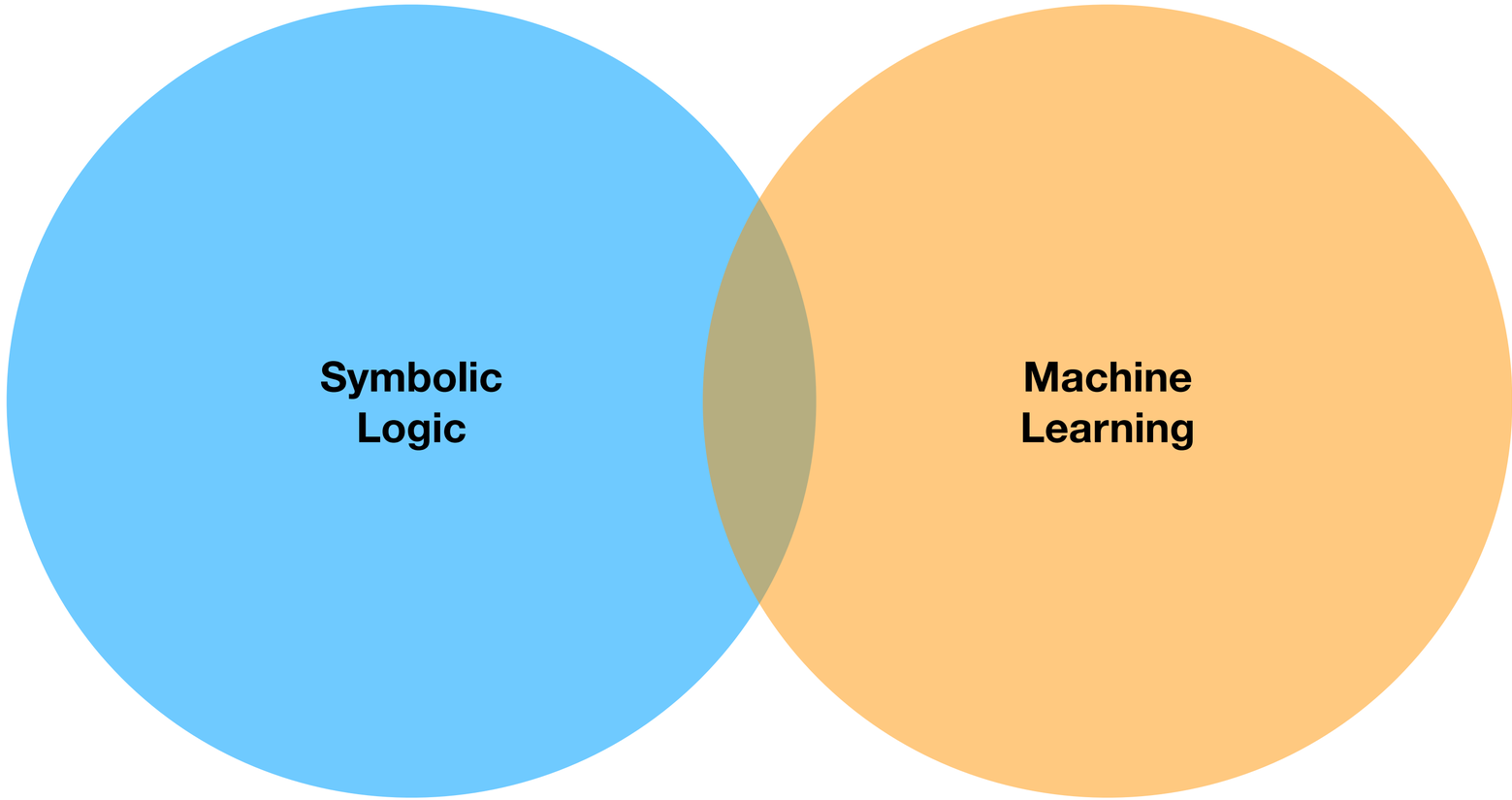
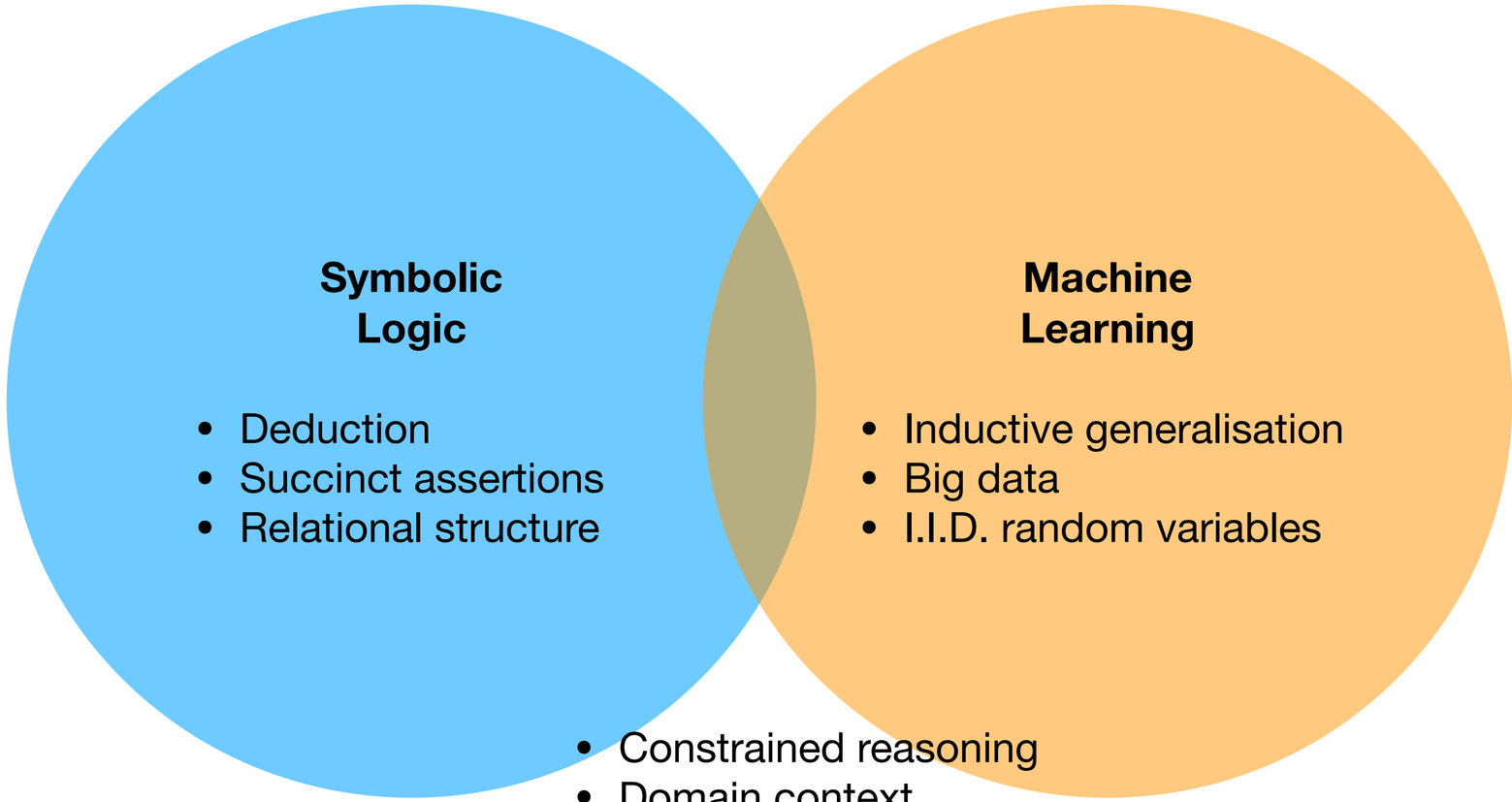
Logic meets Learning

Vaishak Belle
University of Edinburgh & Alan Turing Institute

What’s on for today?



Why?



- Constrained reasoning
- Domain context
- Computability
- Interpretability
- Ethics, …

What will we cover?

- Quick illustrative examples of logical reasoning in various domains: protein graph database, electronic health records, social networks, inhibition effects, modeling unknowns in robotics
- Foundations of probabilistic logical inference: weighted model counting
- Tractable models, causality and counterfactuals
  - Fairness and moral reasoning
- Model counting in continuous + countably infinite domains
- Learning of programs and tractable models
- Other applications of model counting + abstraction
- Neuro-symbolic landscape + loss function approach for deep learning
- Conclusions

Our approach: after *mentioning* illustrative examples, we will deal with toy examples.

Many works focus on applications, but this tutorial is about uncovering the key ideas and questions

Certain details are technical, but hope you walk away with the essence

Illustrative examples of logic, probability and learning

Gene expressions & protein interactions

Probabilistic subgraphs

Biomine

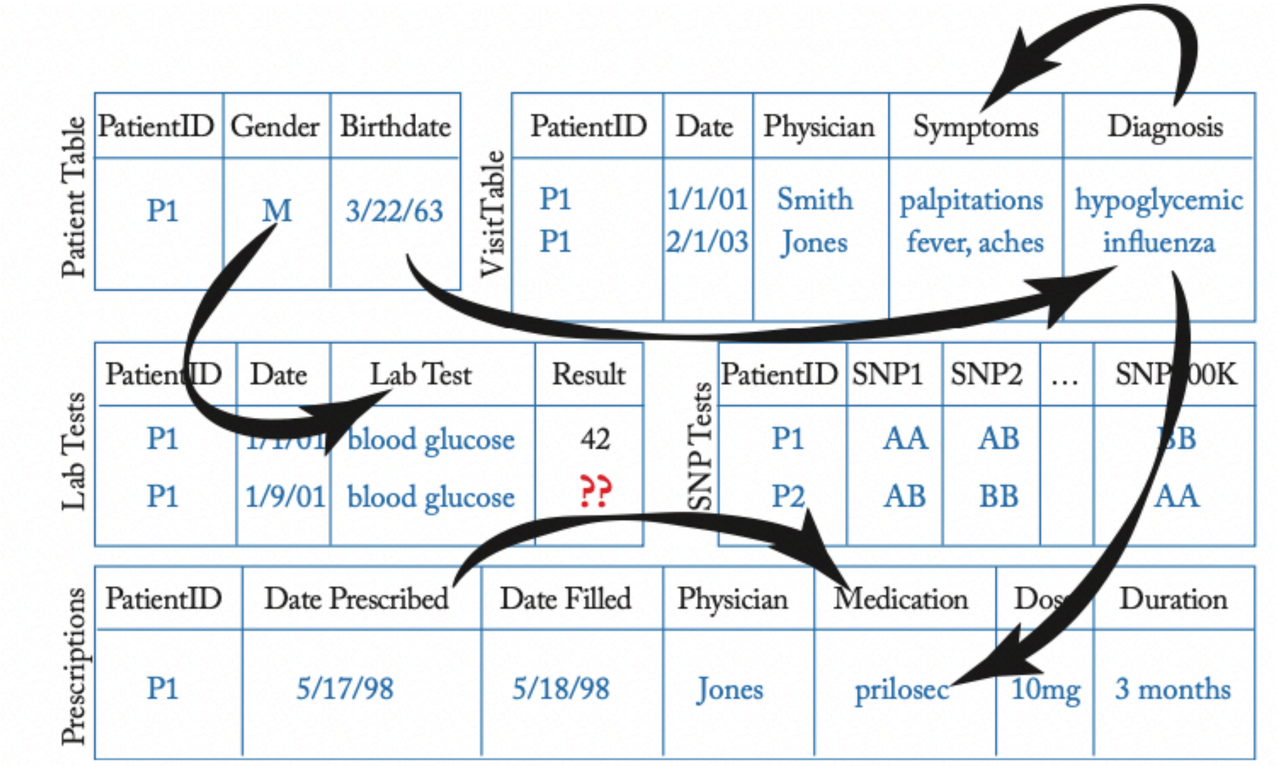
Probability and subgraph questions

- Is gene X involved in disease *Y*?
- What is the probability of involvement?
- Which genes are similar to *X* conditioned on *Y*?
- Which subgraphs are most relevant for studying *Y*?

De Raedt (2009)

Electronic health records with noisy and missing data

Relational models



Patient Table

PatientID	Gender	Birthdate
P1	M	3/22/63

VisitTable

PatientID	Date	Physician	Symptoms	Diagnosis
P1	1/1/01	Smith	palpitations	hypoglycemic
P1	2/1/03	Jones	fever, aches	influenza

Lab Tests

PatientID	Date	Lab Test	Result
P1	1/1/01	blood glucose	42
P1	1/9/01	blood glucose	??

SNP Tests

PatientID	SNP1	SNP2	…	SNP 100K
P1	AA	AB		BB
P2	AB	BB		AA

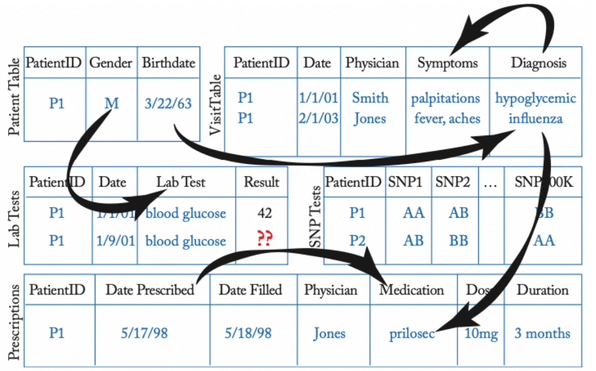
Prescriptions

PatientID	Date Prescribed	Date Filled	Physician	Medication	Dos	Duration
P1	5/17/98	5/18/98	Jones	prilosec	10mg	3 months

De Raedt et al, Morgan Claypool, 2016.

Conditional queries with constraints

- Relational model
- Longitudinal study over many different time periods such as 0, 5, 10 years
- Model identifies complex interaction between gender, age and risk factors at different years of the longitudinal study



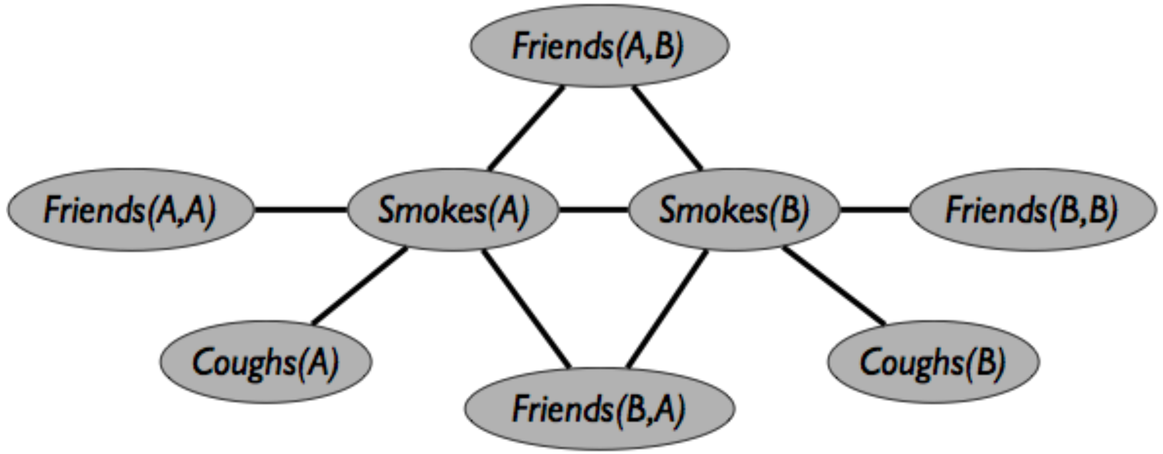
$$0.79 = P(risk(Person) \mid sex(Person, male) \wedge (35 \leq age(Person, year\,7) < 45) \wedge \neg(0 \leq ldl(Person, year\,0) < 100))$$

Social interactions

Finite domain relational logic

$\theta_1 \quad \forall x, y[Smokes(x) \wedge Friends(x, y) \supset Smokes(y)]$

$\theta_2 \quad \forall x[Smokes(x) \supset Coughs(x)]$



- **Evidential:**

$\Pr(Smokes(A) \mid Friends(A, B) \wedge Smokes(B))$

- **Adding new knowledge:**

$\forall x, y[Smokes(x) \wedge Family(x, y) \supset Smokes(y)]$

Markov logic networks

Spread of disease

Query-driven grounding

- Given a set of initially infected people and a graph of connections between individuals in the population, the goal is to predict the spread of the disease
- Two people are in regular contact with each other and one is infected, 0.6 probability of second getting infected

```
1  person(a).
2  person(b).
3
4  0.1::initialInf(X) :- person(X).
5  0.1::contact(X,Y) :- person(X), person(Y).
6
7  inf(X)      :- initialInf(X).
8  0.6::inf(X) :- contact(X, Y), inf(Y).
9
10 query(inf(_)).
```

Evaluate

Query▼	Location	Probability
inf(a)	10:7	0.1054
inf(b)	10:7	0.1054

Probabilistic programming: https://dtai.cs.kuleuven.be/problog/

Making contagion deterministic

- Simply a matter of dropping probabilities on the rule
- Generally, (lifted) rules domain agnostic
  - Individuals and probabilities extracted from data

```
1  person(a).
2  person(b).
3
4  0.1::initialInf(X) :- person(X).
5  0.1::contact(X,Y) :- person(X), person(Y).
6
7  inf(X)      :- initialInf(X).
8  inf(X) :- contact(X, Y), inf(Y).
9
10 query(inf(_)).
```

Evaluate

Query▼	Location	Probability
inf(a)	10:7	0.109
inf(b)	10:7	0.109

Inhibition effects

- Parents independently influence a joint effect
- People may also contract the disease by travelling to particular locations
- Enabled using noisy-or gates

```
1  person(a).
2  person(b).
3
4  0.1::initialInf(X) :- person(X).
5  0.1::contact(X,Y) :- person(X), person(Y).
6  0.1::riskyTravel(X) :- person(X).
7
8  inf(X)      :- initialInf(X).
9  0.6::inf(X) :- contact(X, Y), inf(Y).
10 0.2::inf(X) :- riskyTravel(X).
11
12 query(inf(_)).
```

Evaluate

Query▼	Location	Probability
inf(a)	12:7	0.12424456
inf(b)	12:7	0.12424456

Contextualising influence

- Suppose that certain people are discovered to be especially susceptible
- Append to infection rule

```
1  person(a).
2  person(b).
3
4  0.1::initialInf(X) :- person(X).
5  0.1::contact(X,Y) :- person(X), person(Y).
6  0.1::riskyTravel(X) :- person(X).
7  0.1::susceptible(X) :- person(X).
8
9  inf(X)      :- initialInf(X).
10 0.6::inf(X) :- contact(X, Y), inf(Y), \+susceptible(X).
11 0.8::inf(X) :- contact(X, Y), inf(Y),   susceptible(X).
12 0.2::inf(X) :- riskyTravel(X).
13
14 query(inf(_)).
```

Evaluate

Query▼	Location	Probability
inf(a)	14:7	0.12445271
inf(b)	14:7	0.12445271

Audience questions:

1) What domains have you worked on?
2) Have you encountered the need for such constraints?
3) What combination of expert knowledge and data-driven learning was used?

Language understanding

Probabilistic logical languages serving as an intermediate representation

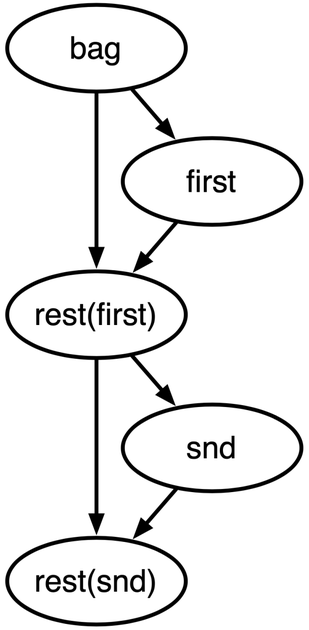
Probabilistic model

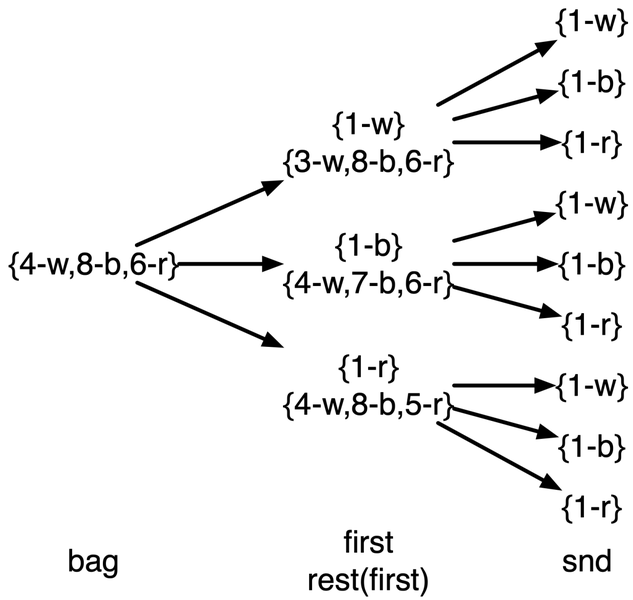
Q2: Mike has a bag of marbles with 4 white, 8 blue, and 6 red marbles. He pulls out one marble from the bag and it is red. What is the probability that the second marble he pulls out of the bag is white?

Conditioning (observation)

Query

From parsing to programs



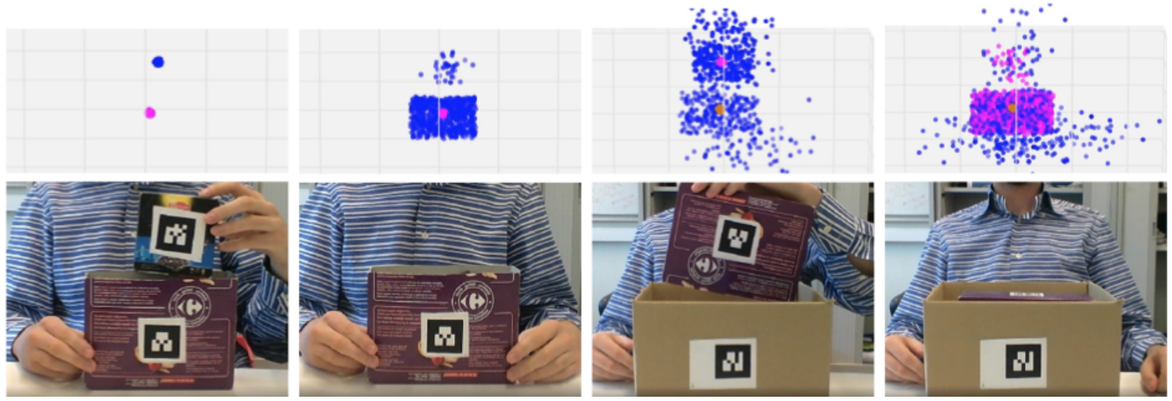



```
1  group(marbles).
2  size(marbles, 18).
3
4  given(exactly(8, marbles, blue)).
5  given(exactly(...
6
7  take(...
8
```

Dries et al. (2017)

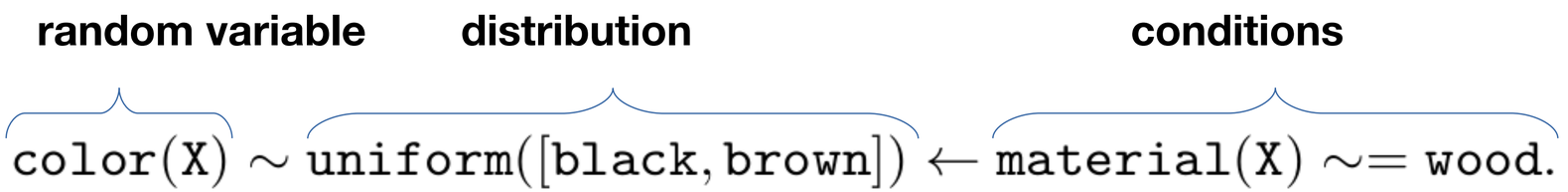
Occlusion modeling

Probabilistic logical languages serving as constrained particle filter



How can we model unknown objects and properties?

