

DeepProbLog: Neural Probabilistic Logic Programming

Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig,
Thomas Demeester*, Luc De Raedt*

* Joint last authors



DTAI



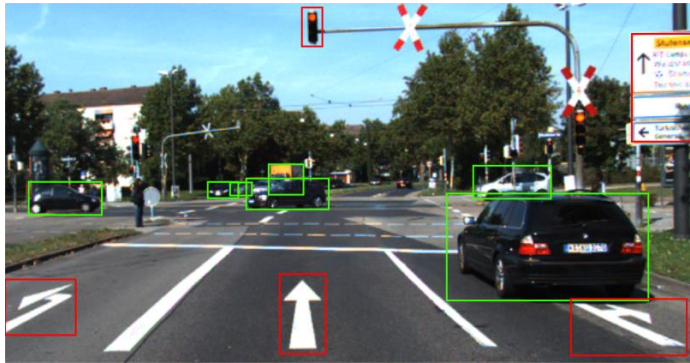
**GHENT
UNIVERSITY**

**CARDIFF
UNIVERSITY**

**PRIFYSGOL
CAERDYDD**

Real-life problems involve two important aspects.

Sub-symbolic perception



Reasoning with knowledge under uncertainty

Stop in front of a red light Obey the speed limit Be in the correct lane ...	$P(\text{light} = \text{red}) = 0.9$ $P(\text{obj1} = \text{car}) = 0.8$ $P(\text{obj1 turn right}) = 0.7$
---	--

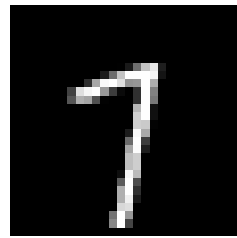
Deep Learning

Probabilistic logic program
ProbLog

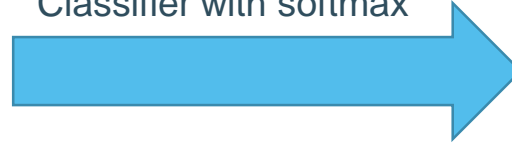


DeepProbLog
= ProbLog + neural predicate

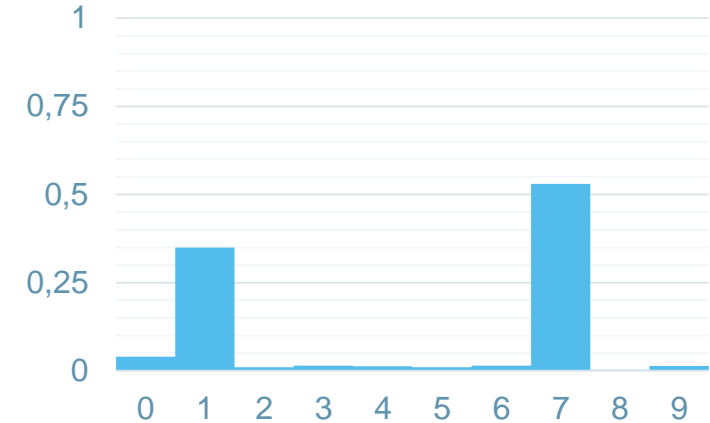
The neural predicate



Classifier with softmax

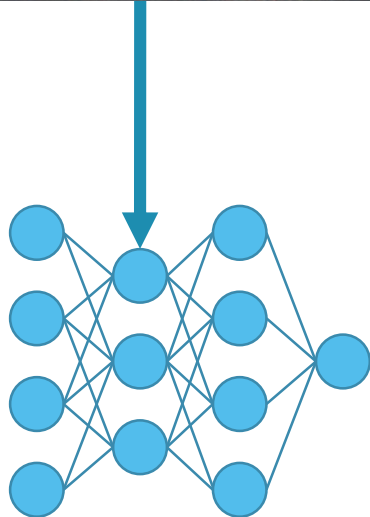
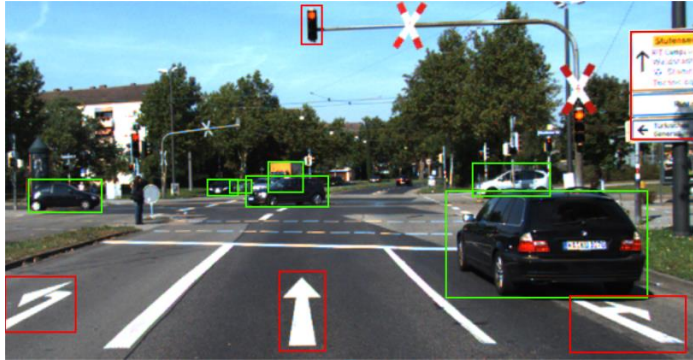


