Neural Probabilistic Logic Programming in Discrete-Continuous Domains

Lennert De Smet, Pedro Zuidberg Dos Martires, Robin Manhaeve, Giuseppe Marra, Angelika Kimmig, Luc De Raedt

DeepSeaProbLog $=$ Neural Nets

$+$ Discrete-Continuous Probability Theory

$+$ Logic Programming

Setting

Neural Networks

Backpropagation

Symbolic Methods

Logic

Neural-Symbolic AI (NeSy) [1, 2]

Discrete-*Continuous* Probability theory

[1] Badreddine, S., et al. "Logic tensor networks." Artificial Intelligence 303 (2022).
[2] Xu, Jingyi, et al. "A semantic loss function for deep learning with symbolic knowledge." ICML (2018).

Neural Networks

Backpropagation

Symbolic Methods

Logic

Probabilistic NeSy [1, 2]

Discrete-*Continuous* Probability theory

[1] Manhaeve, R., et al. “Deepproblog: Neural probabilistic logic programming.” NeurIPS (2018).
[2] Yang, Z., Adam I. and Joohyung L. “Neurasp: Embracing neural networks into answer set programming.” IJCAI (2020).

Neural Networks

Backpropagation

Symbolic Methods

Logic

Probabilistic NeSy 🦕

Discrete-*Continuous* Probability theory

Why?

Why?

Continuous reasoning is crucial for robotics

Background

Existing Approaches

Approach	Neural	Symbolic	Discrete	*Continuous*
DPP	✓		✓*	✓
PLP		✓	✓	✓*
DPLP	✓	✓	✓	
?	✓	✓	✓	✓

Deep Probabilistic Programming [1, 2]

Key Idea

Model and optimise continuous, neurally parametrised probability distributions.

```
import tensorflow as tf # Deep
import tensorflow_probability as tfp # Probabilistic Programming

network = tf.Sequential(layers)
temp = tfp.distributions.Normal(network(data))
```

Logical constraints not supported!

[1] Bingham, E., et al. “Pyro: Deep universal probabilistic programming.” The Journal of Machine Learning Research 20.1 (2019).
[2] Dillon, J. V., et al. “Tensorflow distributions.” arXiv preprint arXiv:1711.10604 (2017).

Probabilistic Logic Programming [1, 2]

Key Idea

Declare discrete knowledge and infer probability of a logical statement.

```
humid ~ bernoulli(0.4).
cloudy ~ categorical(0.3, 0.4, 0.3).

rainy :- humid, cloudy =\= 0.

good_weather :- not rainy.
good_weather :- cloudy =:= 0.
```

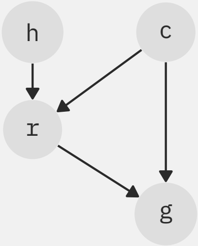
'It rains when humid **AND** clouds.'

'Good weather if no rain **OR** no clouds.'

[1] Dries, A., et al. "Problog2: Probabilistic logic programming." ECML PKDD (2015).
[2] de Morais, E. M. and Finger, M. "Probabilistic answer set programming." Brazilian Conference on Intelligent Systems (2013).

Programming Bayesian Networks [1]

```
humid ~ bernoulli(0.4).
cloudy ~ categorical(0.3, 0.4, 0.3).

rainy :- humid, cloudy =\= 0.

good_weather :- not rainy.
good_weather :- cloudy =:= 0.
```



[1] Darwiche, A. "Bayesian networks." Foundations of Artificial Intelligence 3 (2008).