Unknown color

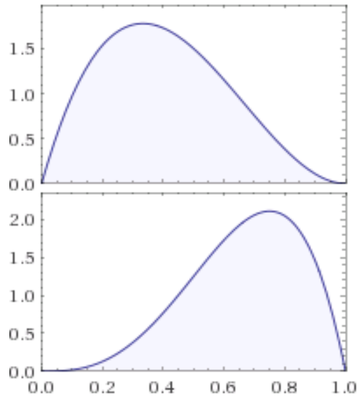


Unknown physical size

$$\texttt{material}(\texttt{X}) \sim \texttt{finite}([0.3:\texttt{wood}, 0.7:\texttt{metal}]) \leftarrow \texttt{between}(1, \texttt{N}, \texttt{X}).$$

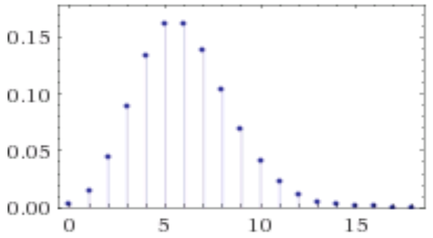
$$\texttt{size}(\texttt{X}) \sim \texttt{beta}(2,3) \leftarrow \texttt{material}(\texttt{X}) \sim= \texttt{metal}.$$

$$\texttt{size}(\texttt{X}) \sim \texttt{beta}(4,2) \leftarrow \texttt{material}(\texttt{X}) \sim= \texttt{wood}.$$



Unknown numbers

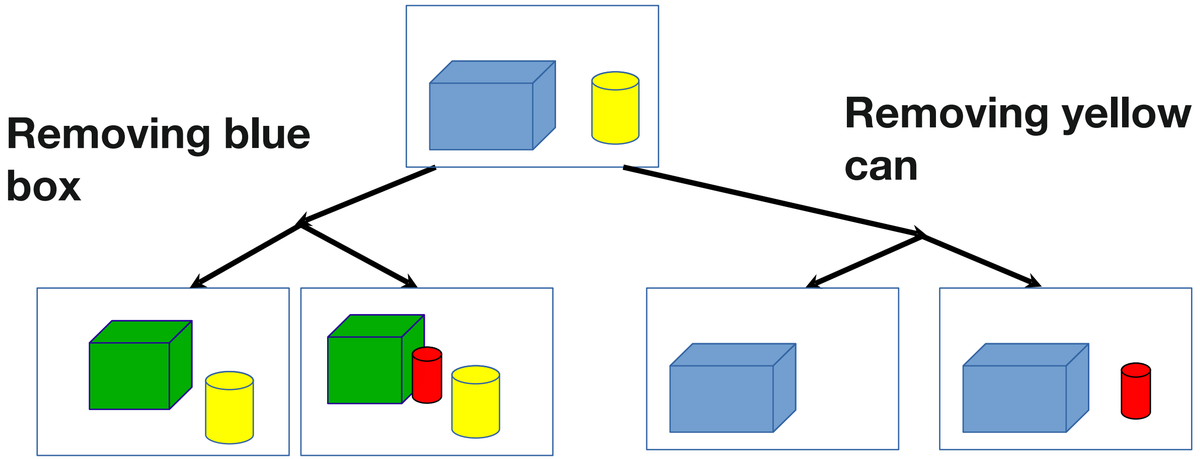
$\texttt{n} \sim \texttt{poisson}(6).$



(Infinite valued discrete distribution)

$$\texttt{material}(\texttt{X}) \sim \texttt{finite}([0.3 : \texttt{wood}, 0.7 : \texttt{metal}]) \leftarrow \texttt{n} \sim= \texttt{N}, \texttt{between}(1, \texttt{N}, \texttt{X}).$$

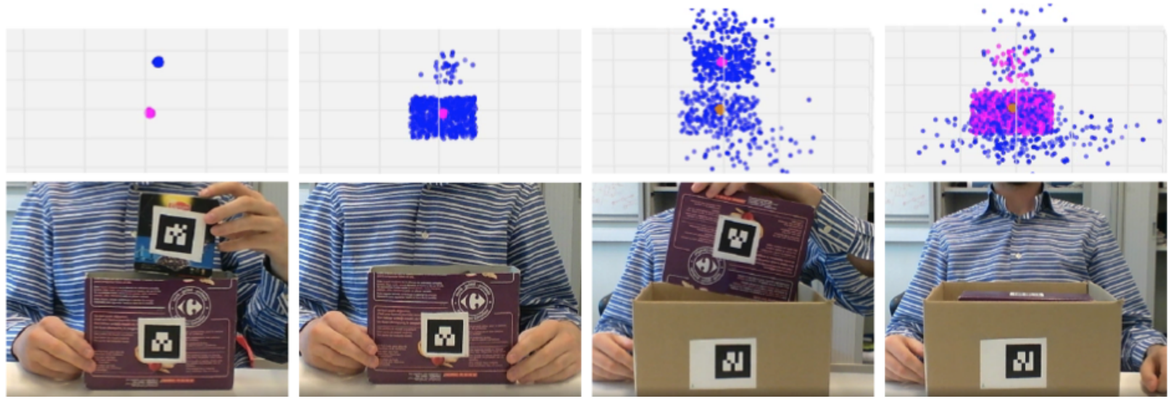
Finding unseen objects



Rules

$$\texttt{inside}_{t+1}(\texttt{ID}, \texttt{B}) \sim \texttt{finite}([0.8{:}\texttt{true}, 0.2{:}\texttt{false}]) \leftarrow$$
$$\texttt{not}(\simeq(\texttt{inside}_t(\texttt{ID}, \texttt{C})) = \texttt{true}),\ \texttt{on}_t(\texttt{ID}, \texttt{B}),$$
$$\texttt{type}(\texttt{B}, \texttt{box}),\ \simeq(\texttt{yaw}_t(\texttt{B})) > 0,\ \texttt{smaller}(\texttt{ID}, \texttt{B}).$$

$$\texttt{inside}(\texttt{A}, \texttt{B}) \leftarrow \texttt{inside}(\texttt{A}, \texttt{C}), \texttt{inside}(\texttt{C}, \texttt{B})$$



Nitti, Belle & De Raedt (2015)

Audience questions:

1) Have you worked on robotics?
2) Seen the need for unknowns?

Taking stock

Deduction vs induction

- Perhaps the most fundamental issue in areas such as philosophy, cognition and artificial intelligence
- Deduction camp concerns itself with questions about the expressiveness of formal languages for capturing knowledge about the world, together with proof systems for reasoning from such knowledge bases — **i.e., logics**
- Learning camp attempts to generalize from examples about partial descriptions about the world — **i.e., properties of random variables**

History

- These camps have loosely divided the development of the field, but advances in cross-over areas such as **statistical relational learning**, **neuro-symbolic systems**, and **high-level control** have illustrated that the *dichotomy is not very constructive*
- Many interesting historical ideas from Aristotle, Hume, Carnap, Pierce, and more recently Plotkin, Muggleton, and others.
- Deeper connection also between logic & probability: **probabilistic logics**, 0-1 laws etc.

- Despite the success of methods such as deep learning, need to address *model re-use, transferability, causal understanding, relational abstraction, explainability, data efficiency*
    - What might common sense knowledge look like? Widely acknowledged to involve concepts such as *time, space, abstraction*
    - ML methods need to be (and are being) further augmented with logical, symbolic and/or programmatic artifacts
- Low-level, data-intensive, reactive computations needs to be tightly integrated with high-level, deliberative computations (parallel often drawn to Kahneman's *System 1* vs *System 2* processing in human cognition)

Opportunities

- Allow human input via **hard & soft constraints**
- Enable context-dependent/user-specific **interpretability**
- Reason about **constraints, verify properties**

Need hybrid systems **integrating reasoning & learning**

- *Starting point:* general-purpose computational schemes
  - One such scheme: **propositional probabilistic reasoning**

Key questions

- What knowledge is provided by the modeler versus what can be acquired by observations?
- What is the language for representing and reasoning with the background knowledge and observations?
- What kind of semantics governs the updating of a priori knowledge given new and possibly conflicting observations (e.g., unknown ground truth)?
- How does the system generalize from low-level observations to high-level structured knowledge?

This tutorial

Not to resolve this debate, but rather provide further evidence for the connections

- **Logic vs. Machine Learning**, including the study of problems that can be solved using either logic-based techniques or via machine learning, …
- **Machine Learning for Logic**, including the learning of logical artifacts, such as formulas, logic programs, . . .
- **Logic for Machine Learning**, including the role of logics in delineating the boundary between tractable and intractable learning problems, …, and the use of logic as a declarative framework for expressing machine learning constructs

- Common misconception that logic is for discrete properties, whereas probability theory and machine learning, more generally, is for continuous properties
  - Logical formulas are discrete structures but can express continuous or uncountably infinite properties

- Modelling fairness and responsibility
- Reasoning about causality and explanations
- Neuro-symbolic landscape

We survey and discuss previews of recent results, details omitted

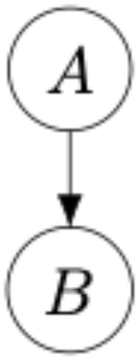
Audience questions:

1) induction vs deduction - how do you understand it?
2) what's your view on background knowledge vs observational data?
3) Which of the three categories (L for M, M for L, L v L) does your work fit in?

Logic vs. Machine Learning

Reveals how logic-based solvers can tackle classical inference and learning problems

Bayesian networks



a	$\Pr(A = a)$
1	0.5
0	0.5

a	b	$\Pr(B = b \mid A = a)$
1	1	0.6
1	0	0.4
0	1	0.1
0	0	0.9

$$\nu_A = 0.5,$$

and

$$\nu_B = 0.6[\lambda_{B=1}] \cdot [\lambda_{A=1}] + 0.4[\neg\lambda_{B=1}] \cdot [\lambda_{A=1}] + 0.1[\lambda_{B=1}] \cdot [\neg\lambda_{A=1}] + 0.9[\neg\lambda_{B=1}] \cdot [\neg\lambda_{A=1}].$$

Like enumerating all possible propositional interpretations, and according them weights

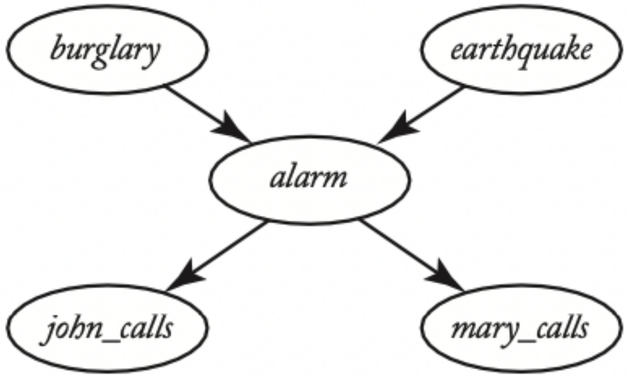
Inference & learning

- Recall given data *D* and Bayesian network *N,*

$$score(N, D) \propto \log \Pr(D \mid N) - size(N)$$

- Pick network only if $score(N', D) > score(N, D)$
- But inference becomes especially challenging with deterministic constraints, logical & relational syntax
  - E.g., imagine constraint that $A \neq B$ and all numbers are unnormalised probabilities

But also extends to other probabilistic logical languages



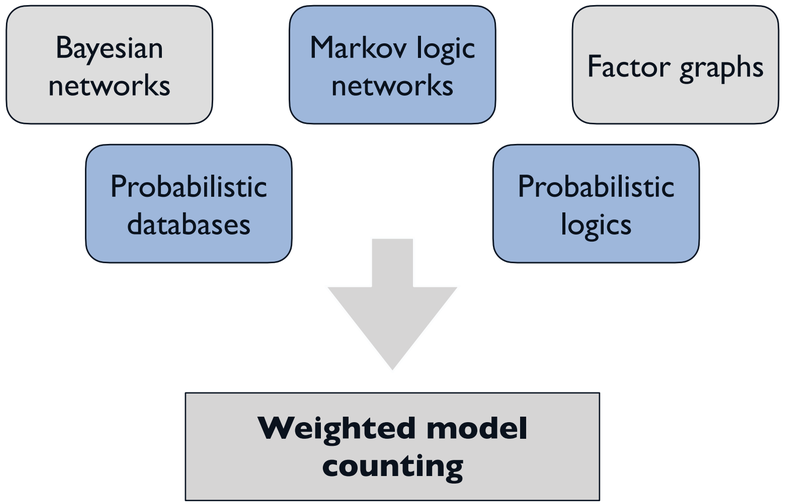
$$0.01 :: burglary.$$
$$0.05 :: earthquake.$$
$$0.7 :: hearsalarm(X).$$

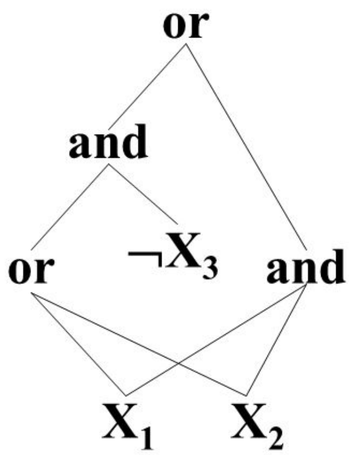
$$alarm \leftarrow burglary.$$
$$alarm \leftarrow earthquake.$$
$$calls(X) \leftarrow alarm, hearsalarm(X) .$$

De Raedt et al, Morgan Claypool, 2016.

Weighted model counting

- Extends #SAT with weights on models, factorized over literals
- Handle hard and soft constraints natively, general-purpose, state-of-the-art on many benchmark problems (e.g., knowledge compilation, sampling)





Chavira and Darwiche (2008)

Formulation

- Given CNF $(x \vee y) \wedge (y \vee \neg z)$,
  - SAT: find a model $(x = 0, y = 1, z = 0)$
  - #SAT returns (0,1,0), (0,1,1), (1,1,0), (1,1,1), (1,0,0)
  - Equivalently, satisfying probability of 5/8

WMC

- $\mathscr{L}$ consisting of finitely many predicate symbols *P* and constants *C*
- Given $\Delta \in \mathscr{L}$ and $w\colon Lits(\Delta) \to \mathbb{R}^{\geq 0}$

$$\mathrm{WMC}(\Delta, w) = \sum_{M \models \Delta} \prod_{l \in M} w(l)$$

$$\Pr(q \mid e, \Delta, w) = \frac{\mathrm{WMC}(q \wedge e \wedge \Delta, w)}{\mathrm{WMC}(e \wedge \Delta, w)}$$

Sample Markov network

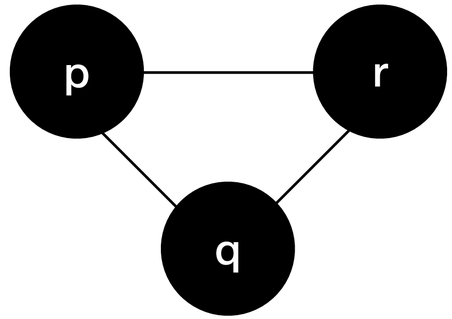
- Equivalently
  $\Delta = \left(\neg p \vee \neg q\right) \wedge \left(f_1 \Leftrightarrow p \vee r\right) \wedge \left(f_2 \Leftrightarrow \neg q \vee r\right)$
- $w(p) = 0.1, w(f_1) = .8,$
  $w(f_2) = .6$ and otherwise,
  weight is 1
- $(p, \neg q, r, f_1, f_2)$ then has weight
  $.1 \times .8 \times .6$

$\neg p \vee \neg q$

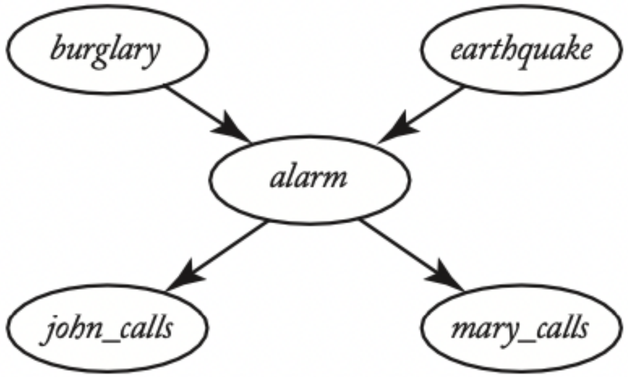
$.1 \quad p$

$.8 \quad p \vee r$

$.6 \quad q \Rightarrow r$



A less abstract example



$$0.01 :: burglary.$$
$$0.05 :: earthquake.$$
$$0.7 :: hearsalarm(X).$$

$$alarm \leftarrow burglary.$$
$$alarm \leftarrow earthquake.$$
$$calls(X) \leftarrow alarm, hearsalarm(X) .$$

De Raedt et al, Morgan Claypool, 2016.

The WMC formulation

$$
\begin{aligned}
&alarm \leftrightarrow burglary \vee earthquake.\\
&calls(john) \leftrightarrow alarm \wedge hearsalarm(john).\\
&w(burglary) = 0.01, w(\neg burglary) = 0.99,\\
&w(earthquake) = 0.05, w(\neg earthquake) = 0.95,\\
&w(hearsalarm(john)) = 0.7, w(\neg hearsalarm(john)) = 0.3\ .
\end{aligned}
$$

We could compute model count by exhaustive search

OR

By cleverly factoring formula into a data structure (aka **knowledge compilation**)

Audience questions:

1) does this correspond to how you think about probabilistic reasoning?
2) what are your favourite exact and approximate schemes?

Tractable probabilistic models: an alternative way to encode a joint distribution

Representing a distribution as a polynomial

- X follows a Bernoulli distribution with probability of success equal to p, then the polynomial $F(X, \bar{X}) = pX + (1 - p)\bar{X}$
  - $Pr(X = 1) = F(1,0) = p$
  - $Pr(X = 0) = F(0,1) = 1 - p$

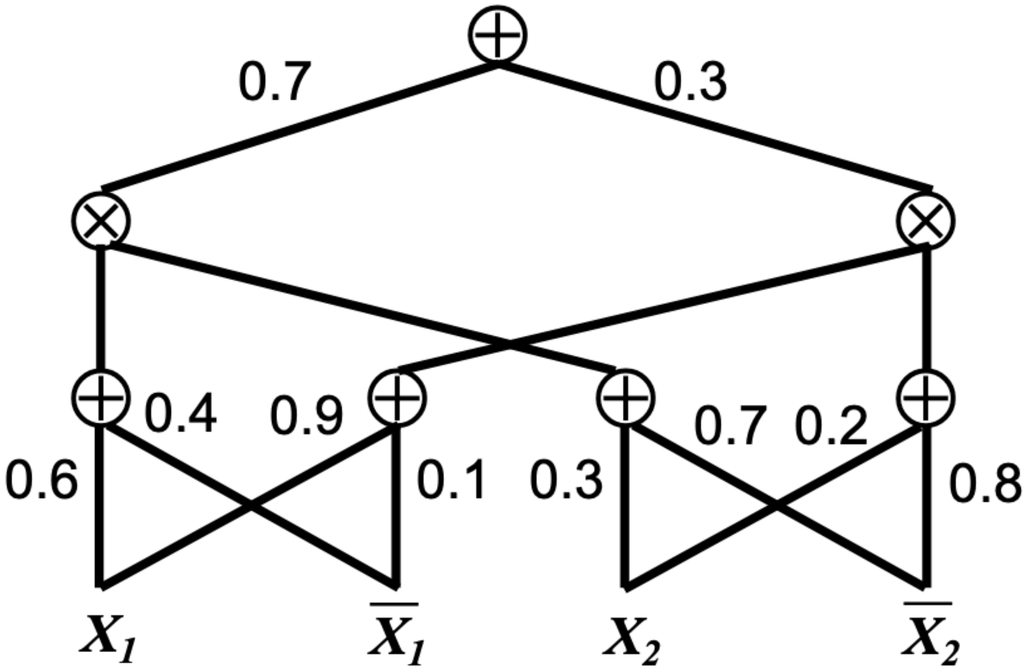
Joint distribution of two binary variables

$$F(X_1, \bar{X}_1, X_2, \bar{X}_2) = Pr(X_1, X_2)X_1X_2 + Pr(\bar{X}_1, X_2)\bar{X}_1X_2 + Pr(X_1, \bar{X}_2)X1\bar{X}_2 + Pr(\bar{X}_1, \bar{X}_2)\bar{X}_1\bar{X}_2$$

- Marginalisation of the first corresponds to setting $X_1, \bar{X}_1$ to 1:
  - $F(1,1,X_2, \bar{X}_2) = Pr(X_2)X_2 + Pr(\bar{X}_2)\bar{X}_2$
- Conditioning requires two computations:
  - $Pr(X_2 \,|\, X_1 = 1) = F(1,0,X_2, \bar{X}_2)/F(1,0,1,1)$