Probability distribution

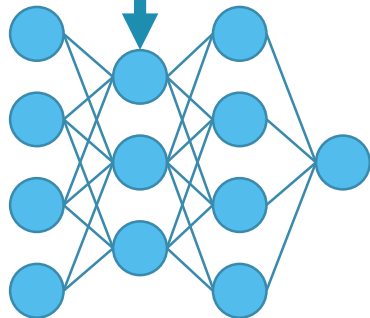
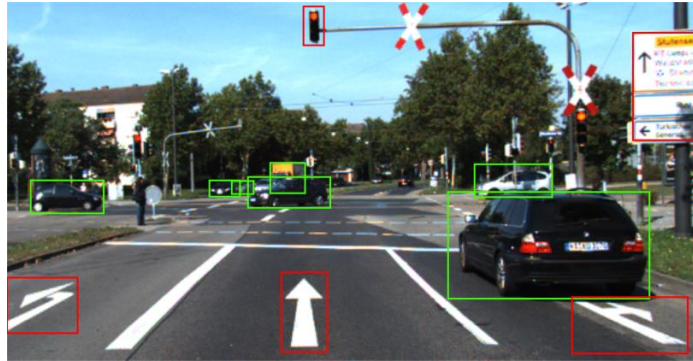


- Classifier defines a **probability distribution** over its output
- **Uncertainty** in the prediction
- Neural predicate: output = **probabilistic** choices in program
- No changes needed in the ProbLog inference or its semantics
- ProbLog can natively calculate the gradient

Perception

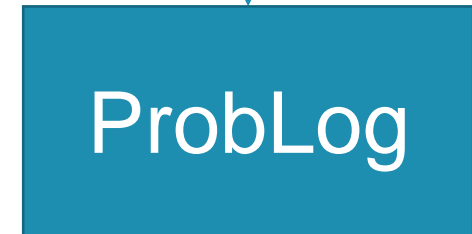


Perception

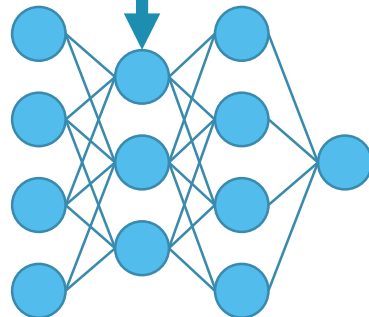
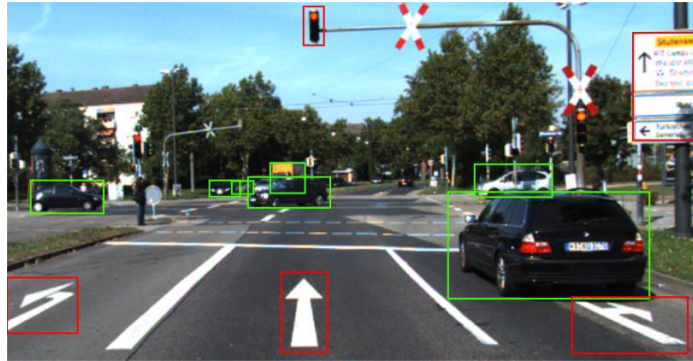


Reasoning

Stop in front of a red light
Obey the speed limit
Be in the correct lane
 $P(\text{light} = \text{red}) = 0.9$



Perception



$P(\text{obj1} = \text{car}) = 0.8$
 $P(\text{obj1 turn right}) = 0.7$

Neural predicate

Reasoning

Stop in front of a red light
Obey the speed limit
Be in the correct lane
 $P(\text{light} = \text{red}) = 0.9$

ProbLog

Related work

Related work	DeepProbLog
Logic is made less expressive	Full expressivity is retained
Logic is pushed into the neural part	Clean separation
Fuzzy logic	Probabilistic logic
Language semantics is unclear	Clear semantics

- Neural-symbolic integration (Garcez)
- Logical constraints as a regularizer (Xu, Diligenti, ...)
- Differentiable logical framework (Rocktäschel and Riedel, Evans and Grefenstette)
- Differentiable interpreters (Graves, Bosnjak)
- ...

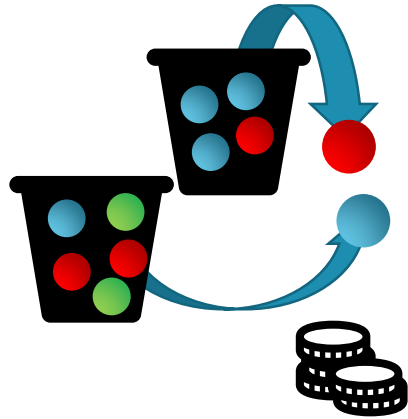
Example task: MNIST addition

The image shows a mathematical addition problem using handwritten digits from the MNIST dataset. The digits are displayed in black boxes on a white background. The first number is 35041, followed by a plus sign, the second number 921, an equals sign, and a question mark. The digits are: 3, 5, 0, 4, 1, +, 9, 2, 1, =, ?