Key Idea

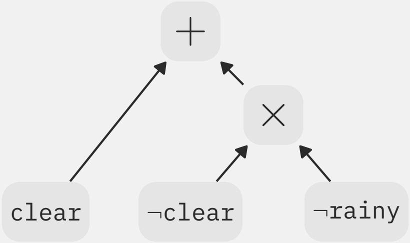
Neural networks parametrise random variables + end-to-end differentiable.

```
humid(🌍) ~ bernoulli(humidity_detector(🌍)).
cloudy(🌍) ~ categorical(cloud_detector(🌍)).

rainy(🌍) :- humid(🌍), cloudy(🌍) =\= 0.
good_weather(🌍) :- not rainy(🌍).
good_weather(🌍) :- cloudy(🌍) =:= 0.
```

Can not declare continuous probabilistic knowledge!

$$\texttt{good\_weather} \iff \texttt{clear} \lor \neg\texttt{rainy}$$

```
rainy ~ bernoulli(0.4).
clear ~ bernoulli(0.2).

good_weather :- clear.
good_weather :- not rainy.
```



Program
Non-differentiable and #*P*-hard.

Probabilistic Circuit
Differentiable and tractable structure.

DeepSeaProbLog

% Neural Distributional Fact (NDF)

detected_quasars ~ poisson(λ_{quasar}).

location() ~ normal([μ_x, μ_y, μ_z], uncertainty_estimation()).

DPP Application

Deep Probabilistic Programming to represent continuous distributions.

% Probabilistic Comparison Formula (PCF)

detected_quasars =:= 10. — ‘Only 10 quasars should be detected.’

f_{distance}(location(), location()) > α_{danger}. — ‘Distance  and  should be > α_{danger}.’

DeepSeaProbLog Program

```
humid(🌍) ~ bernoulli(humidity_detector(🌍)).
cloudy(🌍) ~ categorical(cloud_detector(🌍)).
temp(🌍) ~ normal(temperature_sensor(🌍)).

rainy(🌍) :- humid(🌍), cloudy(🌍) =\= 0.

good_weather(🌍) :-
    temp(🌍) > 20, not rainy.
good_weather(🌍) :-
    temp(🌍) < 0, rainy.
```

`query(good_weather(``)).`

Variables **predicted** by neural nets

‘It rains when humid **AND** clouds.’

‘Good weather if warm **AND** not rainy.’

‘Good weather when it snows.’

‘Probability good weather on Mars?’

Inference

Knowledge Compilation

Logic as tractable probabilistic circuit [1]

$$P(\mathrm{q}) = \int \sum \prod \mathbb{1}(c(x))\, p_{\wedge}(x)\, \mathrm{d}x$$

$$P(\texttt{temp(}\text{🌎}\texttt{) < 0}) = \int \mathbb{1}(x<0) \frac{\exp\left(-\frac{\left(x-\mu(\text{🌎})\right)^2}{2\sigma^2(\text{🌎})}\right)}{\sqrt{2\pi}\sigma(\text{🌎})}\, \mathrm{d}x$$

Weighted Integration

Neural distributions as weighing functions

[1] Zuidberg Dos Martires, P., Anton D. and De Raedt, L. "Exact and approximate weighted model integration with probability density functions using knowledge compilation." AAAI (2019).

Obstacle	Solution	Result
Indicators not differentiable $\mathbb{1}(c(x))$	$\xrightarrow{\text{relaxation}^1}$	$\sigma(\tau \cdot c(x))$
Sampling blocks gradient $\sigma(\tau \cdot c(x))\ p_\Lambda(x)$	$\xrightarrow{\text{reparametrisation}^2}$	$\sigma(\tau \cdot c(r_\Lambda(\boldsymbol{u})))\ p(\boldsymbol{u})$

Advantages

- Relaxations implemented in a straight-through manner
- Reparametrisation standard in DPP

[1] Petersen, F., et al. "Learning with algorithmic supervision via continuous relaxations." NeurIPS (2021).
[2] Ruiz, F. J. R., Titsias M. K. and David M. B. "The generalized reparameterization gradient." NeurIPS (2016).

Theorem
Unbiased gradients for $\tau \to +\infty$.