Sum product networks

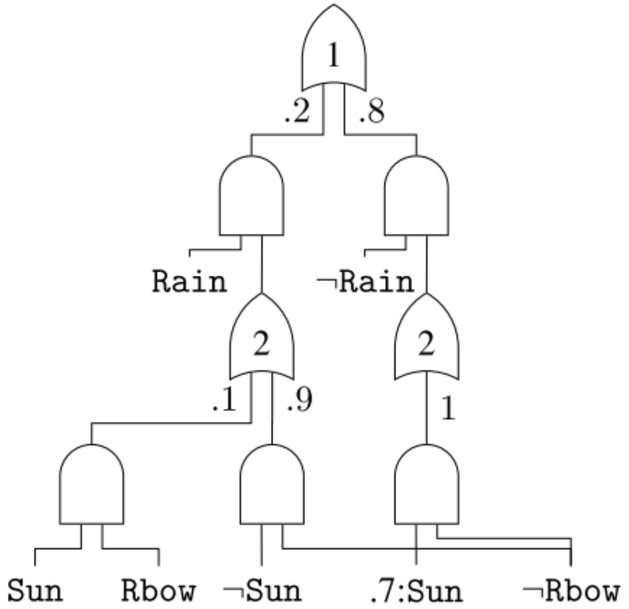
- Rooted directed graphs that explicitly represent the computations in a network polynomial
- Leaf nodes: tractable univariate distributions
- Weighted sums of products (i.e., weighted mixtures)
- Certain distributions that can be encoded as compact polynomial sized SPNs, that cannot be compactly represented as BNs



standard deep models $P(y \mid x)$ **vs SPNs:** $P(x, y)$

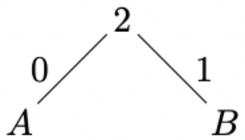
Probabilistic sentential decision diagrams (PSDDs)

- Parameterized sentential decision diagrams
- Represent a propositional formula as $(p_1 \wedge s_1) \vee \ldots \vee (p_k \wedge s_k)$, primes and subs,
  - the primes form a partition,
  - the variables appearing in all primes are the same, the variables in the subs are the same, and they do not overlap
- Many useful quantities can be computed efficiently

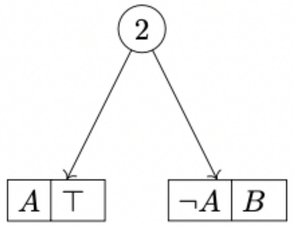


Simple example

Formula: $A \vee B \equiv (A \wedge \top) \vee (\neg A \wedge B)$

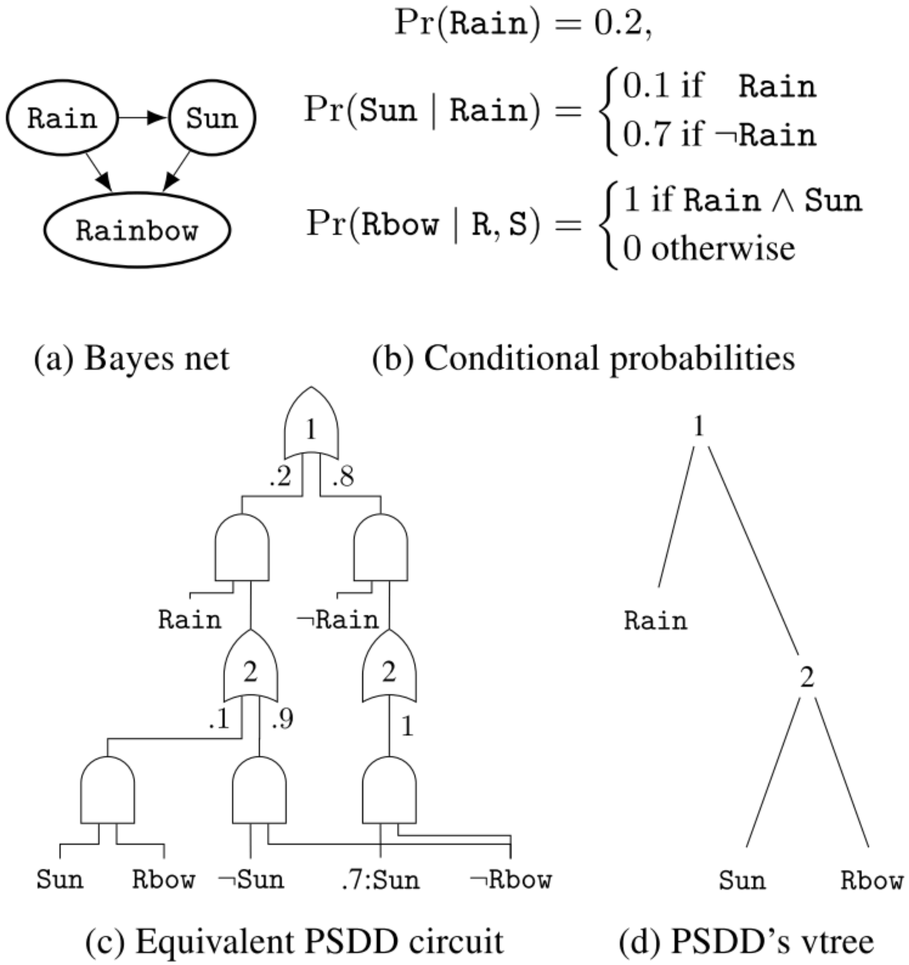


(a) A vtree



(b) The resulting SDD

Bayesian network



Kisa et al. (2014)

We could see TPM purely as an intermediate computational target for high-level languages

OR

Study and use TPMs on their own for tractable structure learning and reasoning

Audience question: how does this relate to your intuition about, e.g. deep learning?

Causal properties

Quick recap

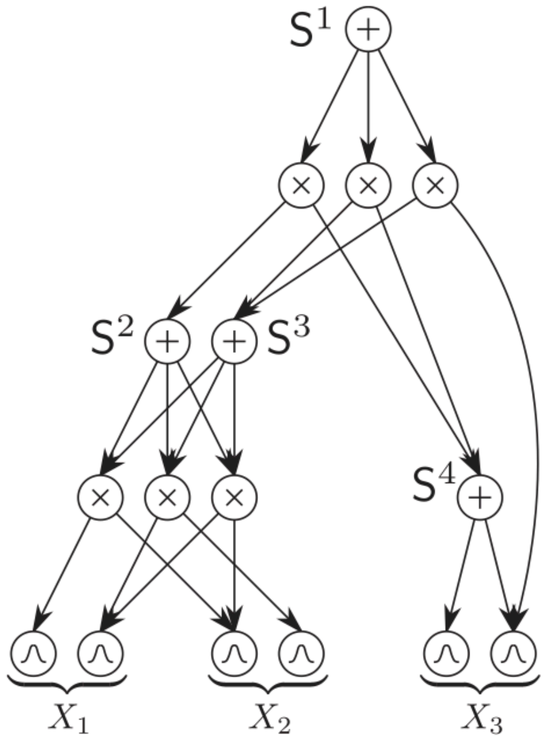
- In standard probabilistic models, one is simply interested in conditioning on observations $\Pr(y \mid x)$: e.g., what is the likelihood of being tall given that you eat cheese?
- Causal reasoning allows **interventions** $\Pr(y \mid do(x))$: e.g., what is the probability of a person being tall given that he/she is made to eat cheese?
- **Counterfactual queries** allows alternate worlds $\Pr(y \mid x^*)$: e.g., how tall would a person be if he/she was eating carrots instead of cheese?

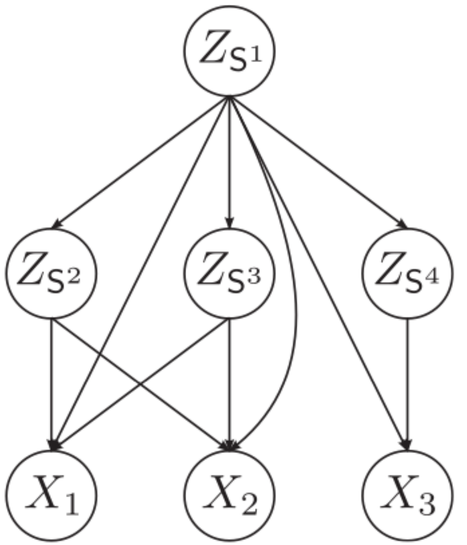
BNs and SPNs

- Any BN can be compiled into a SPN or a PSDD
  - However, TPM internal representation makes it very challenging to identify relationships and dependencies among the variables (in contrast to BNs, where it is immediate to uncover all the conditional independencies)
  - In other words, *BNs are transparent models, while SPNs and PSDDs essentially black-boxes*
- Many attempts to transform TPMs back to BNs in order to make the underlying relationships clear, hoping that it would also enable TPMs to be used in causal inference applications, for example

.

The structure of the resulting BNs can be used to study probabilistic dependencies and causal relationships between the variables of the original SPNs.
Zhao et al, ICML 2015.


S^1
S^2
S^3
S^4
X_1
X_2
X_3


Z_{S^1}
Z_{S^2}
Z_{S^3}
Z_{S^4}
X_1
X_2
X_3

What we show

- Existing SPN to BN decompilations (i.e., transformations) result in graphs that support only trivial causal queries

- Identify a set of sufficient conditions that lead to an exact SPN to BN de-compilation (thus enhancing the transparency of SPNs by accurately uncovering their internal representations) — *omitted*

- Establish a relationship between PSDDs and chain graphs (CGs) that both enhances the PSDDs' transparency, as wells as allow for equipping them with a form of causal semantics — *omitted*

Recap: computing interventions

What follows is an essential graphical tool for deciding under what conditions we can reduce interventional queries to conditional ones. Here, $G_{\overline{x}}$ denotes the graph obtained after deleting all the edges pointing to X, $G_{\underline{x}}$ the one resulting after deleting all the edges emerging from X, and $G_{\overline{x}\underline{z}}$ refers to the graph where both edges incoming to X and stemming from Z are deleted.

Definition 3.3.1 ***(Rules of do-Calculus)*** *Let G be a DAG corresponding to a SEM and $\Pr(\cdot)$ the probability measure induced by it. If* ***X, Y, Z, W*** *are disjoint sets, then the following hold:*

Rule 1: $\Pr(\boldsymbol{y}|do(\boldsymbol{x}), \boldsymbol{z}, \boldsymbol{w}) = \Pr(\boldsymbol{y}|do(\boldsymbol{x}), \boldsymbol{w})$ *if* $(\boldsymbol{Y} \perp \boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{W})_{G_{\overline{X}}}$.

Rule 2: $\Pr(\boldsymbol{y}|do(\boldsymbol{x}), do(\boldsymbol{z}), \boldsymbol{w}) = \Pr(\boldsymbol{y}|do(\boldsymbol{x}), \boldsymbol{z}, \boldsymbol{w})$ *if* $(\boldsymbol{Y} \perp \boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{W})_{G_{\overline{X}\underline{Z}}}$.

Rule 3: $\Pr(\boldsymbol{y}|do(\boldsymbol{x}), do(\boldsymbol{z}), \boldsymbol{w}) = \Pr(\boldsymbol{y}|do(\boldsymbol{x}), \boldsymbol{w})$ *if* $(\boldsymbol{Y} \perp \boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{W})_{G_{\overline{X}\overline{Z(W)}}}$, *where* ***Z(W)*** *is the set of* ***Z****-nodes that are not ancestors of any* ***W****-node in* $G_{\overline{X}}$.

The case of SPNs

Note: when no edge coming out of the observable variables implies that all interventional distributions are trivial

Proposition *Let G be a DAG, $\boldsymbol{V}$ the set of its nodes, and $\boldsymbol{X} \subseteq \boldsymbol{V}$ such that no node in $\boldsymbol{X}$ has an edge coming out of it, then* $\Pr(\boldsymbol{V}_{-\boldsymbol{X}}|do(\boldsymbol{X})) = \Pr(\boldsymbol{V}_{-\boldsymbol{X}})$.

Theorem *The BN, G, that results after transforming an SPN using the procedure described in [197], [144], or [28], satisfies the property* $\Pr(\boldsymbol{V}_{-\boldsymbol{X}}|do(\boldsymbol{X})) = \Pr(\boldsymbol{V}_{-\boldsymbol{X}})$, *for any $\boldsymbol{X} \subseteq \boldsymbol{V}$, where $\boldsymbol{V}$ is the set of the observable variables.*

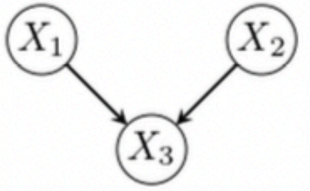
- This limitation stems from the fact that sum nodes are interpreted as latent variables, which leads to every probabilistic dependency been attributed to unobserved confounders, not to direct interaction between the variables.
- In turn, it is not surprising that all interventions are trivial, since the resulting BNs hold no meaningful information about the relationships between the variables.
- Furthermore, it is concerning that even when SPNs are compiled from BNs that are both transparent and suitable for causal applications, the de-compiled BNs do not retain these properties.

- We show PSDDs at least allow non-trivial interventional and counterfactual queries to be posed
- Research towards learning SPNs from data in a way that guarantees they satisfy conditions that permit an exact de-compilation is a very promising direction, since it would allow for very expressive transparent models to be learnt directly from data
- E.g., train SPNs directly on the interventional distribution (Zecevic et al., 2021)

Audience questions:

1) How many have worked with causal graphs?
2) How many on causal reasoning in non-standard models (e.g., deep learning)?

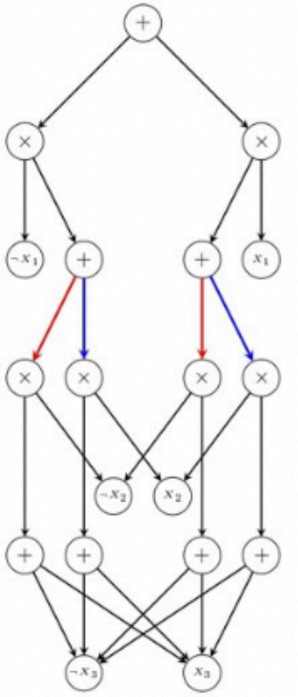
Enforcing constraints on trained SPNs



$X_1 \perp\!\!\!\perp X_2$

$X_1 \not\perp\!\!\!\perp X_2 | X_3$

(a) BN



(b) SPN

Example of BN inducing an independence constraint, and the corresponding SPN

Our results

- It is possible to enforce (unconditional) independence constraints, conditional independence constraints, as well as interventional ones to SPNs
- Correspond to the SPNs' parameters satisfying a multivariate system of equations
- Propose optimization schemes for training SPNs, in the presence of both hard and soft constraints
- Works directly on the SPN without having original BN

Main idea

- SPNs exhibit a correspondence between parameters and probabilities (obviously, right?)

- Note: “A is independent of B” equivalent to $\Pr(A, B) = \Pr(A) \Pr(B)$
  - Rewrite as system of equations on parameters

Conditional constraints

Theorem *Let S be an SPN representing the joint distribution of variables $X_1, \cdots, X_n$. Let X_i, X_j, X_k be binary variables, then a conditional independence constraint of the form $X_i \perp X_j | X_k$ is equivalent to a quadratic multivariate system of equations on the SPN's parameters.*

$$\Pr(X_i, X_j | X_k) = \Pr(X_i | X_k) \cdot \Pr(X_j | X_k) \implies \frac{\Pr(X_i, X_j, X_k)}{\Pr(X_k)} = \frac{\Pr(X_i, X_k)}{\Pr(X_k)} \cdot \frac{\Pr(X_j, X_k)}{\Pr(X_k)}$$

Express these in the SPN

$$S(\mathbf{x}) = \sum_{\mathbf{x}} f(\mathbf{x}) \prod_{n=1}^{N} 1_{x_n}$$

$$\Pr(X_i, X_j, X_k) = \sum_{\mathbf{x}: x_i, x_j, x_k} f(\mathbf{x}) 1_{x_i} 1_{x_j} 1_{x_k} + \sum_{\mathbf{x}: \neg x_i, x_j, x_k} f(\mathbf{x}) 1_{\neg x_i} 1_{x_j} 1_{x_k} + \sum_{\mathbf{x}: x_i, \neg x_j, x_k} f(\mathbf{x}) 1_{x_i} 1_{\neg x_j} 1_{x_k}$$
$$+ \sum_{\mathbf{x}: x_i, x_j, \neg x_k} f(\mathbf{x}) 1_{x_i} 1_{x_j} 1_{\neg x_k} + \sum_{\mathbf{x}: \neg x_i, \neg x_j, x_k} f(\mathbf{x}) 1_{\neg x_i} 1_{\neg x_j} 1_{x_k} + \sum_{\mathbf{x}: \neg x_i, x_j, \neg x_k} f(\mathbf{x}) 1_{\neg x_i} 1_{x_j} 1_{\neg x_k}$$
$$+ \sum_{\mathbf{x}: x_i, \neg x_j, \neg x_k} f(\mathbf{x}) 1_{x_i} 1_{\neg x_j} 1_{\neg x_k} + \sum_{\mathbf{x}: \neg x_i, \neg x_j, \neg x_k} f(\mathbf{x}) 1_{\neg x_i} 1_{\neg x_j} 1_{\neg x_k}$$

$$\Pr(X_i, X_k) = \sum_{\mathbf{x}: x_i, x_k} f(\mathbf{x}) 1_{x_i} 1_{x_k} + \sum_{\mathbf{x}: \neg x_i, x_k} f(\mathbf{x}) 1_{\neg x_i} 1_{x_k} + \sum_{\mathbf{x}: x_i, \neg x_k} f(\mathbf{x}) 1_{x_i} 1_{\neg x_k} + \sum_{\mathbf{x}: \neg x_i, \neg x_k} f(\mathbf{x}) 1_{\neg x_i} 1_{\neg x_k}$$

On simplification, leads to a set of constraints

$$\sum_{\mathbf{x}:x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_i,x_j,x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:x_i,x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_j,x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_i,x_j,x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:\neg x_i,x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_j,x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_i,\neg x_j,x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:x_i,x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_j,x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_i,\neg x_j,x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:\neg x_i,x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_j,x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_i,x_j,\neg x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:x_i,\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_j,\neg x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_i,x_j,\neg x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:\neg x_i,\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_j,\neg x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_i,\neg x_j,\neg x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:x_i,\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_j,\neg x_k} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_i,\neg x_j,\neg x_k} f(\mathbf{x}) = \sum_{\mathbf{x}:\neg x_i,\neg x_k} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_j,\neg x_k} f(\mathbf{x})$$

Likewise for interventions and independence

Theorem *Let S be an SPN representing the joint distribution of variables $X_1, \cdots, X_n$. Let X_i be a binary variable, then the constraint $\Pr(\boldsymbol{X}_{-i}|do(X_i = 0)) = \Pr(\boldsymbol{X}_{-i}|do(X_i = 1))$ is equivalent to a multivariate linear system of equations on the SPN's parameters.*

Theorem *Let S be an SPN representing the joint distribution of variables $X_1, \cdots, X_n$. Let X_i, X_j be two variables, then an (unconditional) independence constraint of the form $X_i \perp X_j$ is equivalent to a quadratic multivariate system of equations on the SPN's parameters.*

$$\sum_{\mathbf{x}:x_i,x_j} f(\mathbf{x}) = \sum_{\mathbf{x}:x_i} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_j} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:\neg x_i,x_j} f(\mathbf{x}) = \sum_{\mathbf{x}:\neg x_i} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:x_j} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:x_i,\neg x_j} f(\mathbf{x}) = \sum_{\mathbf{x}:x_i} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_j} f(\mathbf{x})$$

$$\sum_{\mathbf{x}:\neg x_i,\neg x_j} f(\mathbf{x}) = \sum_{\mathbf{x}:\neg x_i} f(\mathbf{x}) \cdot \sum_{\mathbf{x}:\neg x_j} f(\mathbf{x})$$

A loss term approach

Algorithm Training with soft constraints

Require: SPN structure S, dataset D, constraints C_n, hyperparameters α_n, learning rate γ
Initialize $\mathbf{w}$
while convergence criterion is not satisfied **do**
    Sample a $d \in D$
    $\mathbf{w}^{(k)} \leftarrow \mathbf{w}^{(k-1)} + \gamma(\nabla_{\mathbf{w}} \mathbf{S}(d) + \sum_n \alpha_n \nabla_{\mathbf{w}} C_n)$
    NormalizeWeights(S)
end while
return S

Algorithm 4 Training with hard constraints

Require: SPN structure S, dataset D, constraints C_n, learning rate γ
Initialize $\mathbf{w}$
while convergence criterion is not satisfied **do**
    Sample a $d \in D$
    $\mathbf{w}^{(k)} \leftarrow \mathbf{P}_{C_1,\ldots,C_n}(\mathbf{w}^{(k-1)} + \gamma \nabla_{\mathbf{w}} \mathrm{S}(d))$
    NormalizeWeights(S)
end while
return S

Audience question: what kind of constraints make sense for fairness?

Diverse counterfactuals

Our results

- We extend the methodology in [Russell 2019] to multilinear models, which include DTs, RFs, and BNCs, discriminative SPNs.

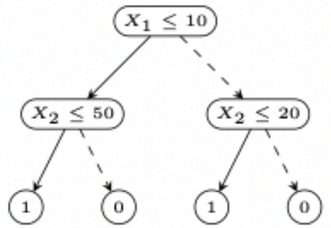
- We express the counterfactual generation problem as an integer linear program (ILP).

- For DTs and RFs, the result can be an infinite set of counterfactuals, instead of a single instance.

  - Diversity constraints can be easily incorporated, just as in [Russell 2019].

0 and 1-decisions

Let $G : \mathbf{X} \to \{0, 1\}$ be a binary classification function, and $P_0^G(\mathbf{X}), P_1^G(\mathbf{X})$ be multilinear polynomials of indicator variables, where all coefficient are equal to 1 and there is no constant term. Then $P_0^G(\mathbf{X})$ (respectively, $P_1^G(\mathbf{X})$) is called the 0-decision (resp. 1-decision) polynomial of G, iff
$G(\mathbf{X}) = 0 \Leftrightarrow P_0^G(\mathbf{X}) = 1$ (resp. $G(\mathbf{X}) = 1 \Leftrightarrow P_1^G(\mathbf{X}) = 1$)



The above DT is defined over variables, X_1, X_2, or over the rules, $X_1 \leq 10, X_2 \leq 50, X_2 \leq 20$. For example, the 0-decision polynomial is:

$$P_0^G(X_1, X_2) = \mathbb{1}[X_1 \leq 10] \cdot (1 - \mathbb{1}[X_2 \leq 50]) + (1 - \mathbb{1}[X_1 \leq 10]) \cdot (1 - \mathbb{1}[X_2 \leq 20]$$

We define the distance between two points as follows:

$$\|d - d'\|_{1,w} = w_1 |\mathbb{1}[X_1 \leq 10] - \mathbb{1}[X_1' \leq 10]| + w_2 |\mathbb{1}[X_2 \leq 20] - \mathbb{1}[X_2' \leq 20]| + w_3 |\mathbb{1}[X_2 \leq 50] - \mathbb{1}[X_2' \leq 50]|,$$

- Suppose $P_0(X_1, X_2, X_3) = X_1X_2X_3 + X_1X_2(1 - X_3)$ in a SPN.

