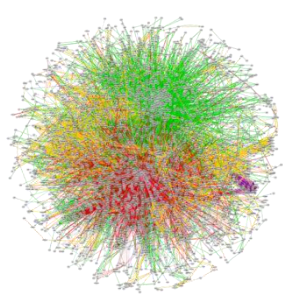
Discrete probabilistic reachability program:

Logic Program (ProbLog) = Functional Program (Scala-like)

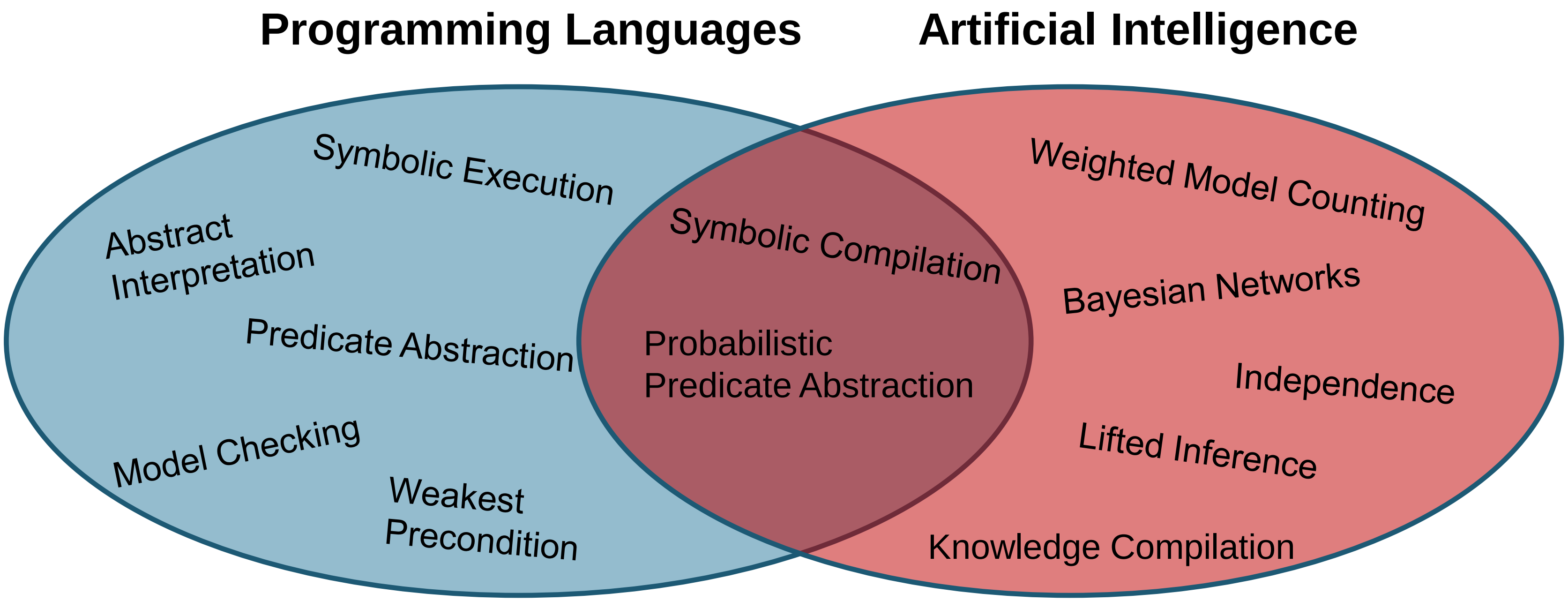
```
path(X,Y) :- edge(X,Y).
path(X,Y) :- edge(X,Z),
             path(Z,Y).
edge(X,Y) :- …random vars…
```

```
def path(start,end,visited=List())={
    if(start == end)
        return true
    if(visited.contains(start))
        return false
    return start.neighbors.exists{
        path(_,end,(visited+start))
    }
}
nodeA.neighbors = …random vars…
nodeB.neighbors = …random vars…
```





Probabilistic Programming Research



Probabilistic Databases

- Tuple-independent probabilistic database

Scientist

x	P
Erdos	0.9
Einstein	0.8
Pauli	0.6

Coauthor

x	y	P
Erdos	Renyi	0.6
Einstein	Pauli	0.7
Obama	Erdos	0.1

- Learned from the web, large text corpora, ontologies, etc., using **statistical** machine learning.