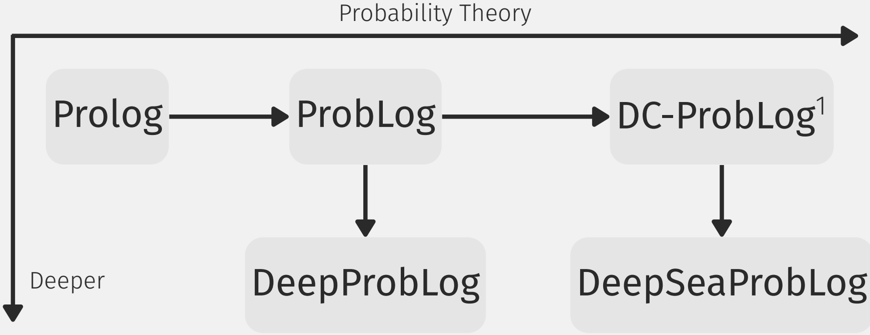
$$\partial_\lambda P(\mathbf{q}) \approx \int \partial_\lambda \sum \prod \sigma(\boldsymbol{\tau} \cdot c(r_\wedge(\boldsymbol{u}))) \, p(\boldsymbol{u}) \, \mathrm{d}\boldsymbol{u}$$

Efficient learning
Backpropagation through the deep, probabilistic-logical model

Overview

- Extended existing **Semantics**
- **Semantics** for gradients

- Turing complete language

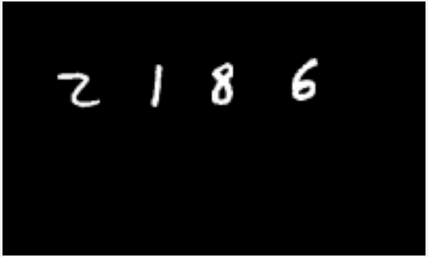


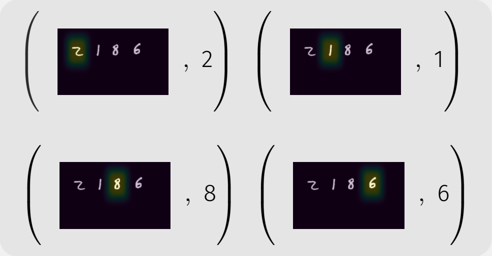
[1] Zuidberg Dos Martires, P., De Raedt, L. and Kimmig, A. “Declarative Probabilistic Logic Programming in Discrete-Continuous Domains.” arXiv preprint arXiv:2302.10674 (2023).

Experiments

Task

Predict + **localise** digits **without** bounding box supervision





Neural-Symbolic Attention

Box regression

Mask classification

```
box(X) ~ generalisednormal(X).

left_to_right( , , , ) :-
    box( ) < box( ),
    box( ) < box( ),
    box( ) < box( ).
```

Box ordering

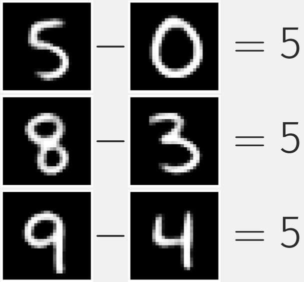
```
class(Mask) ~ categorical(class_net(Mask)).

supervise( , , , ) :-
    class( ) =:= 1, class( ) =:= 2,
    class( ) =:= 5, class( ) =:= 3.
```

Supervision

Method	Results	
	acc.	IoU
DeepSeaProbLog	93.77 ± 0.57	17.69 ± 0.23
LTN	76.50 ± 12.10	10.73 ± 1.69
Neural Baseline	54.71 ± 14.33	6.26 ± 1.77

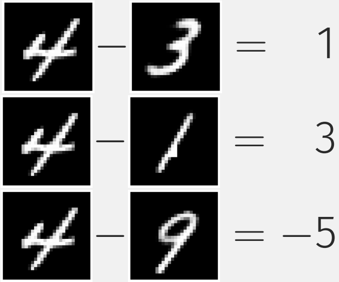
Task 1: Learning

Learn to fill in $\boxed{?}-\boxed{?}=5$



Task 2: Conditional Generation

Zero-shot completion of $-\boxed{?}=5$ in **same style**.



Conclusion

Conclusion

Main Contributions

- End-to-end differentiable declarative programming language
- Unbiased derivatives for sound learning
- Experimental argument for reasoning over discrete and continuous variables

Personal

Code

Twitter