- As long as $X_1 = X_2 = 1$, a datapoint is classified in the 0 class, regardless of what value X_3 has. In turn, we can reduce the 0-DP to $P_0(X_1, X_2, X_3) = X_1X_2$.

Generating counterfactuals

- Suppose a datapoint is classified as 0. A natural way to address this situation would be to find a point s.t. decision polynomial is equal to 1, while minimising distance.

Proposition *Let* $X_1, X_2, \cdots X_k \in \{0,1\}$*, then* $X_1 \cdot X_2 \cdots X_{k-1} \cdot X_k = 1 \Leftrightarrow X_1 + X_2 + \cdots + X_k = k$ *and* $X_1 \cdot X_2 \cdots X_{k-1} \cdot X_k = 0 \Leftrightarrow X_1 + X_2 + \cdots + X_k \leq k-1$.

Proposition *Let* $P_0(\boldsymbol{X}) = X_{11} \cdot X_{12} \cdots X_{1k} + X_{21} \cdot X_{22} \cdots X_{2m} + \cdots + X_{n1} \cdot X_{n2} \cdots X_{nl}$*, where each* $X_i \in \{0,1\}$*, be the* 0*-decision polynomial of a model. Furthermore, let the constraints* $X_{11} + X_{12} \cdots + X_{1k} \geq k \cdot \delta_1, X_{21} + X_{22} \cdots + X_{2m} \geq m \cdot \delta_2, \ldots, X_{n1} + X_{n2} \cdots + X_{nl} \geq l \cdot \delta_n, \sum_{i=1}^{n} \delta_i = 1$*, where* $\delta_i \in \{0,1\}$*. If an assignment,* $\boldsymbol{X'}$*, satisfies these constraints, then* $P_0(\boldsymbol{X'}) = 1$*. An analogous statement holds for* $P_1(\boldsymbol{X})$.

Constraint for the DT example

$$
\begin{aligned}
&min\ \ w_1(1-1[X_1' \leq 10]) + w_2 1[X_2' \leq 20] + w_3(1-1[X_3' \leq 50])\ s.t.\\
&1[X_1' \leq 10] + (1-1[X_2' \leq 50]) \geq 2 \cdot \delta_1,\\
&(1-1[X_1' \leq 10]) + (1-1[X_2' \leq 20]) \geq 2 \cdot \delta_2,\\
&\delta_1 + \delta_2 = 1
\end{aligned}
$$

- Provide simplifications for solving optimisation problem
- Allows additional constraints such as $sex = male,$ and $X_1' > 3$ to be added to the objective function

	Sex	Age	Race	Juvenile felonies	Prior crimes	Two year residivism	Outcome
Factual	Male	33	Caucasian	0	2	Yes	Low score
Counterfactual	Male	33	Caucasian	$> \mathbf{0}$	2	Yes	High score
Diverse counterfactual	**Female**	33	Caucasian	(=0) 0	2	Yes	High score
Factual	Male	21	Black	0	0	Yes	High score
Counterfactual	Male	$\leq \mathbf{19.5}$	Black	0	0	Yes	Low score
Diverse counterfactual	Male	$(\geq 20) > \mathbf{21.5}$	Black	0	0	Yes	Low score
Factual	Male	27	Black	0	0	No	Low score
Counterfactual	Male	27	Black	0	$> \mathbf{4.5}$	No	High score
Diverse counterfactual	Male	$< \mathbf{21.5}$	Black	0	(=0) 0	No	High score
Factual	Male	32	Black	0	0	No	Low score
Counterfactual	Male	27	Black	0	$> \mathbf{1.5}$	No	High score
Diverse counterfactual	Male	32	**Caucasian**	$\geq \mathbf{1}$	(≤ 1) 0	No	High score
Factual	Male	43	Caucasian	0	2	No	Low score
Counterfactual	**Female**	43	Caucasian	0	2	No	High score
Diverse counterfactual	(Male)	43	Caucasian	0	$> \mathbf{4}$	No	High score

COMPAS dataset

	Sex	LSAT	Race	UGPA	Outcome
Factual	Male	34	White	3	Pass
Counterfactual	Male	$< \mathbf{19.25}$	White	3	Fail
Diverse counterfactual	Male	(> 25) 34	**Black**	$< \mathbf{1.95}$	Fail
Factual	Male	36.5	White	3.2	Pass
Counterfactual	Male	**20.75**	**Black**	$\leq \mathbf{1.95}$	Fail
Diverse counterfactual	Male	**19.25**	(White) White	**2.15**	Fail
Factual	Female	43	White	2.8	Pass
Counterfactual	Female	**26.75**	White	2.8	Fail
Diverse counterfactual	Female	$\leq \mathbf{19.25}$	(White) White	$\leq \mathbf{2.15}$	Fail
Factual	Male	35	White	2.7	Pass
Counterfactual	Male	35	**Black**	**1.95**	Fail
Diverse counterfactual	Male	$\leq \mathbf{19.25}$	(White) White	2.7	Fail
Factual	Male	33	White	3	Pass
Counterfactual	Male	33	**Black**	**1.85**	Fail
Diverse counterfactual	Male	$\leq \mathbf{19.25}$	(White) White	3	Fail

LSAT data instances

Audience question:

1) Suggest a reasonable constraint for credit decisioning
2) Motivate a case of your own

Some applications to ethics

Fairness

ML in social contexts

- ML techniques have become pervasive across a range of different applications having a “social” component
  - *Recidivism prediction* — historically disadvantaged groups in poorer neighborhoods with high false positives in conviction rate
  - *Consumer credit-risk analysis* — especially if determined by geographic location
  - *Job applications filtering* — especially if trained on historical data with gender and race bias

Fair algorithms

- Should be “fair” wrt **protected attributes** (ethnicity, sex, age, nationality and marital status) — such attributes should not affect prediction
- Potential to discriminate on the basis of **proxy variables** (e.g., name, primary school, location can reveal gender or race)
- Designing fair algorithms important in public, political & legal landscapes

Multiple (and often mutually exclusive) definitions

Examples

- Fairness through unawareness $f: X \rightarrow \hat{y}$ where $a_p \notin X$
- Prevents algorithm from learning direct bias on the basis of the protected attribute
- But doesn’t prevent indirect bias, which the algorithm can learn by exploiting the relationship between other training variables and the protected attribute

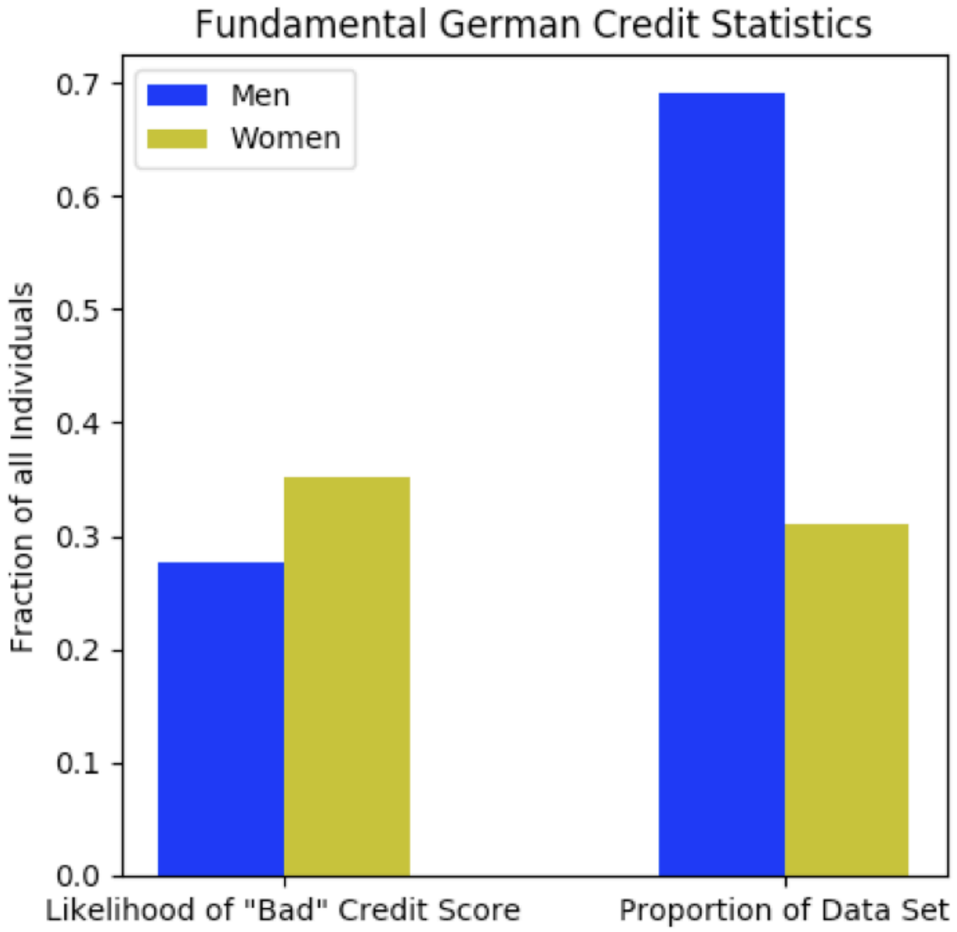
Examples contd.

- Demographic parity: $\Pr(y = 1 \mid a_p = 1) = \Pr(y = 1 \mid a_p = 0)$
- It is equally likely to make the prediction $y = 1$ regardless of the value of the protected attribute a_p
- But instance rate of $y = 1$ may differ starkly between different demographic groups
  - Women comprise less than 10% of the incarcerated population in the United Kingdom — prediction of future criminality would unfairly penalise women

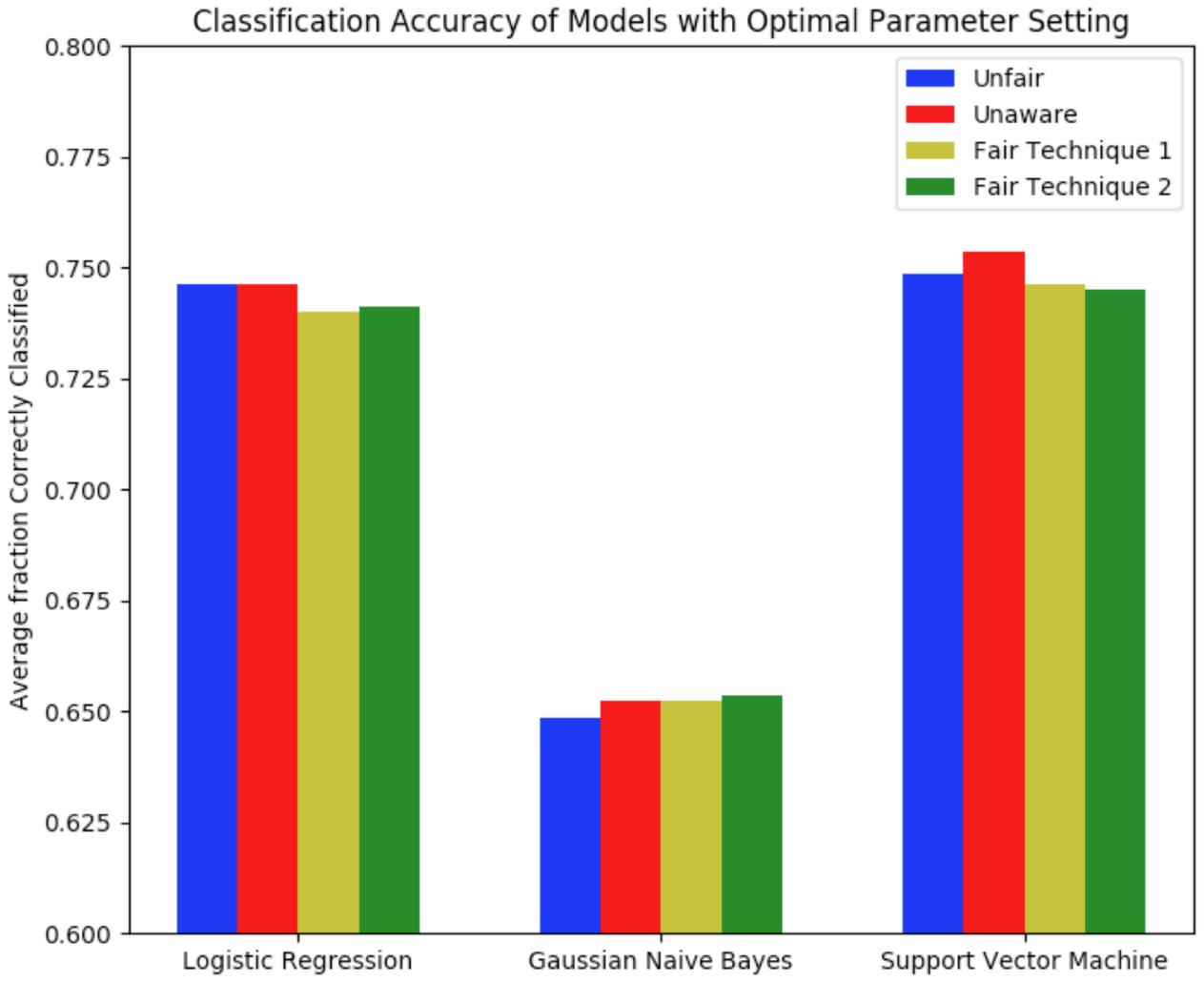
Our approach

- Contribution of the protected is “backed-out” $X' = X - \mathbb{E}(X \mid P)$
  - Technique 1: remove it before training
  - Technique 2: remove it after training (mitigate concerns about the potential for the model to reacquire explicit information about the protected attribute)

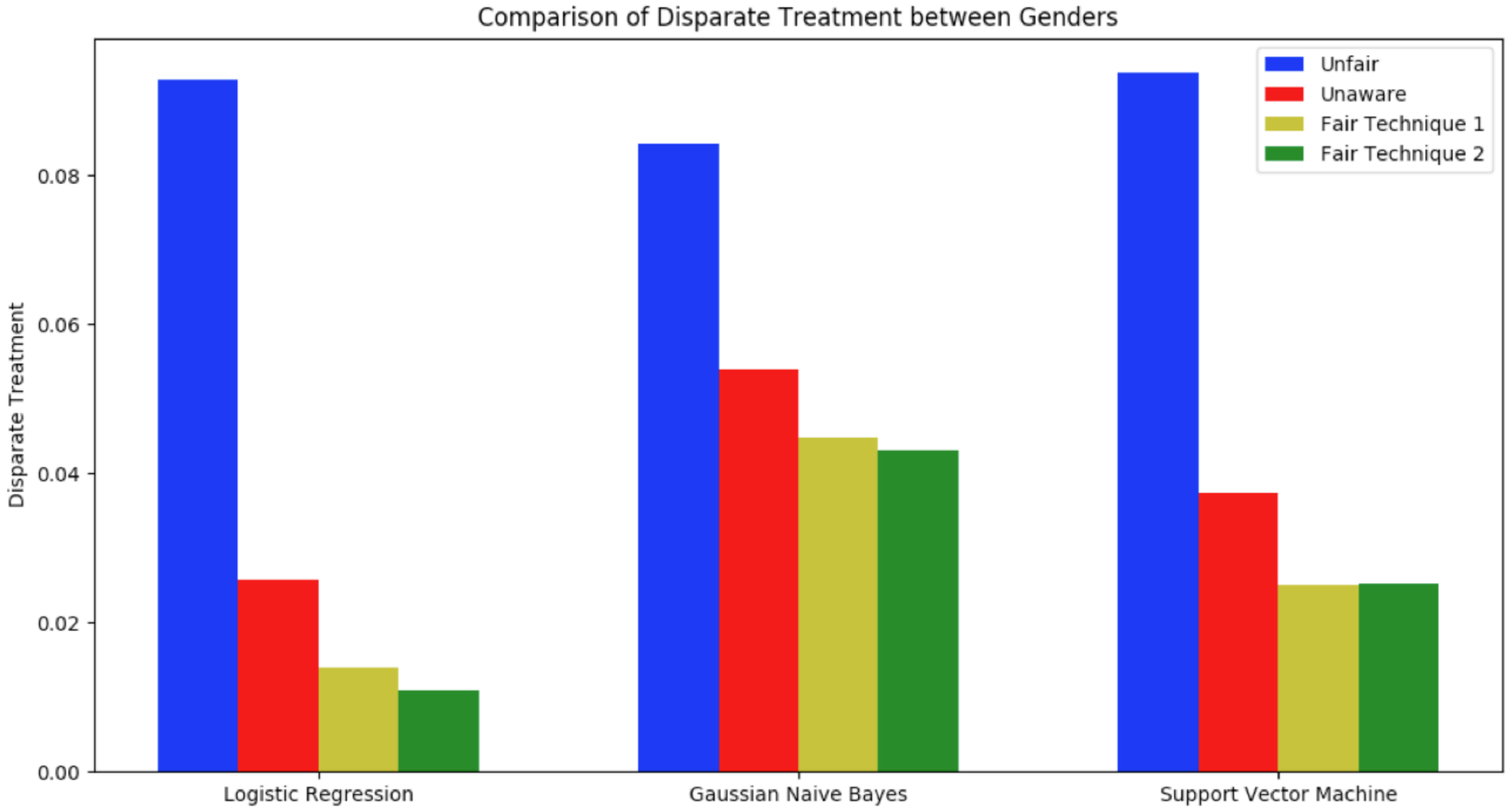
Unfair dataset: predict good/bad credit score



Classification accuracy

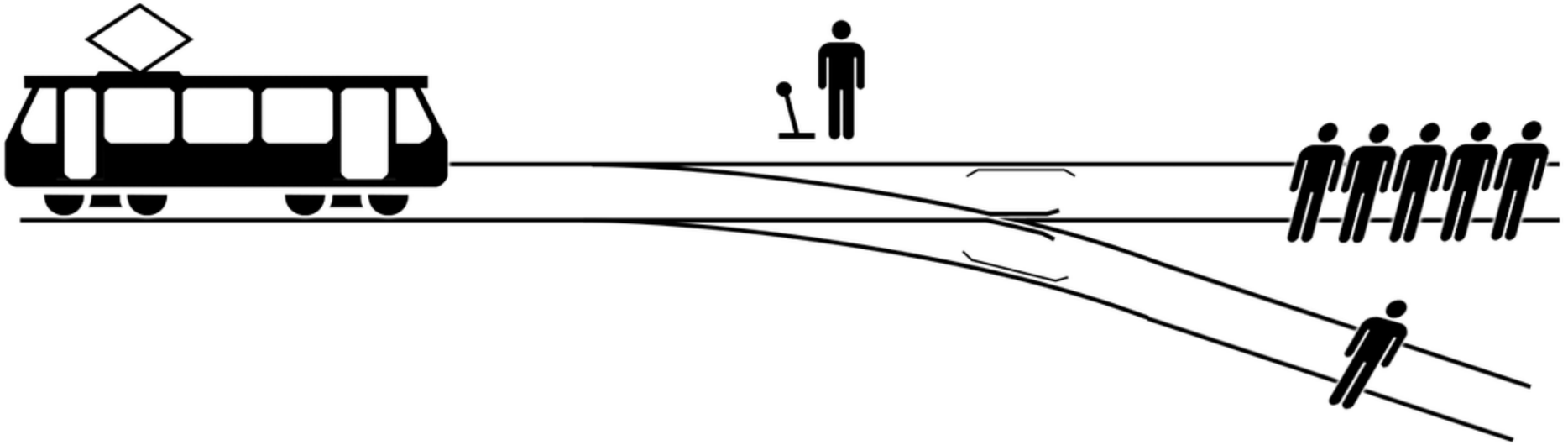


Disparate treatment



Audience question: What measures can we take to address disparity? Is ML helping or making things worse?

Can we endow machines with moral responsibility?



Too extreme?

- Possible regulation for driverless cars — e.g., a driverless car should always opt for property damage over personal injury
  - What if its a very expensive property, but injury chance is very small
- Kill passenger instead 5 pedestrians?
- Would you drive in such a car?
- Would you drive with your child?

We will want to defer to human expert, and may want to “hard-code” choices, but should the machine reason about consequences and blame?

What is blameworthy?

- Can’t be blamed if my action/inaction didn’t cause event
- Can’t be blamed if don’t know of event even if my actions could, in principle, stop it
- What if event was unintended consequence?
- Difference between pulling lever vs pushing muscular person onto track

Causal models

- **Model *M*:** ⟨**Exogenous Vars**, **Endogenous Vars**, **Structural Equations**⟩
- **Lever action (endogenous):** $A = 1$ if pulled, $A = 0$ otherwise
- **Outcome (endogenous):** $T1 = 1$ if five die, 0 otherwise; $T2 = 1$ if person on sidetrack dies
- **Structural equations:** $T1 = 1 - A$
- May have uncertainty about causal model

Blameworthiness

- Likelihood of event on performing one action vs another

$$\delta_{a,a',\varphi} = max\left(\left[\sum_{(M,\mathbf{X})\in[\![[A\leftarrow a]\varphi]\!]} \Pr(M,\mathbf{X}) - \sum_{(M,\mathbf{X})\in[\![[A\leftarrow a']\varphi]\!]} \Pr(M,\mathbf{X})\right], 0\right)$$

- **Degree of blame** <- attempt actions with lower costs

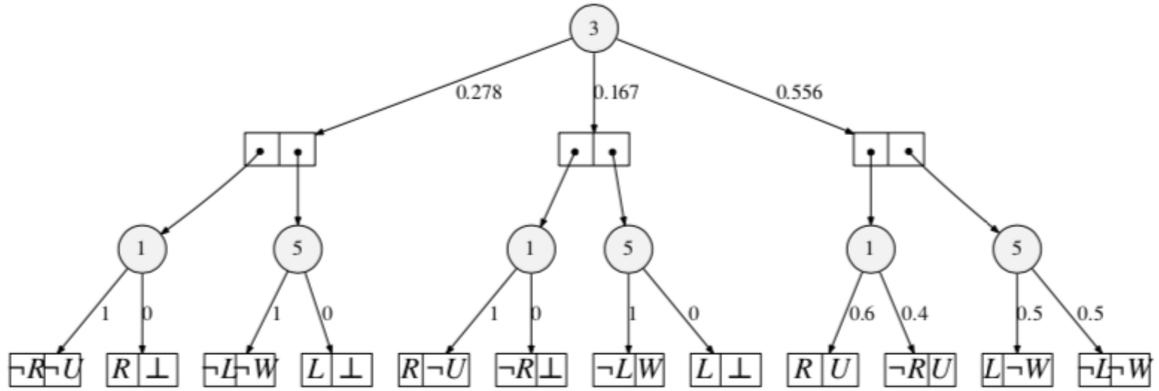
Halpern and Kleiman-Weiner (2018)

Can you reason tractably about moral scenarios?