[Suciu'11]

Pure Logic **Probabilistic World Models** **Pure Learning**

<u>Probabilistic Logic Programming</u>

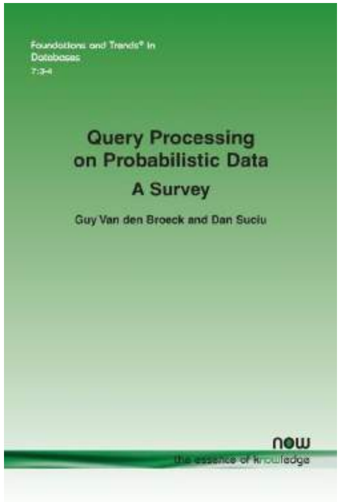
Prolog meets probabilistic AI

Talk to Luc, Angelika, Vaishak, Kristian, etc.

<u>Probabilistic Databases</u>

Databases meets probabilistic AI

Talk to Dan, Dan, Ismail, Carsten, etc.



<u>Weighted Model Integration</u>

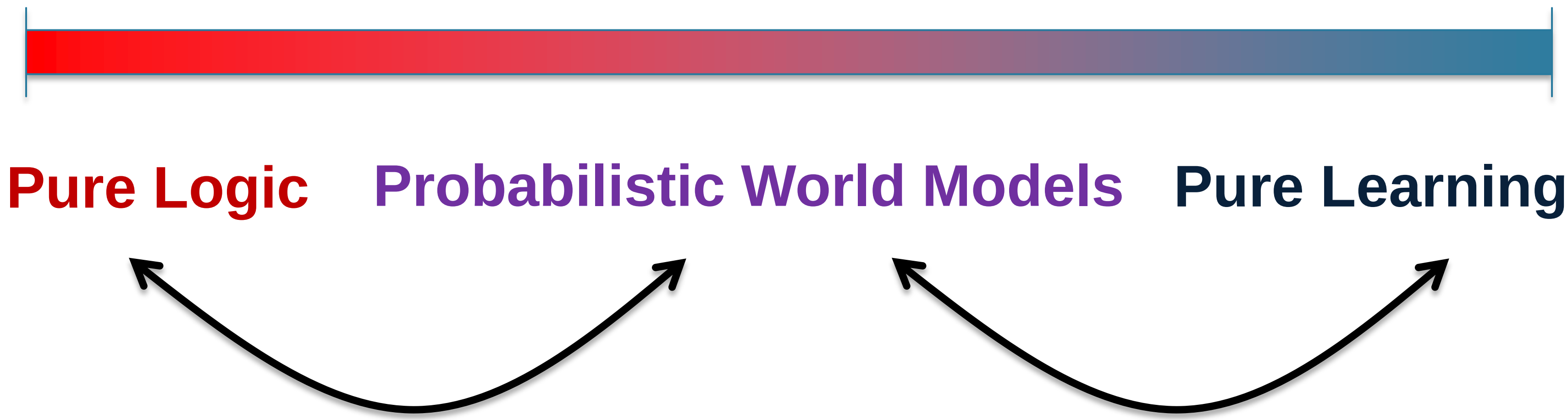
SAT modulo theories meets probabilistic AI

Talk to Vaishak

Approximate Lifted Probabilistic Inference

- Message passing symmetries
  - Identify which nodes will receive identical messages throughout algorithm
  - Fractional automorphisms
  - Found by color passing
  - *Talk to Kristian, Sriraam, Martin Grohe*
- Lifted MCMC
  - Compute exact automorphisms
  - Fun with group theory tools
  - Make MCMC samplers mix exponentially faster

Conclusions



Bring high-level representations, general knowledge, and efficient high-level reasoning to probabilistic models

Bring back models of the world, supporting new tasks, and reasoning about what we have learned, without compromising learning performance

Conclusions

- There is a lot of value in working on pure logic, pure learning
- But we can do more
  by finding a synthesis, a confluence

Let’s get rid of this false dilemma…

Thanks