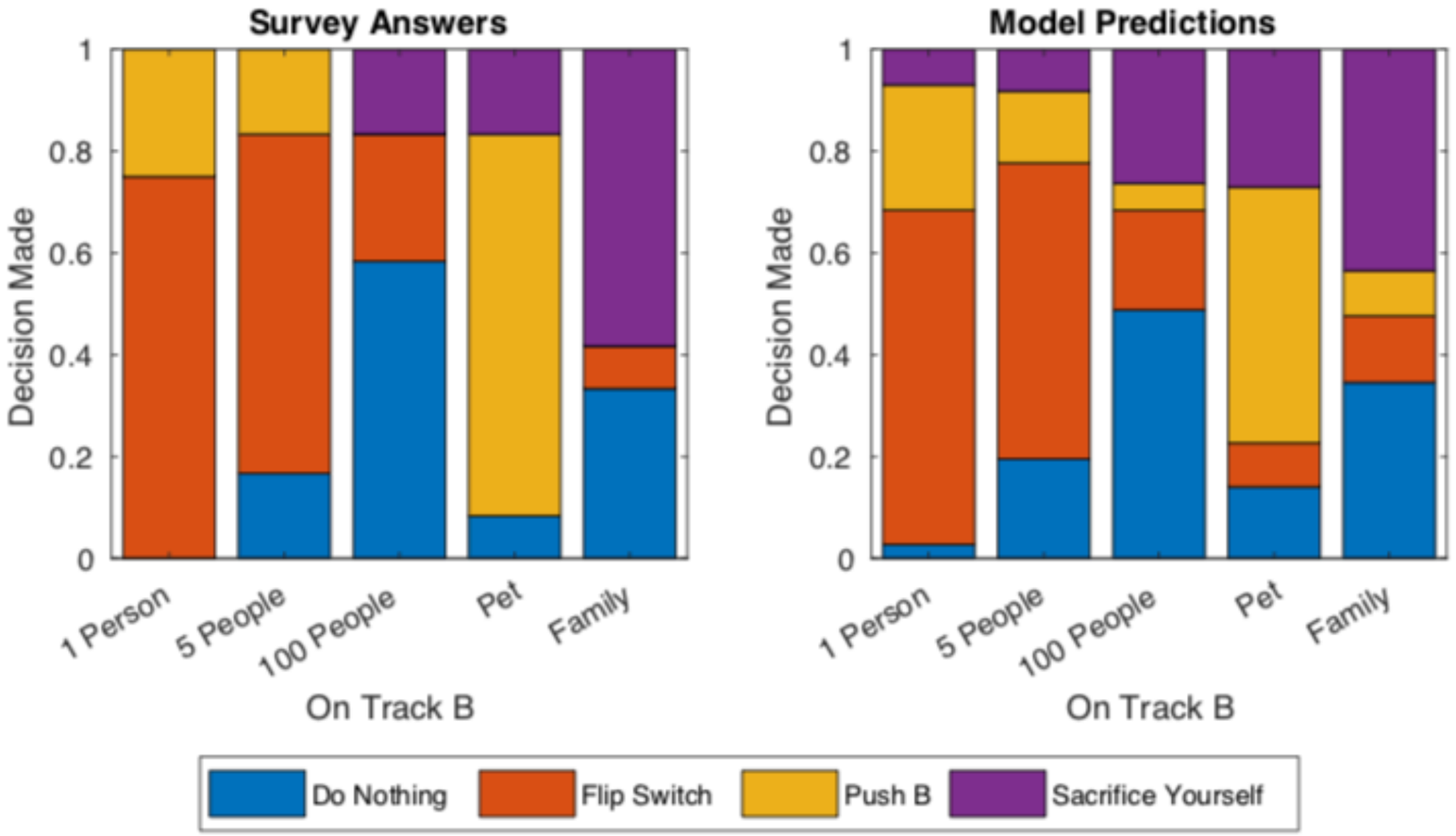
Hammond and Belle (2018)

Circuit for structural equations

Number of data points	360
Number of variables	23
Context variables ($\mathcal{X}$)	One Person On Track A (A_1), ... , Family On Track A (A_{Fa}), One Person On Track B (B_1), ... , Family On Track B (B_{Fa})
Decision variables ($\mathcal{D}$)	Inaction (I), Flip Switch (F), Push B (P), Sacrifice Oneself (S)
Outcome variables ($\mathcal{O}$)	One Person Lives (L_1), ... , Family Lives (L_{Fa}), You Live (L_Y)
Constraints	$\bigvee_{i\in\{1,...,Fa\}} A_i$ $\bigvee_{i\in\{1,...,Fa\}} B_i$ $\neg(A_i \wedge B_i) \forall i \in \{1, ..., Fa\}$ $A_i \rightarrow \neg \bigvee_{j\in\{1,...,Fa\}\setminus i} A_j \forall i \in \{1, ..., Fa\}$ $B_i \rightarrow \neg \bigvee_{j\in\{1,...,Fa\}\setminus i} B_j \forall i \in \{1, ..., Fa\}$ $\bigvee_{D\in\{N,F,P,S\}} D$ $D \rightarrow \neg \bigvee_{D'\in\{N,F,P,S\}\setminus D} D'$ $(A_i \wedge N) \rightarrow \neg L_i \forall i \in \{1, ..., Fa\}$ $(B_i \wedge N) \rightarrow L_i \forall i \in \{1, ..., Fa\}$ $L_i \rightarrow (A_i \vee B_i) \forall i \in \{1, ..., Fa\}$ $(S \wedge (A_i \vee B_i)) \rightarrow L_i \forall i \in \{1, ..., Fa\}$ $L_Y \leftrightarrow \neg S$ $(L_i \wedge (P \vee F)) \rightarrow \neg \bigvee_{j\in\{1,...,Fa\}\setminus i} L_j \forall i \in \{1, ..., Fa\}$ $(\neg L_i \wedge (P \vee F)) \rightarrow \bigvee_{j\in\{1,...,Fa\}\setminus i} L_j \forall i \in \{1, ..., Fa\}$
Model count	180
Utilities given?	No

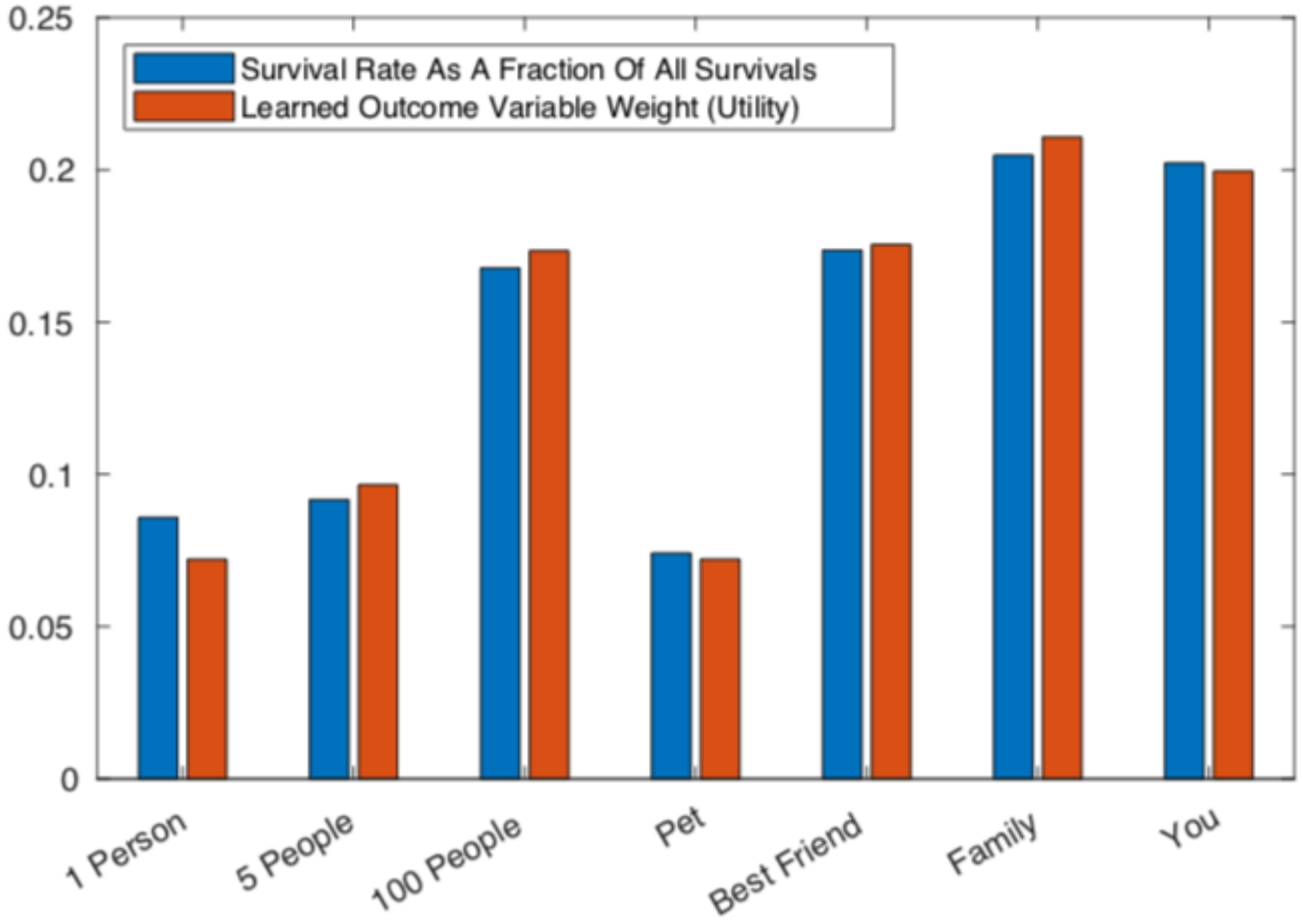


Alignment of decisions



Survey people to learn costs (here: consider situation with best friend on main track)

Learned costs



Some challenges

- Consequentialist stance (cardinal utility function vs ordinal), not uncontroversial, but commonly made (QALY, value of life)
- Not meant to be prescriptive, but like an "ethics bot" (reflect majority, but can be restricted to experts)
- Unclear if **formal definitions** (fairness, blame) fully capture various viewpoints
  - But deferring to humans may not be realistic in time-sensitive scenarios
- Perhaps AI can help us understand if **alignment is even possible** between human agenda and numeric objectives
  - But do we have common values? People disagree significantly on such issues across race, gender, culture, age, etc.

From discrete to continuous: brief notes

From discrete to continuous and/or countably infinite?

- Propositional logic: discrete (finite outcome) distributions only
- Planning, robotics, web mining, vision: discrete and continuous (hybrid) random variables, e.g., Gaussian nodes with discrete parents
- Reasoning with first-order generalizations: countably infinite domains, e.g., parent of parent is grandparent for entire infinite universe
- Literature: variational methods with asymptotic guarantees or analytical results for some families (e.g., Gaussians)

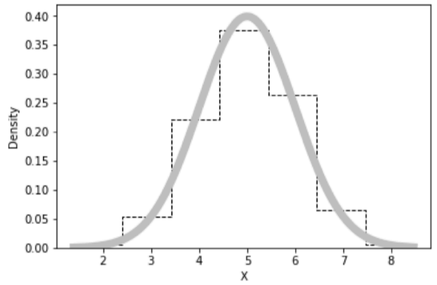
Can we engineer general-purpose methods that can deal with complex logical constraints in infinite domains?

Weighted Model Integration

$$\mathrm{WMI}(\Delta, w) = \sum_{M \models \Delta^{-}} \int \prod_{l^{+} \in M} w(l)$$

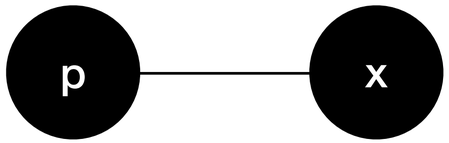
$$
\begin{array}{ll}
 & -3 \leq u \leq 3 \\
0.043 & u \leq -1.5 \\
0.241 & -1.5 < u \leq -0.5 \\
0.383 & -0.5 < u \leq 0.5 \\
0.241 & 0.5 < u \leq 1.5 \\
0.043 & 1.5 < u
\end{array}
$$

$$
\begin{array}{ll}
 & -3 \leq u \leq 3 \\
(2+u)^3/6 & -2 < u \leq -1 \\
(4-6u^2-3u^3)/6 & -1 < u \leq 0 \\
(4-6u^2+3u^3)/6 & 0 < u \leq 1 \\
(2-u)^3/6 & 1 < u < 2
\end{array}
$$



Belle et al. (2015); Kolb et al. (2018); Zeng & Van Den Broeck (2019)

Example Markov network



$\dfrac{p \vee (0 \leq x \leq 10)}{p \vee q}$	$w(p) = .1, w(\neg p) = 2x, w(q) = 1, w(\neg q) = 0$

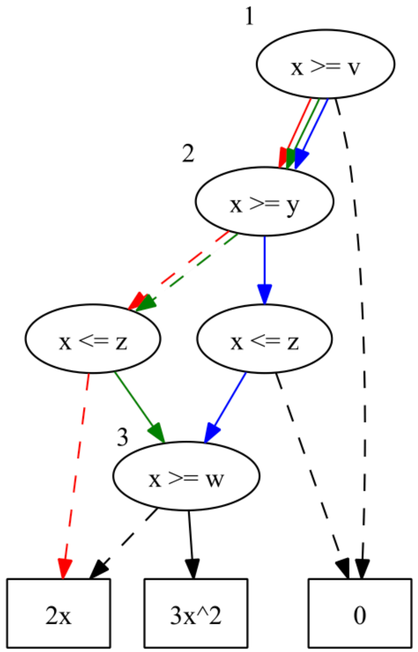
$$WMI(p \vee q) = VOL(p \wedge q) + VOL(p \wedge \neg q) + VOL(\neg p \wedge q)$$

$$VOL(p \wedge q) = \int_{0 \leq x \leq 10} .1 \; dx = 1$$

$$VOL(\neg p \wedge q) = \int_{0 \leq x \leq 10} 2x \; dx = 100$$

Slightly more complex example
With XADDs

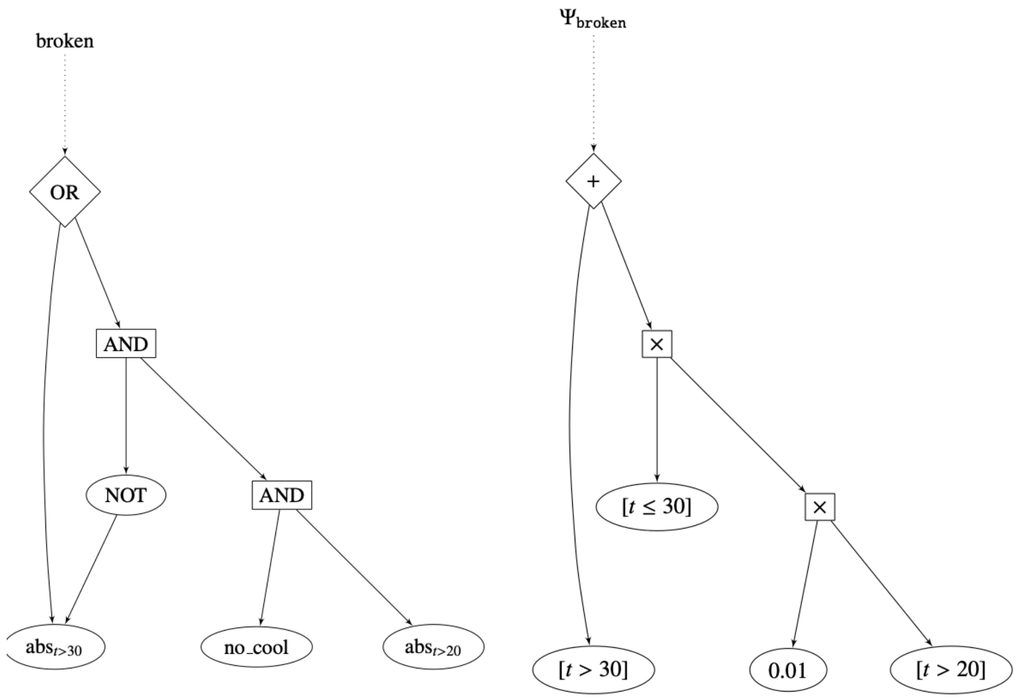
$$\theta = x \geq v \wedge (x < y \vee x \leq z), w = ite(x \leq z \wedge x \geq w, 3x^2, 2x)$$



Kolb et al. (2018)

Can exploit SDDs

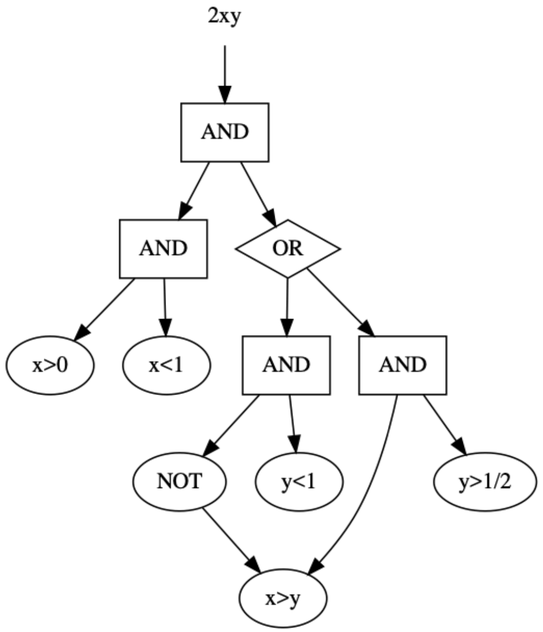
$$broken \leftrightarrow (nocool \wedge (t > 20)) \vee (t > 30)$$

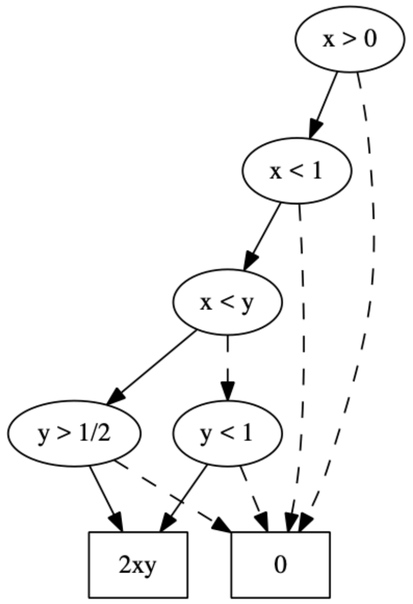


Dos Martires et al. (2019)

XADDs vs SDDs

$$[(x > 0) \wedge (x < 1)] \wedge [(y < 1) \vee ((x > y) \wedge (y > 1/2))]$$





Kolb et al. (2019)

Countably infinite domains (e.g., $\mathbb{N}$, $\mathbb{Z}$)

Reason with $\forall x, y, z : \mathbb{N}(parent(x, y) \wedge parent(y, z) \supset grandparent(x, z))$

- In the absence of the finite domain assumption (**open universe**), reasoning in the first-order setting suffers from undecidability properties
- **Forward reasoning**, where samples needed for probability estimation are obtained from the facts and declarations in the probabilistic model: each sample = possible world
  - But there may be (countably or uncountably) infinitely many worlds, and so exact inference is usually sacrificed
- Restrict the model wrt the query and evidence atoms and define estimation from the resulting finite sub-model: **local grounding**

Russell (2015); Belle (2017)

Open-universe WMC

- Recall with $\forall x : D\ Smoker(x)$ and $|D| = 2$, we get 2 atoms. So if $D = \mathbb{N}$, then infinitely many atoms
- However, define OUGND as grounding wrt constants mentioned + a few arbitrarily chosen new ones (=number of vars)
- **Theorem: $\Delta \models \alpha$ (Clausal, QF resp.) iff $\Delta \wedge \neg\alpha$ is UNSAT iff $OUGND(\Delta \wedge \neg\alpha)$ is UNSAT**
  - E.g., with $\forall x : D\ Smoker(x)$ and $D = \{A, B\}$, we'd need one extra constant, say C

Example

- Let Δ be the union of $\forall(Smoker(x) \wedge Friends(x, y) \supset Smoker(y))$, $Smoker(john)$, and $\forall y(Friends(john, y))$
- Let $\alpha = Smoker(jane)$
- Apply theorem to see that indeed $\Delta \models \alpha$

Audience question: what’s been your exposure on continuous features? Or mixed features? Or countably infinite features?

Machine learning for logic

Reveals how logic can used for pushing the boundaries of learnable representations

History & motivation

- Inductive reasoning has been a core issue for the logical worldview, as we need a mechanism for obtaining axiomatic knowledge (at least since Socrates)
- Learning of logical and symbolic artifacts is an important issue in AI, and computer science more generally (e.g., program induction, plan synthesis)
- Considerable work in propositional & finite automaton settings

In propositional context

- **Entailment-based scoring**: Given *L*, background knowledge $B \subset L$, examples *D*, find a hypothesis $H \in \overline{H}$, $H \subset L$ such that $B \cup H$ entail the instances in *D* (the set $\overline{H}$ places restrictions of the syntax of *H* so as to control model complexity and generalization)

- **Likelihood-based scoring:** pick *H* such that $score(H, D) > score(H', D)$ where $score(H, D) \propto \log \Pr(D \mid H) - size(H)$

- *Many variants possible:* Bayesian scoring, neural synthesis, provably correct *H* wrt unknown ground truth (e.g., PAC semantics)

Can we learn such programs
(i.e., generative model)?

(a) parameter (b) structure

An instance of entailment-based scoring

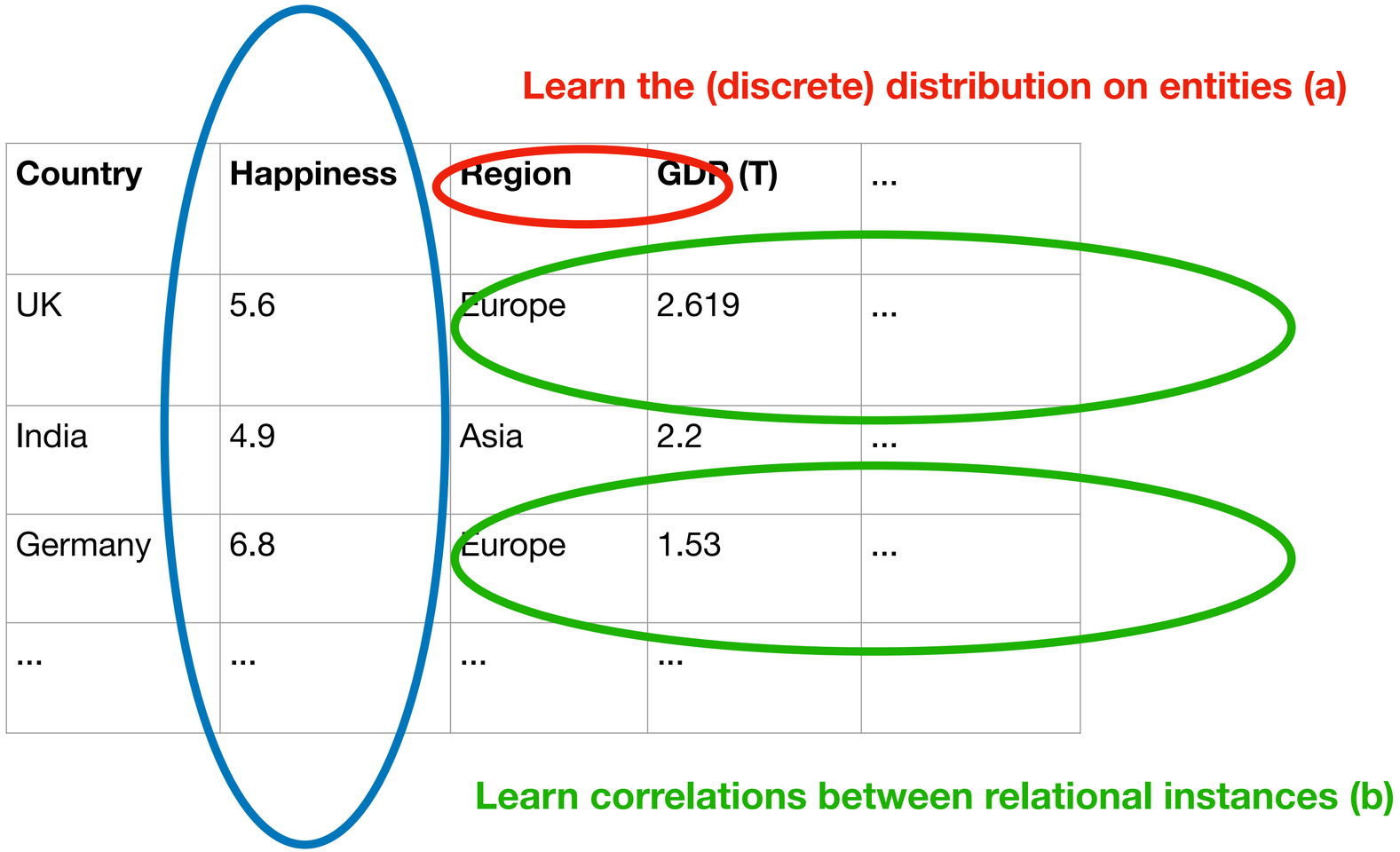
Probabilistic rule learning

- Dataset contains 10567 facts. Each fact has a probability value attached
  - E.g., $0.934 :: athleteplaysforteam(thurman\_thomas, buffalo\_bills)$
- Find hypothesis such that $H = argmin_H loss(H, B, E)$
- and $loss(H, B, E) = \sum_{(x_i,p_i)\in E} |P(B \cup H \models x_i) - p_i|$
- Where *E* is a set of examples, consisting of pairs (x_i, p_i), where x_i is a ground fact for the unknown target predicate *t* and p_i is a target probability

De Raedt et al (2015)

0.9375::athleteplaysforteam(A,B)	←	athleteledsportsteam(A,B).
0.9675::athleteplaysforteam(A,B)	←	athleteledsportsteam(A,V1), teamplaysagainstteam(B,V1).
0.9375::athleteplaysforteam(A,B)	←	athleteplayssport(A,V1), teamplayssport(B,V1).
0.5109::athleteplaysforteam(A,B)	←	athleteplaysinleague(A,V1), teamplaysinleague(B,V1).
0.9070::athleteplayssport(A,B)	←	athleteledsportsteam(A,V2), teamalsoknownas(V2,V1), teamplayssport(V1,B), teamplayssport(V2,B).
0.9070::athleteplayssport(A,B)	←	athleteplaysforteam(A,V2), teamalsoknownas(V2,V1), teamplayssport(V1,B), teamplayssport(V2,B),teamalsoknownas(V1,V2).
0.9070::athleteplayssport(A,B)	←	athleteplaysforteam(A,V1), teamplayssport(V1,B).
0.9286::athleteplaysinleague(A,B)	←	athleteledsportsteam(A,V1), teamplaysinleague(V1,B).
0.7868::athleteplaysinleague(A,B)	←	athleteplaysforteam(A,V2), teamalsoknownas(V2,V1), teamplaysinleague(V1,B).
0.9384::athleteplaysinleague(A,B)	←	athleteplayssport(A,V2), athleteplayssport(V1,V2), teamplaysinleague(V1,B).
0.9024::athleteplaysinleague(A,B)	←	athleteplaysforteam(A,V1), teamplaysinleague(V1,B).

Discrete-continuous tabular data



Country	Happiness	Region	GDP (T)	...
UK	5.6	Europe	2.619	...
India	4.9	Asia	2.2	...
Germany	6.8	Europe	1.53	...
...	...	...	...	

Unsupervised framework

- Learn distributions for individual relations

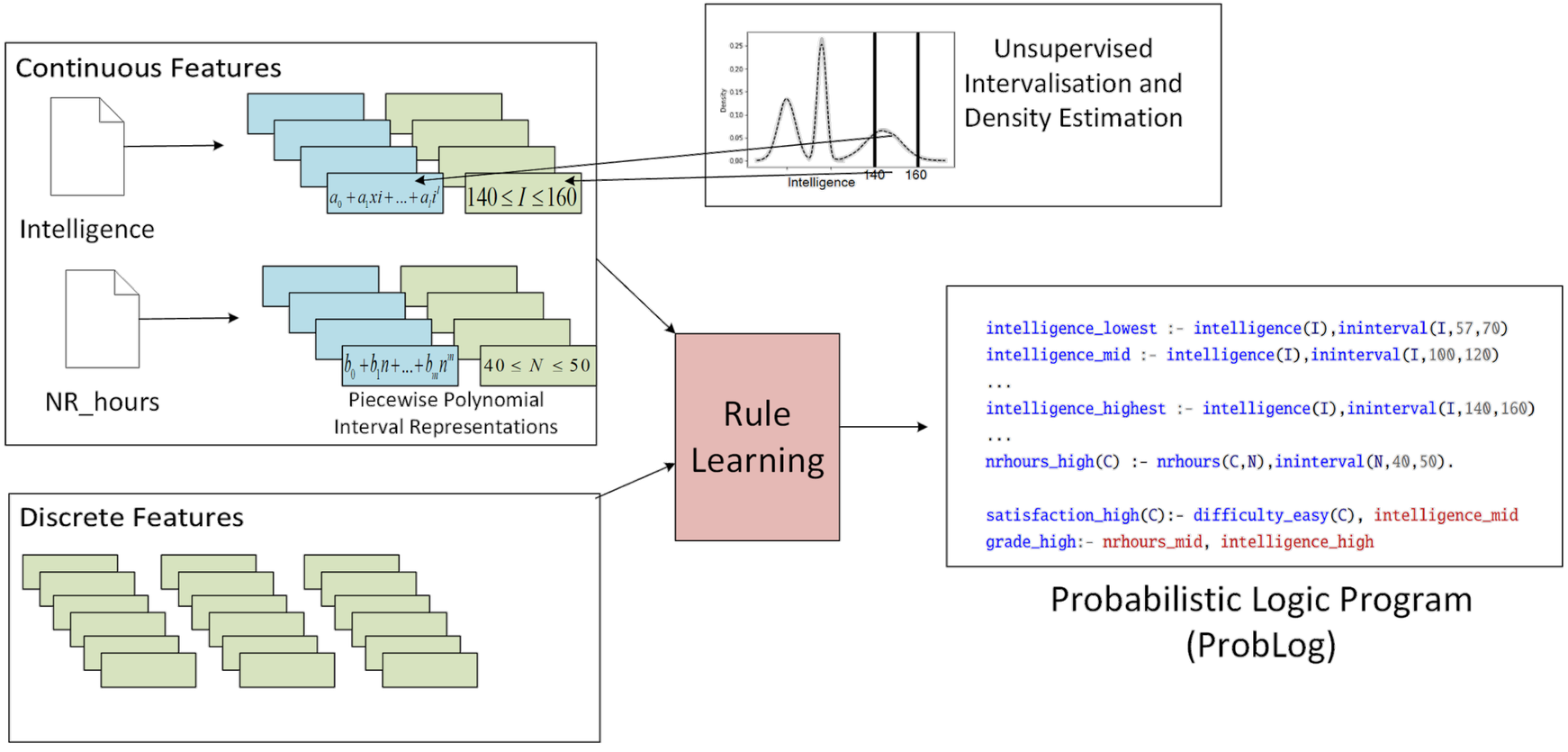
$$\overrightarrow{\delta}(X) = \begin{cases} 0 & x < \mathsf{c}_0 \\ \delta_1(x) & \mathsf{c}_0 \leq x \leq \mathsf{c}_1 \\ \cdots & \\ \delta_l(x) & \mathsf{c}_{l-1} \leq x \leq \mathsf{c}_l \\ 0 & x > \mathsf{c}_l \end{cases}$$

$$BIC(x, \overrightarrow{\delta}(x)) = \mathscr{L}(x \mid \overrightarrow{\delta}(x)) - \frac{S \times log(|X|)}{2}$$

- Learn correlations: **input —** instances (examples) + *background knowledge (e.g., ontologies)*

$$\min_{H} \textbf{loss}(H, B, E)$$

Speichert & Belle (2018)


Continuous Features
Intelligence
NR_hours
$a_0 + a_1 xi + ... + a_i i^i$
$140 \leq I \leq 160$
$b_0 + b_1 n + ... + b_m n^m$
$40 \leq N \leq 50$
Piecewise Polynomial
Interval Representations
Discrete Features
Rule Learning
Density
Intelligence
140
160
Unsupervised
Intervalisation and
Density Estimation
intelligence_lowest :- intelligence(I),ininterval(I,57,70)
intelligence_mid :- intelligence(I),ininterval(I,100,120)
...
intelligence_highest :- intelligence(I),ininterval(I,140,160)
...
nrhours_high(C) :- nrhours(C,N),ininterval(N,40,50).
satisfaction_high(C):- difficulty_easy(C), intelligence_mid
grade_high:- nrhours_mid, intelligence_high
Probabilistic Logic Program
(ProbLog)

Learned program

```
grade_high(A,B) :- sat_low(A,B), \+diff_hard(B), \+iq_1(A,X).
grade_high(A,B) :- takes(A,B), \+diff_hard(B).
```

```
grade_high(A,B) :- nrhours_4(B,Y), iq_1(A,X).
grade_high(A,B) :- nrhours_4(B,Y), course(B).
grade_high(A,B) :- sat_low(A,B)
```

```
-0.099723+0.0015473 * Y :: nrhours_4(B,Y):-
 course(B), nrhours(Y),between(64.45,78.73,Y)
0.05636635932439037 3 -0.0015982833846236427* X
+ 1.2573927973878 37e-05 * X ^ 2 ::iq_1(A,X):-
student(A),iq(X),between(52.6,103.4,X)
```

Another instance

```
%discrete relation and facts
student(S).
0.8::luck.

%Interval declaration
6.0408699785088995 -0.09831811654956664* I**1 +0.00040032255503358305* I**2 ::
int_high(S,I):-intelligence(S,I),between(I,114,124).

%Rules learned by rule learner
rank_distinction(S):-intelligence_high(S,I),luck

% evidence
evidence(student(john)).

%query
query(rank_distinction(john)).
```

$$\int_{114}^{124} 6.04 - 0.0983 * I + 0.0004003 * I**2 \, dI = 0.1331$$

0.1331 * 0.8 = 0.10648

P(rank_distinction(john)) = 0.10648

Speichert & Belle (2019)

From interpretable representations to scalable, explainable querying

An instance of likelihood-based scoring

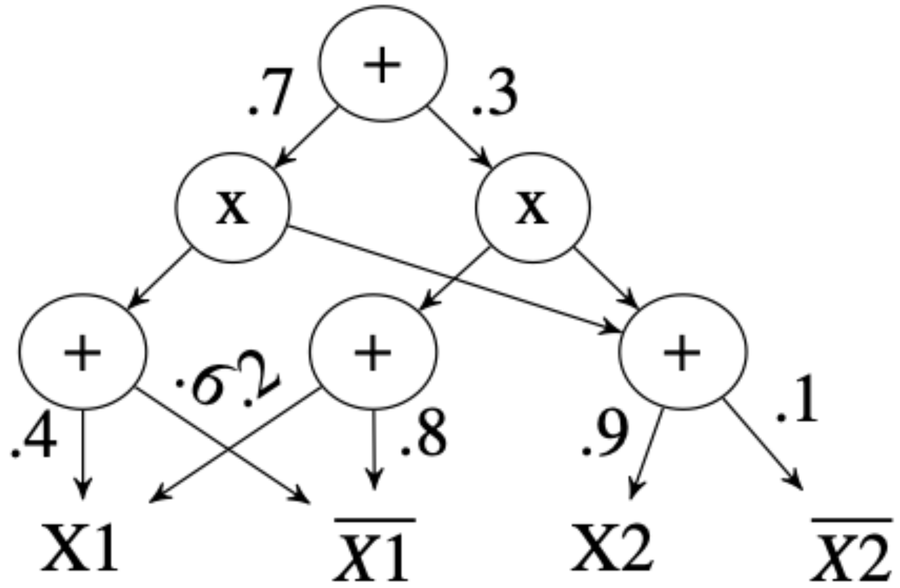
What if structure is irrelevant?

- Very large, very granular, etc.
- Perhaps we only need to provide a **query interface** to users

$$\Pr(gender = male \mid height > 180 \ \wedge \ weight > 200) = ?$$

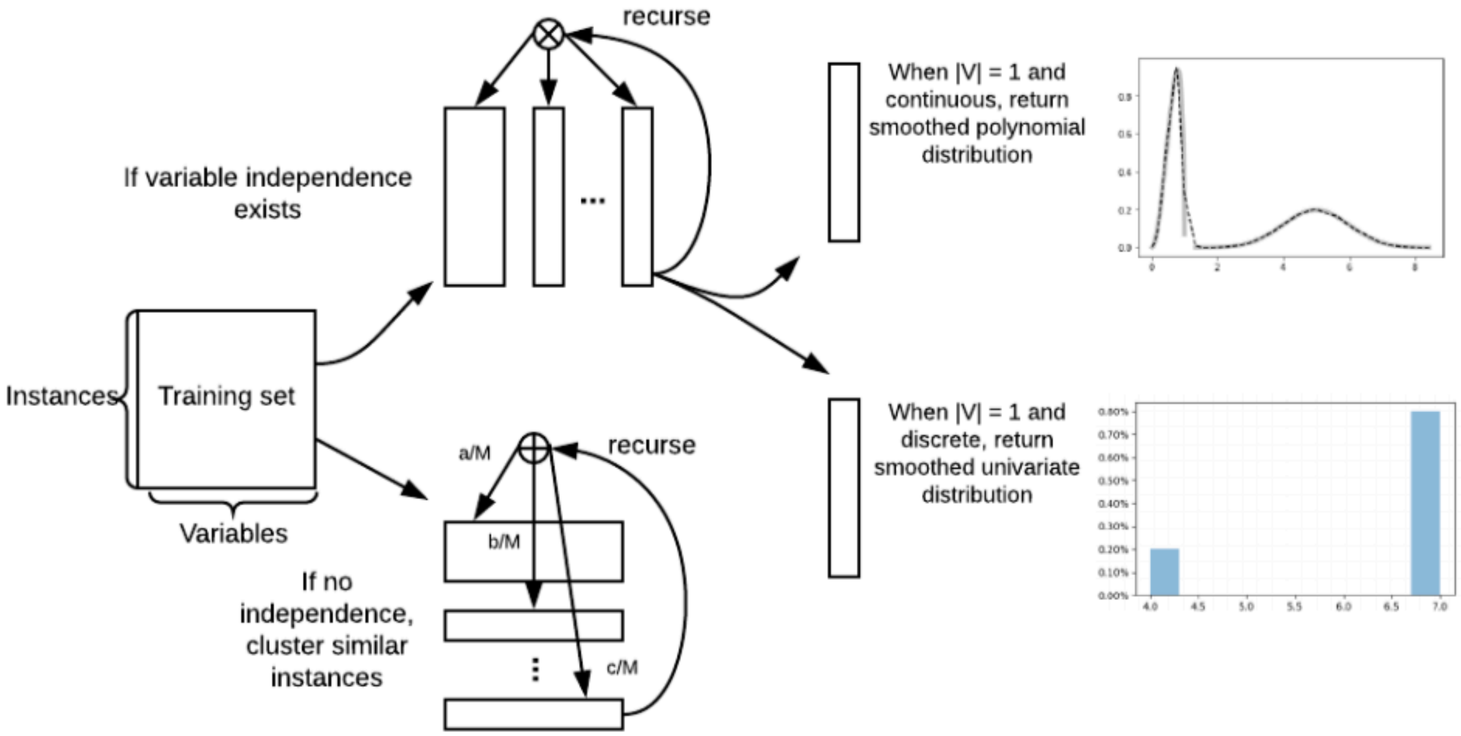
Recap: tractable models

- Leaf nodes: tractable univariate distributions, weighted sums of products (i.e., weighted mixtures)
- Certain computations linear in size of circuit (recall WMC & knowledge compilation)
- Standard deep models $P(y \mid x)$ vs here: $P(x, y)$
- Learning: $\log \mathscr{L}(S \mid \mathscr{D})$



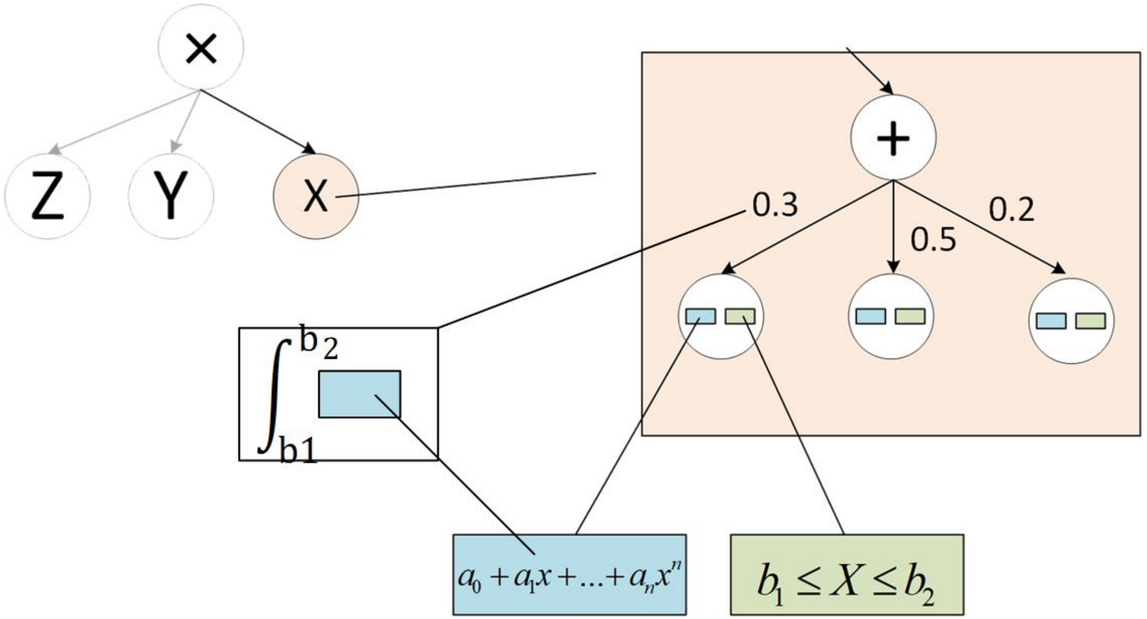
Darwiche (2000); Poon & Domingos (2011)

Approach for mixed discrete continuous data



Tractable querying

- Unsupervised regime
- Probabilistic semantics
- Dynamic query decomposition
- Tractable (requires linear passes on the network)



$$\Pr(gender = male \mid height > 180 \ \wedge \ weight > 200) = ?$$

Bueff, Speichert & Belle (2018)

Dataset statistics

Dataset	$\|V\|$	$\|V'\|$	Train	Valid	Test
anneal-U	38	95	673	90	134
australian	15	50	517	69	103
auto	26	85	119	16	23
car	9	50	294	39	58
cleave	14	35	222	29	44
crx	15	54	488	65	97
diabetes	9	33	576	76	115
german	21	76	750	99	150
german-org	25	70	750	99	150
heart	14	35	202	27	40
iris	5	11	112	15	22

Comparisons

Dataset	WMI-SPN	Gower-MSPN		RDC-MSPN	
	LearnWMISPN	hist	iso	hist	iso
anneal-U	**-14.543**	-63.553	-38.836	-60.314	-38.312
australian	**-10.473**	-18.513	-30.379	-17.891	-31.021
auto	**-27.126**	-72.998	-69.405	-73.378	-70.066
car	**-9.111**	-30.467	-31.082	-29.132	-30.516
cleave	**-13.829**	-26.132	-25.869	-29.132	-25.441
crx	**-10.525**	-22.422	-31.624	-24.036	-31.727
diabetes	**-6.299**	-15.286	-26.968	-15.930	-27.242
german	**-20.429**	-40.828	-26.852	-38.829	-32.361
german-org	**-21.144**	-43.611	-26.852	-37.450	-27.294
heart	**-14.875**	-20.691	-26.994	-20.376	-25.906
iris	-3.560	-3.616	-2.892	-3.446	**-2.843**

Model complexity

Dataset	Bins	2nd-Order	3rd-Order	4th-Order	5th-Order	6th-Order
australia	2	0	16.667	16.667	33.3	33.3
australia	5	16.667	33.333	16.667	16.667	16.667
auto	2	15.8	36.842	31.579	12.789	0
auto	5	73.684	15.789	10.526	0	0
german-org	2	0	33.333	0	0	66.666
german-org	5	33.333	33.333	0	33.333	0
heart	2	0	20	20	60	0
heart	5	0	60	40	0	0
iris	2	50	0	50	0	0
iris	5	75	25	0	0	0
statlog	2	42.857	0	0	0	57.143
statlog	5	28.571	57.143	0	14.289	0
cloud	2	10	10	40	10	30
cloud	5	30	50	10	0	10

Beyond intervals (weighted & unweighted)

- Combining discrete and Gaussian atoms in probabilistic logic programs: e.g., $\texttt{pos(id)}_{\texttt{t+1}} \sim \texttt{gaussian}(\simeq \texttt{pos}_{\texttt{t}}\texttt{(id)}, 0.2) \leftarrow \simeq \texttt{move}_{\texttt{t}} = \texttt{id}$
- Oblique constraints — e.g., $x + 3 > y$ — by linear hypothesis generation from (noise-free) measurements of $x, y, \ldots$
- Learning hypothesis in linear and non-linear domains wrt unknown ground truth (i.e., PAC semantics, cf. following slides)
  - Given $x + y > z, x > 5, \ldots$ and noisy measurements for variables $z = .1, z = .9, \ldots$, does it follow that $y > z$?

Nitti et al. (2016); Kolb et al (2019); Mocanu et al (2020)

What about countably infinite?

- Recall that for **continuous:** we see finitely many real-valued samples as being drawn from an (unknown) interval, and we could inspect these samples to crudely infer a lower and upper bound
- Based on finitely many relational atoms, we would need to infer
  $\forall x, y, z : \mathbb{N}(parent(x, y) \wedge parent(y, z) \supset grandparent(x, z))$

What justifies this? How can finitely many samples allow us to infer properties about infinitely many things?

Belle & Juba (2019)

Could we leverage open-universe scheme?

Theorem: $\Delta \models \alpha$ **(Clausal, QF resp.) iff** $\Delta \wedge \neg\alpha$ **is UNSAT iff** $OUGND(\Delta \wedge \neg\alpha)$ **is UNSAT**

- Suppose given *B* (possibly containing universally quantified formulas), and given QF query α, can finitely many observations allow us to distinguish between positive and negative infinitary statement?
- **Yes, if we draw enough samples then it can be shown that with high probability, a first-order formula can be learned**
- However, intractable to learn explicit hypothesis!

Example

- Suppose background knowledge is
  $\forall[x \neq logan \supset Mutant(x)], \forall[x \neq y \supset Mutant(x) \wedge Team(x, y) \supset Mutant(y)]$
- Observe $\{Team(scott, logan), Team(jean, logan)\}$
- Query $Mutant(logan)$ is **entailed**
- Because $\forall[x \neq logan \supset Team(x, logan)]$ is implicitly induced
- And, **background knowledge** $\cup$ **implicit theory** $\models$ **Query**

Audience question: what’s been your experience with structure learning?

Logic for Machine Learning

Reveals how logic can be used for correctness, data efficiency and compositionality

Many dimensions

- Machine learning methods that leverage symbolic domain knowledge (relational embeddings, loss calibration)
- Hybrid KR languages that combine human-specified and learned predicates
- Logical alternatives to standard ML (e.g., circuits)
- Verifying machine learning models by logical specification or formulation
- **Here:** preview *probabilistic programming* & *abstraction*: **logic as meta-theory**

Distinction between finite vs infinite not so urgent, but could imagine FO constraints, for example

Recall: ProbLog

```
% Probabilistic facts:

0.5::heads1.

0.6::heads2.

% Rules:

twoHeads :- heads1, heads2.

% Query:

query(twoHeads).
```

```
% Mixture distribution

0.7::heads.

0.5::a.

0.6::b.

tails :- \+heads.

mix :- heads, a.

mix :- tails, b.
```

Probabilistic programming

- Support probabilistic primitives in the language, with the intention of making learning modules re-usable, modular and compositional
- Underlying computational task?

$$\text{WMC}(\Delta, w) = \sum_{M \models \Delta} \prod_{l \in M} w(l)$$

Fierens et al (2011); Holtzen et al (2020)

Algebraic model counting

$$AMC(\phi, w) = \oplus_{M \models \phi} \otimes_{l \in M} w(l)$$

- Generalizes WMC wrt **commutative semirings**, allowing soft constraints, gradients etc.

- What about **continuous properties? Non-standard semantics? Composition?**

(semiring) program = logic + semiring + solver

Kimmig et al (2012); Belle & De Raedt (2020)

```
(set-logic FOL)
(set-algebra [NAT,+,0])
(set-type COLOR={r,b,g})
(set-type NODE={1,2,3})
(declare-predicate node (NODE))
(declare-predicate edge (NODE,NODE))
(declare-predicate color (NODE,COLOR))
N = (node(1) and node(2) and node(3))
E = (edge(1,2) and edge(2,3) and edge(3,1))
DATA = (N and E)
CONS = /* coloring constraints (omitted) */
(declare-weight TRUE 1)
(count (DATA and CONS))
```

Counting graph coloring

```
(set-logic LRA)
(set-algebra [REAL,inf,0])
(declare-function input (INT,INT) REAL)
...  /* declare left, right, app */
(declare-function err () REAL)
DATA = /* entries in input matrix (omitted) */
F = app(x,y) == sum{e} left(x,e)*right(e,y)
G = err == norm(sum{x,y} input(x,y) - app(x,y))
(declare-weight TRUE err)
(count (DATA and F and G))
```

Matrix factorization

```
(set-logic PL)
(set-algebra [NAT,max,*,0,1])
(declare-predicate p ())
(declare-predicate q ())
F = (p or q)
(declare-weight (p 1))
(declare-weight ((neg p) 2))
(declare-weight (q 3))
(declare-weight ((neg q) 4))
(count F)
```

MPE

```
(set-logic QF_LIA;PL)
(set-algebra [NAT,max,0])
(set-type INT={1,...,10})
(declare-function x1 () INT)
(declare-function x2 () INT)
(declare-predicate p1 ())
(declare-predicate p2 ())
F = ((p1 or p2) => 3*x1 <= 4)
G = (p2 => (2*x2 <= 5))
H = ((F and G) and p2)
(declare-weight TRUE x1*x2)
(count H)
```

Logical-integer programming

	FT	FN	IT	IN	C	S	R
APROBLOG	✓	×	×	×	✓	✓	✓•
CHURCH	✓	×	✓	×	✓•	×	×
ESSENCE	×	✓	×	×	✓•	×	×
CSP	×	✓•	×	✓•	✓•	×	×
SEMIRING CSP	×	✓•	×	×	✓•	✓	×
AMPL	×	✓	×	✓	✓•	×	×
ASP	✓	✓•	×	×	✓	×	✓
(O)SMT	×	✓	×	✓	✓	×	×
#SMT	✓	×	✓	×	✓	×	×
SP	✓	✓	✓	✓	✓	✓	✓

A comparison, where F = finite, T = factorized, N = non-factorized, I = infinite, C = logical connectives and quantifiers, R = non-standard, S = semiring apparatus, and • denotes that a feature is available in a restricted sense.

Abstraction is a powerful idea widely used in science, to model, reason and explain the behavior of systems at the appropriate granularity, by omitting **irrelevant** details

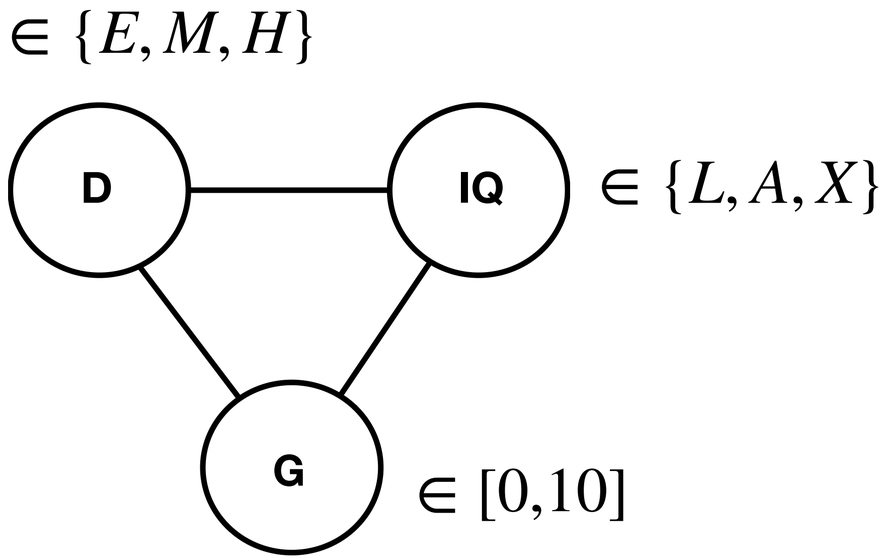
- Relate **high-level concepts** to low-level sensory data and learning

- Structure knowledge, hierarchically, for explanations, commonsense reasoning, etc.

- Abstracting problem domain to a smaller search space (even with tractable representations)

Example(s) of possible abstractions

- Collapse domain (e.g., $\mathbf{D} \in \{E, \overline{E}\}$)
- Make **G** a discrete variable
- Obtain a **qualitative** abstraction (e.g., alert admin if low-IQ students fail easy course)



- What is a **faithful alignment** between two probabilistic (relational) models $\mathscr{H}$ & $\mathscr{L}$?
- How can we incorporate a low-level observations to maintain alignment?
- How do we synthesize $\mathscr{H}$ given $\mathscr{L}$ and a vocabulary?

Given a mapping from high-level **atoms** to low-level **formulas,** ensure that every $\mathscr{H}$-model has a $\mathscr{L}$-model that agrees on truth wrt mapping

Then, stipulate that probabilities of $\mathscr{H}$-atoms = probability of mapped $\mathscr{L}$-formulas

Framework

- **Refinement mapping** *m:* Given Δ_h, Δ_l,

$$\forall p \in \mathscr{L}(\Delta_h), m(p) = \theta_p \in \mathscr{L}(\Delta_l)$$

  - E.g., $grade(fail)$ could be mapped to $grade(1) \vee \ldots \vee grade(7)$

- **Isomorphisms:** for $M_h \in \mathscr{M}(\Delta_h), M_l \in \mathscr{M}(\Delta_l)$

$$M_h \sim_m M_l \doteq \forall p \in \mathscr{L}(\Delta_h), M_h \models p \text{ iff } M_l \models m(p)$$

- Require that $\forall M_h, \exists M_l \,.\, M_h \sim_m M_l$, and vice versa

Some results

$$alert \doteq \exists x[IQ(x, L) \wedge takes(x, y) \wedge D(y, E) \wedge G(x,0)]$$

- **Theorem:** Theories at least agree on certain and improbable events
  - But you could also additionally require **probabilities of atomic events are equal**, in which case you get **conditional queries have the same probabilities**
- How to synthesize high-level abstraction given a low-level model?
  - Fix a vocabulary (set of relations and constants)
  - **Theorem:** Worst-case exponentially many formula guesses, but certain types of syntactic substitutions are correctness preserving

Some results

- **Theorem:** Without requiring probabilistic alignment, theories at least agree on certain and improbable events
- **Theorem:** With alignment, conditional probabilities agree
- Incorporating observations only possible after **weakening**
- Provide **algorithm for synthesis** with worst-case exponentially many formula guesses, but certain types of syntactic substitutions are correctness preserving

Audience question: have you encountered the need for hybrid problem solving, and/or abstraction?

Neuro-symbolic landscape

Joint work with Jonathan Feldstein, Paulius Dilkas and Efi Tsamoura

Motivation

- Pitfalls of classical logic: unambiguous conclusions but hard constraints, can be hard to come by
- Pitfalls of deep learning: highly scalable, but not data efficient and may not respect domain constraints
- Health knowledge graphs, biomedical databases, robotics, language models, vision understanding, among others, need to integrate reasoning and learning: **integrate deep learning and logic**

Key concerns

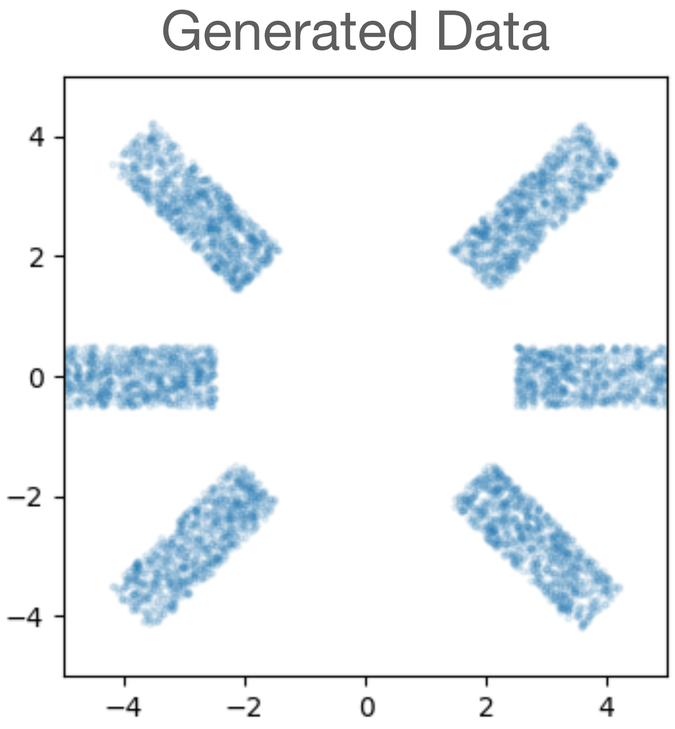
- Including Logical knowledge
- Learning of additional knowledge, eg via NNs with labeled training examples
- Representing the logical and learned knowledge (possibly as a hybrid artifact, involving either logical or neural or both formalisms)
- Executing the model (i.e., reasoning with learned knowledge)
- Contrasting and comparing the performance of the model against one without domain knowledge

Besold et al. (2021)

Three popular schemes

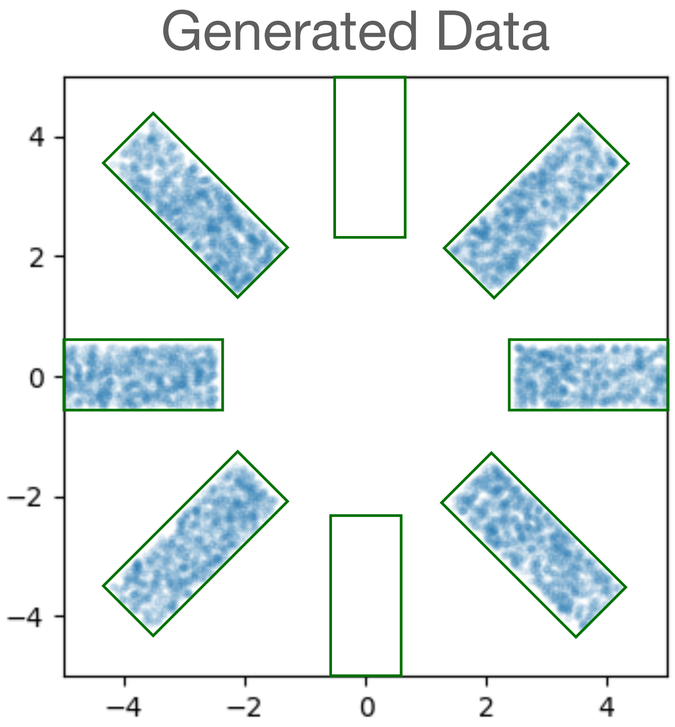
- A fully neural representation of logical artifacts: e.g., Neural theorem provers provide an end-to-end differentiable regime.
  - The key idea is to provide a recursive scheme for replicating the functionality of backward chaining.
  - The background knowledge is converted to a vector representation, against which a proof by this recursive scheme is differentiable.
- Neural artifacts in logical engines, e.g. DeepProbLog
- Symbolic logic as regularisation — *only shown here*

Motivating Example - Density Estimation Task



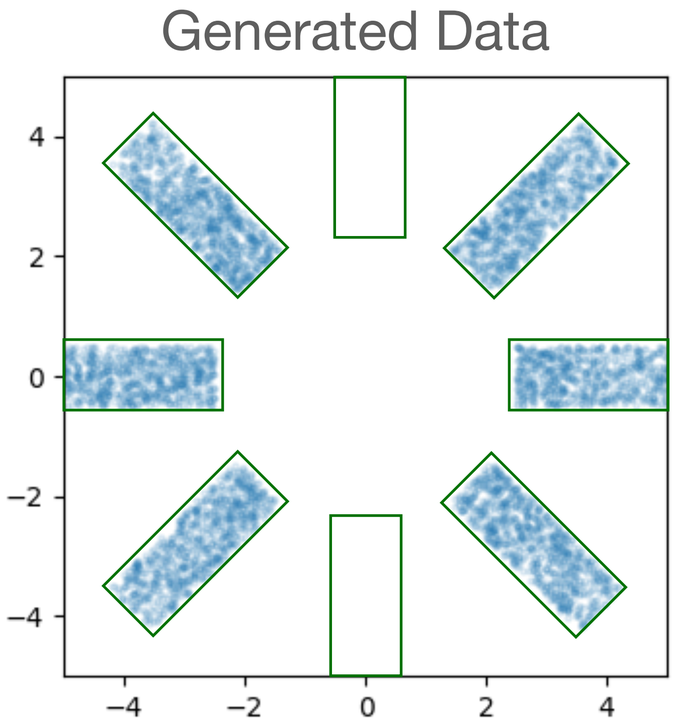
Hoernle et al (2021)

Motivating Example - Knowledge of Domain Constraints



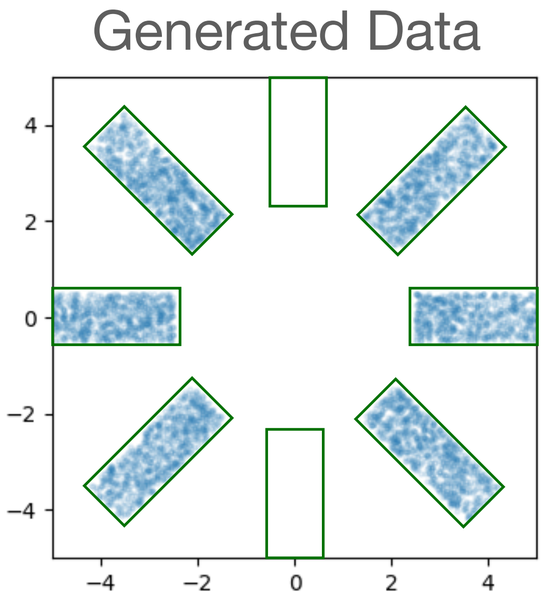
$$\Phi = (x_1 > -.5 \wedge x_1 < .5 \wedge x_2 > .5 \wedge x_2 < 4) \vee \ldots$$
$$\ldots \vee (x_1 + x_2 > -.5 \wedge x_1 + x_2 < .5 \wedge x_1 - x_2 > .5 \wedge x_1 - x_2 < 4)$$

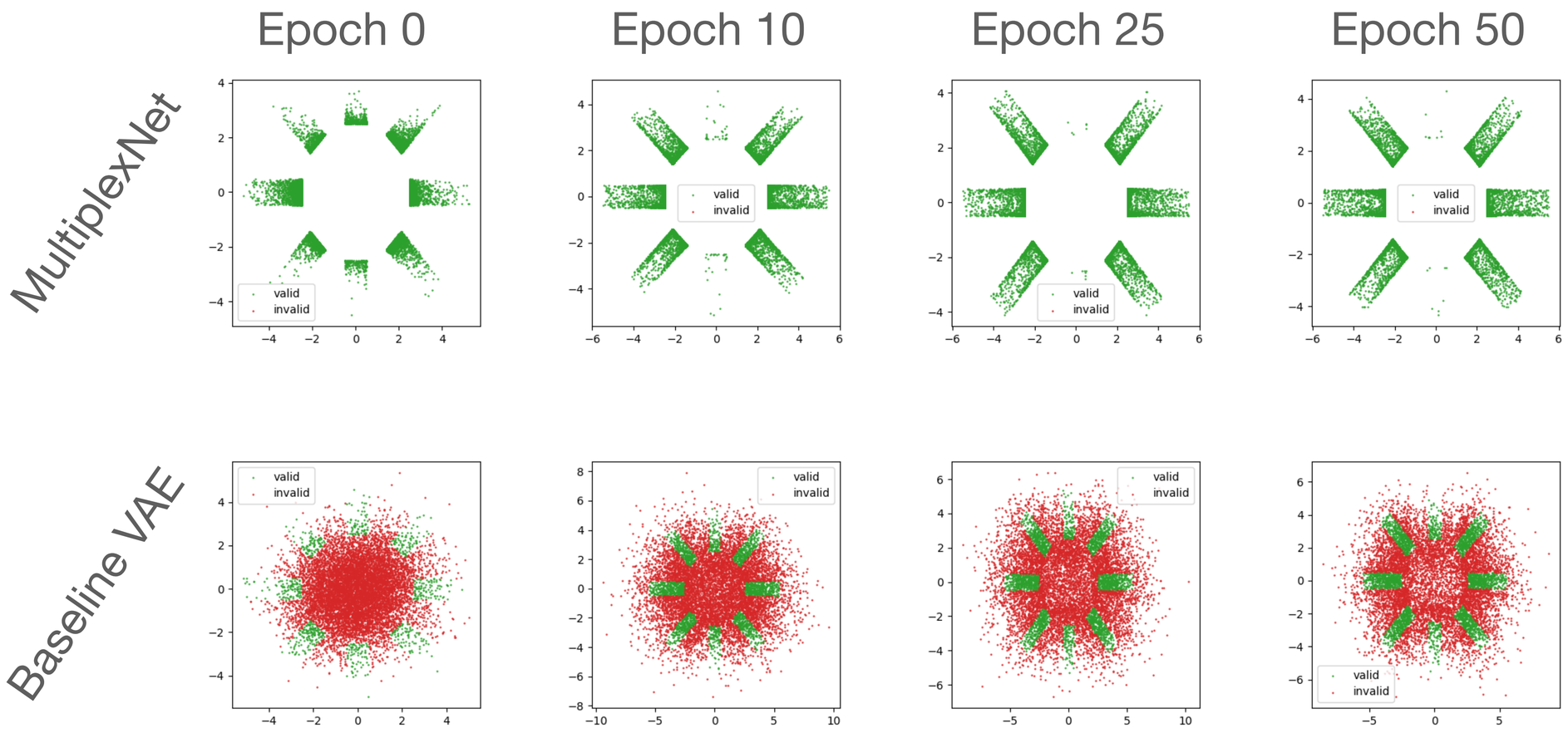
Motivating Example - How to use Constraints in Training?



$$\Phi = (x_1 > -.5 \wedge x_1 < .5 \wedge x_2 > .5 \wedge x_2 < 4) \vee \ldots$$
$$\ldots \vee (x_1 + x_2 > -.5 \wedge x_1 + x_2 < .5 \wedge x_1 - x_2 > .5 \wedge x_1 - x_2 < 4)$$

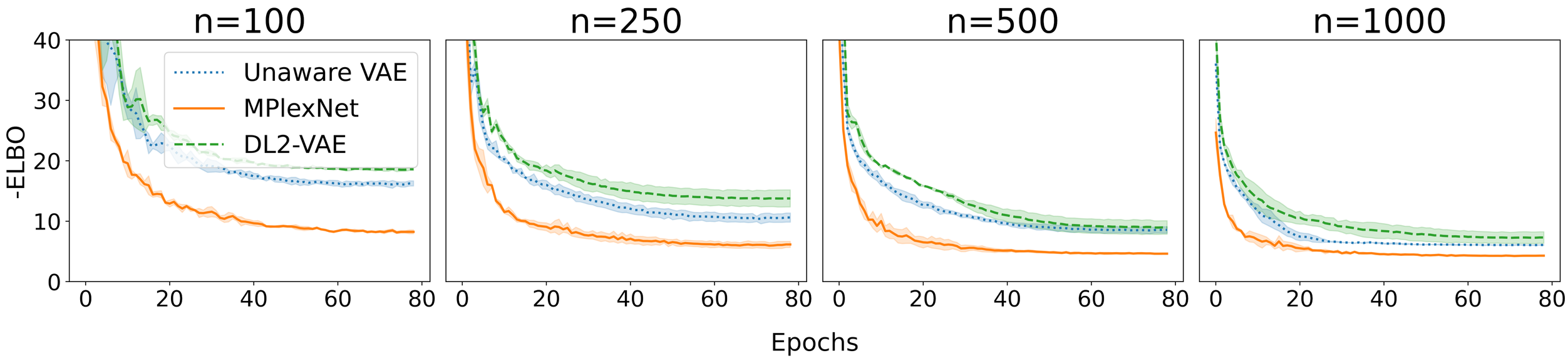
Motivating Example - Force Constraint Satisfaction



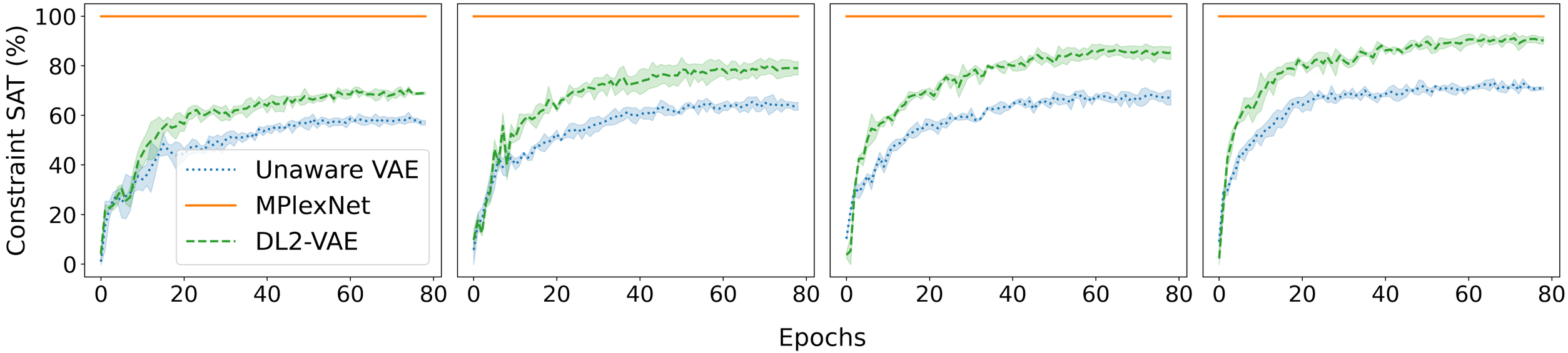


Motivating Example - Desiderata

(1) Data Efficiency

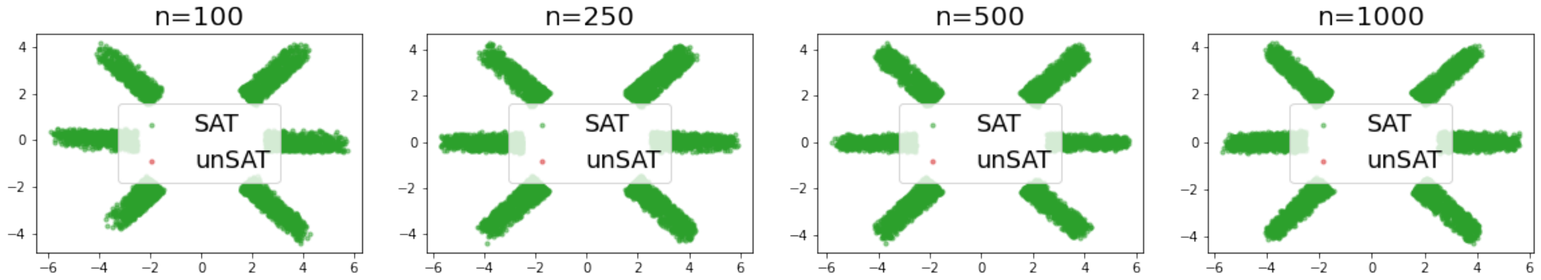


(2) Predictability (safety critical systems)

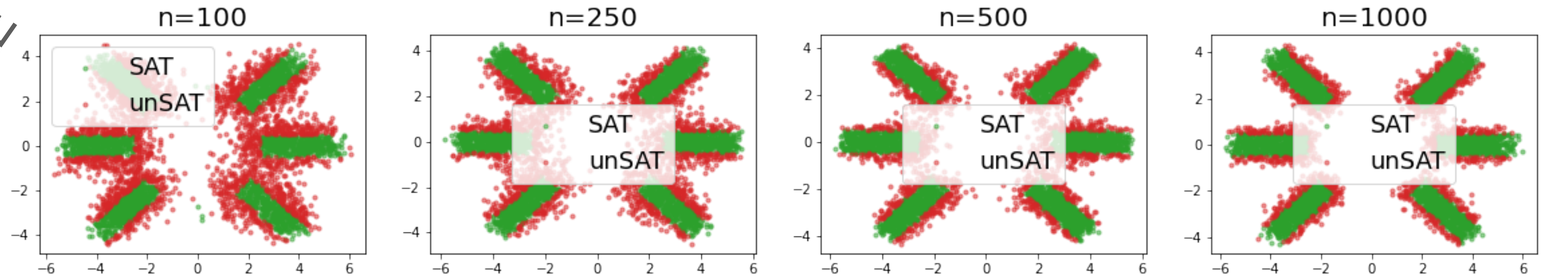


Posterior samples

MultiplexNet



Baseline VAE

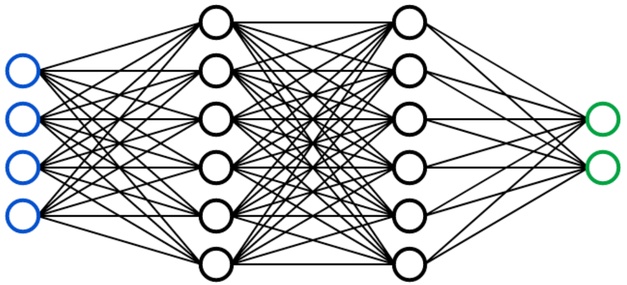


Constraining Probabilistic Models

(1) Given a dataset from unknown density p* but known to entail Φ:

$$X = \{x^{(0)}, \ldots, x^{(N)} \mid x^{(i)} \sim^{iid} p^*(x), x^{(i)} \models \Phi\}$$

Standard training



$p_\theta(x)$

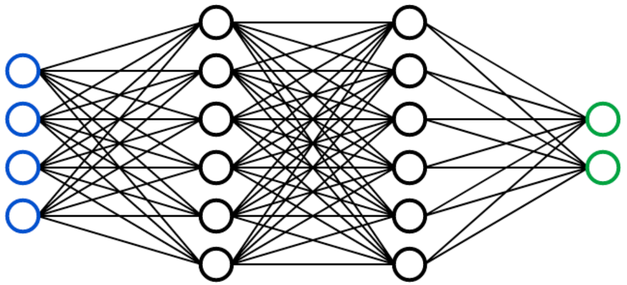
(1) Given a dataset from unknown density p* but known to entail Φ:

$$X = \{x^{(0)}, \ldots, x^{(N)} \mid x^{(i)} \sim^{iid} p*(x), x^{(i)} \models \Phi\}$$

(2) Train a parameterised model to maximise the likelihood of the data:

Design: $p_\theta(x)$

Train: $p_{\theta*}(x) = \arg\max_\theta(\log p_\theta(X))$



$p_\theta(x)$

(1) Given a dataset from unknown density p* but known to entail Φ:

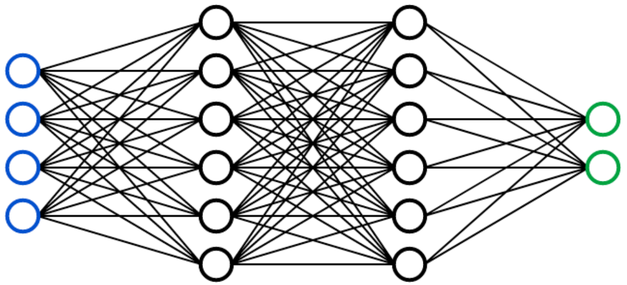
$$X = \{x^{(0)}, \ldots, x^{(N)} \mid x^{(i)} \sim^{iid} p^*(x), x^{(i)} \models \Phi\}$$

(2) Train a parameterised model to maximise the likelihood of the data:

Design: $p_\theta(x)$

Train: $p_{\theta^*}(x) = \arg\max_\theta(\log p_\theta(X))$

But what about Φ?



$p_\theta(x)$

(1) Append a loss term to training:

Train: $p_{\theta*}(x) = \arg\max_\theta [\log p_\theta(X) + L_\Phi(X)]$

(2) Reparameterise output of network:

Design: $p_\theta(x)$ *such that the output of the network follows* Φ *by construction.*

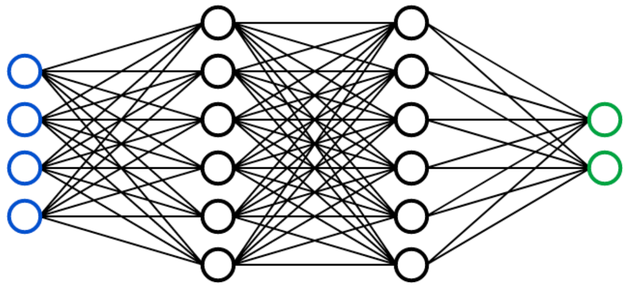
Fischer, M., Balunovic, M., Drachsler-Cohen, D., Gehr, T., Zhang, C. and Vechev, M., 2019, May. Dl2: Training and querying neural networks with logic. ICML

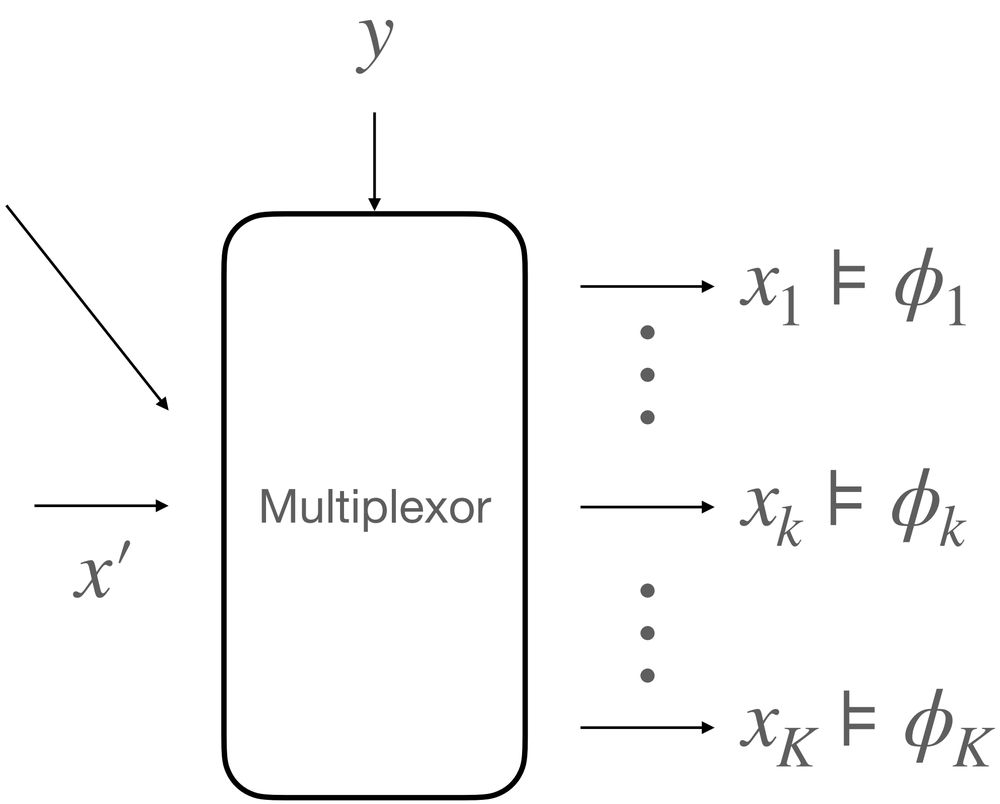
Xu, J., Zhang, Z., Friedman, T., Liang, Y. and Broeck, G., 2018, July. A semantic loss function for deep learning with symbolic knowledge. ICML

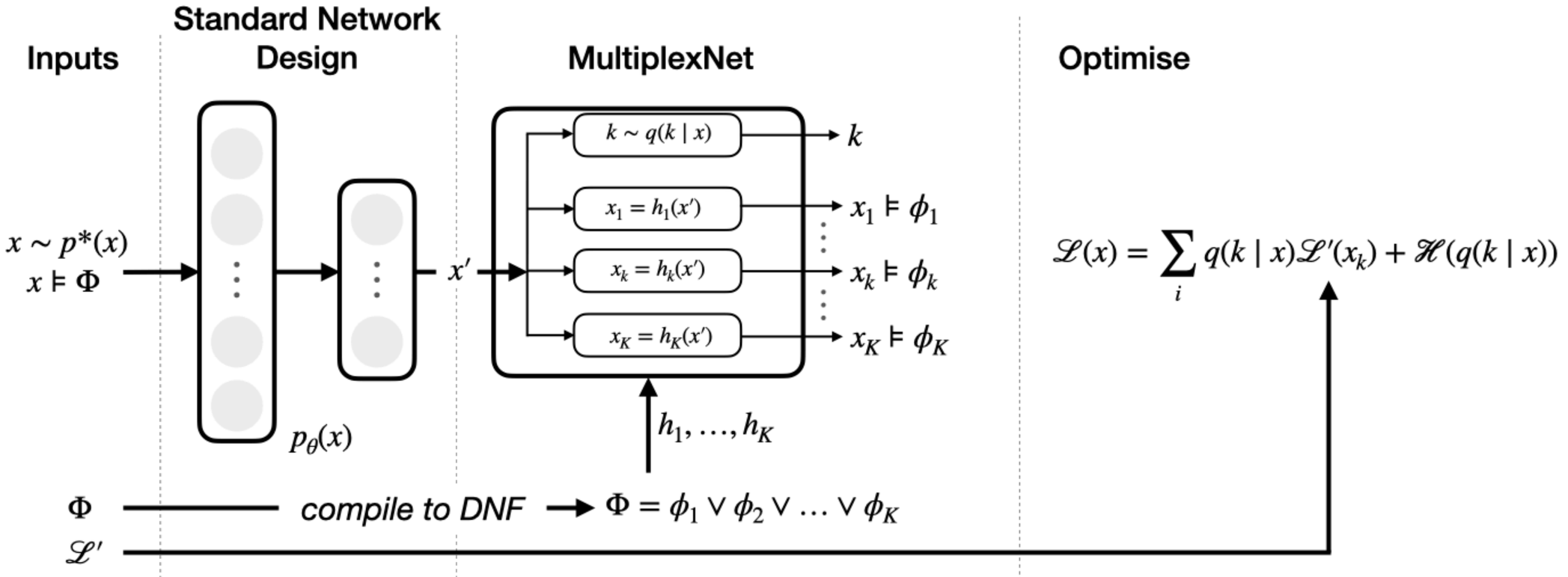
Innes, C. and Ramamoorthy, S., 2020. Elaborating on learned demonstrations with temporal logic specifications.

Multiplex architecture

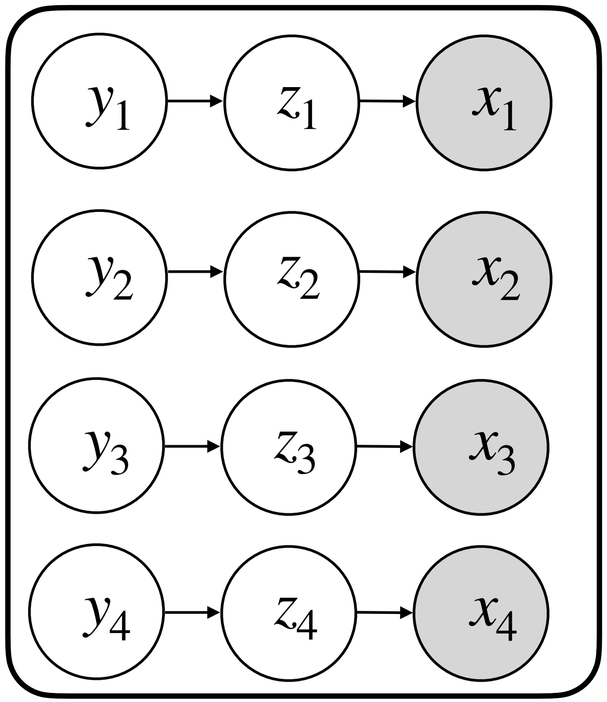
$$\Phi = \phi_1 \vee \phi_2 \vee \ldots \vee \phi_K$$






Inputs
Standard Network Design
MultiplexNet
Optimise
$x \sim p^*(x)$
$x \vDash \Phi$
$p_\theta(x)$
x'
$k \sim q(k \mid x)$
$x_1 = h_1(x')$
$x_k = h_k(x')$
$x_K = h_K(x')$
k
$x_1 \vDash \phi_1$
$x_k \vDash \phi_k$
$x_K \vDash \phi_K$
$h_1, \ldots, h_K$
Φ
compile to DNF
$\Phi = \phi_1 \vee \phi_2 \vee \ldots \vee \phi_K$
$\mathscr{L}'$
$\mathscr{L}(x) = \sum_i q(k \mid x)\mathscr{L}'(x_k) + \mathscr{H}(q(k \mid x))$

Label-free supervision



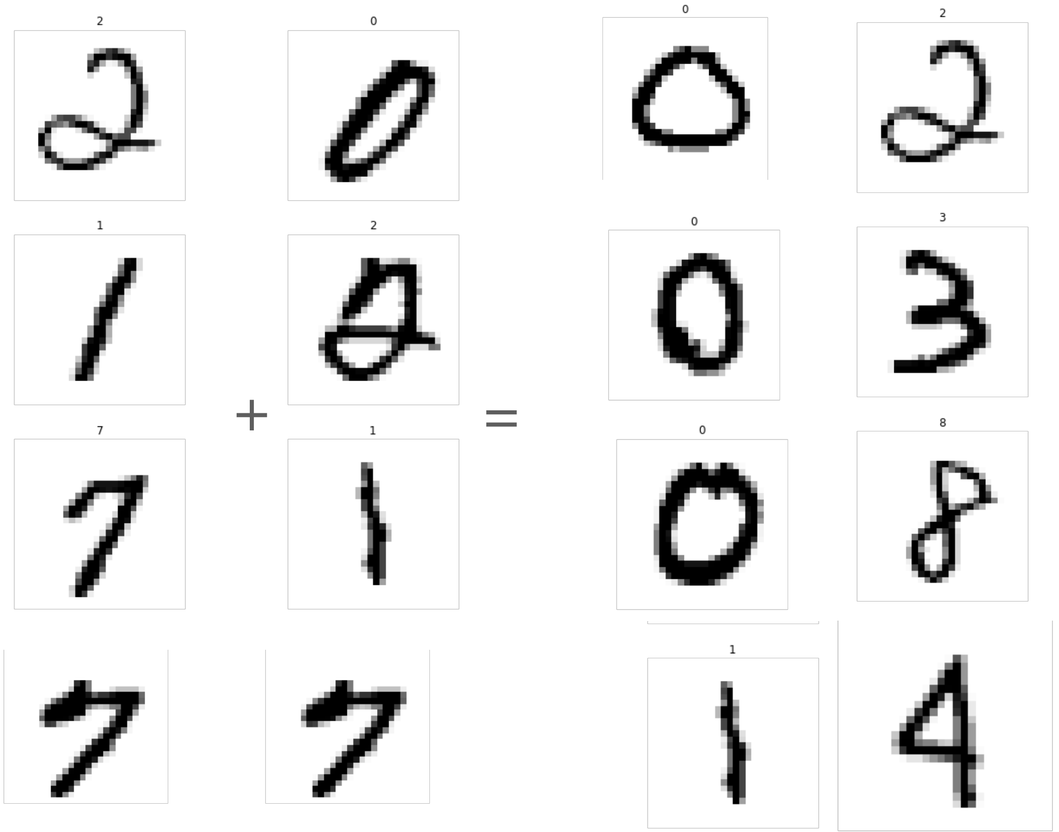


Image 1 + Image 2 = Image 3 Image 4

$$MPlexNet(\theta) = \sum_{k=1}^{K} \pi_k \left(\mathcal{L}_{h_k}(\theta) + \log \pi_{h_k} \right)$$

$$\mathcal{L}(\theta) = \sum_{i,j,k} \pi_h \big[V(x_1, y_1 = i) + V(x_2, y_2 = j) + V(x_3, y_3 = k_1) + V(x_4, y_4 = k_2) + \log \pi_h \big]$$

Many other intriguing topics

- Learning explanatory programs / circuits / formulas with neural machinery)
- Computing consistent explanations, approximate inference …

Barcelo (2021); Bertossi (2021); Darwiche (2022); Sarker and Hitzler (2022); Speichert and Belle (2017); Papantonis and Belle (2021); Belle (2021)

Audience question: have you delved into neuro-symbolic AI?

What did we cover?

- Quick illustrative examples of logical reasoning in various domains: protein graph database, electronic health records, social networks, inhibition effects, modeling unknowns in robotics
- Foundations of probabilistic logical inference: weighted model counting
- Tractable models, causality and counterfactuals
  - Fairness and moral reasoning
- Model counting in continuous + countably infinite domains
- Learning of programs and tractable models
- Other applications of model counting + abstraction
- Neuro-symbolic landscape + loss function approach for deep learning

Conclusions: Connections between logic & learning are deep

- *Alternative computational schemes:* logic-based solvers for constrained probabilistic reasoning and learning
- *Representation and expressiveness:* logic can help us model the relational, continuous & countably infinite
- *Contexualising*: verify, add domain knowledge, enable modularity and re-usability, combine declared and learned knowledge, meta-theory

On other hand, learning can address the fundamental question of how to arrive at symbolic knowledge (from data)