- Only labeled sums, not single digits
- Train using only neural networks? Not suited!
- DeepProbLog can solve this:
 - Neural predicate
 - From pixels to distribution over digits
 - NN trained from scratch
 - Logic:
 - Combine predictions into larger numbers
 - Perform addition

Combined reasoning

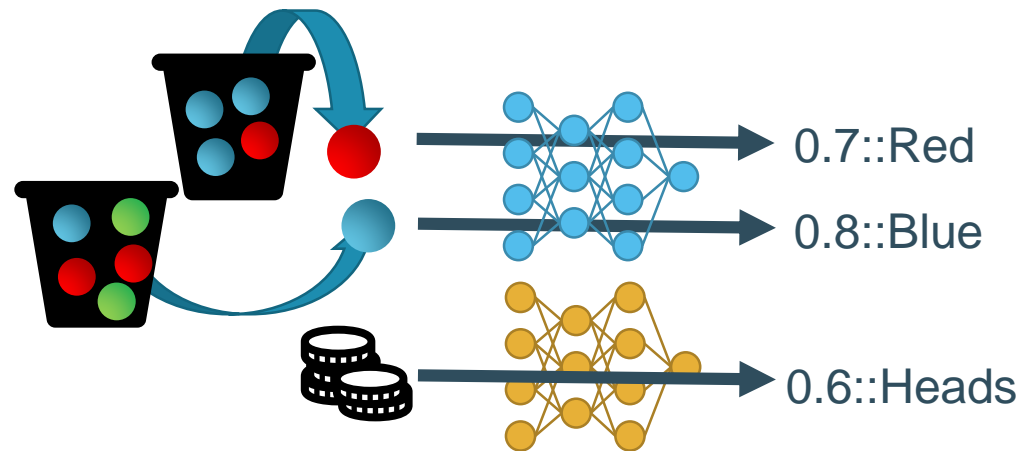
Unknown distribution



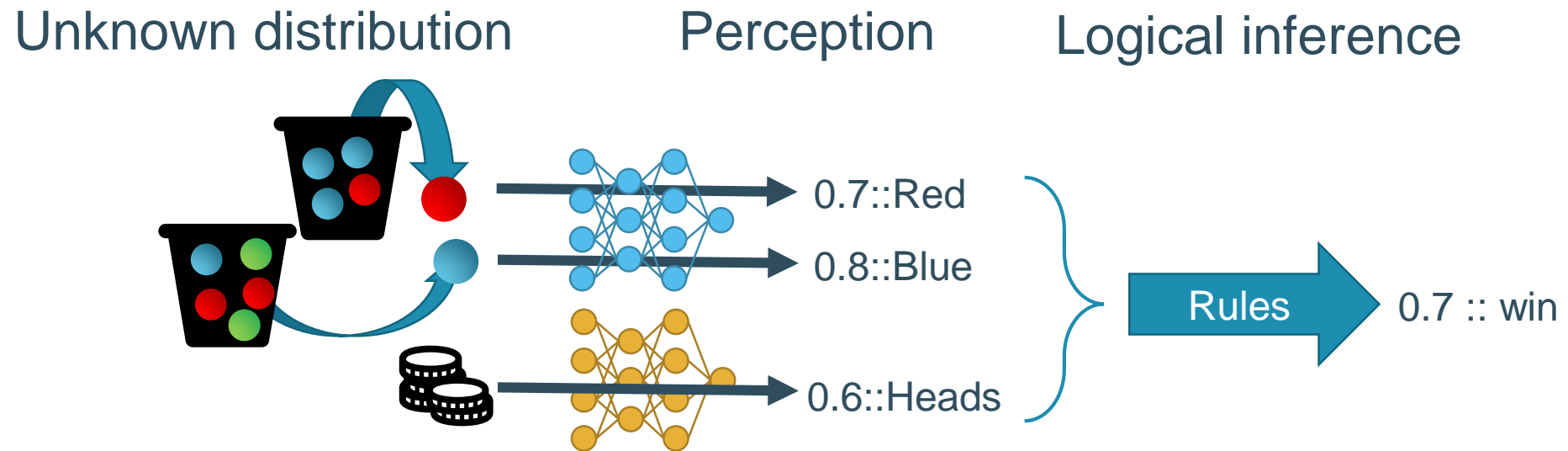
Combined reasoning

Unknown distribution

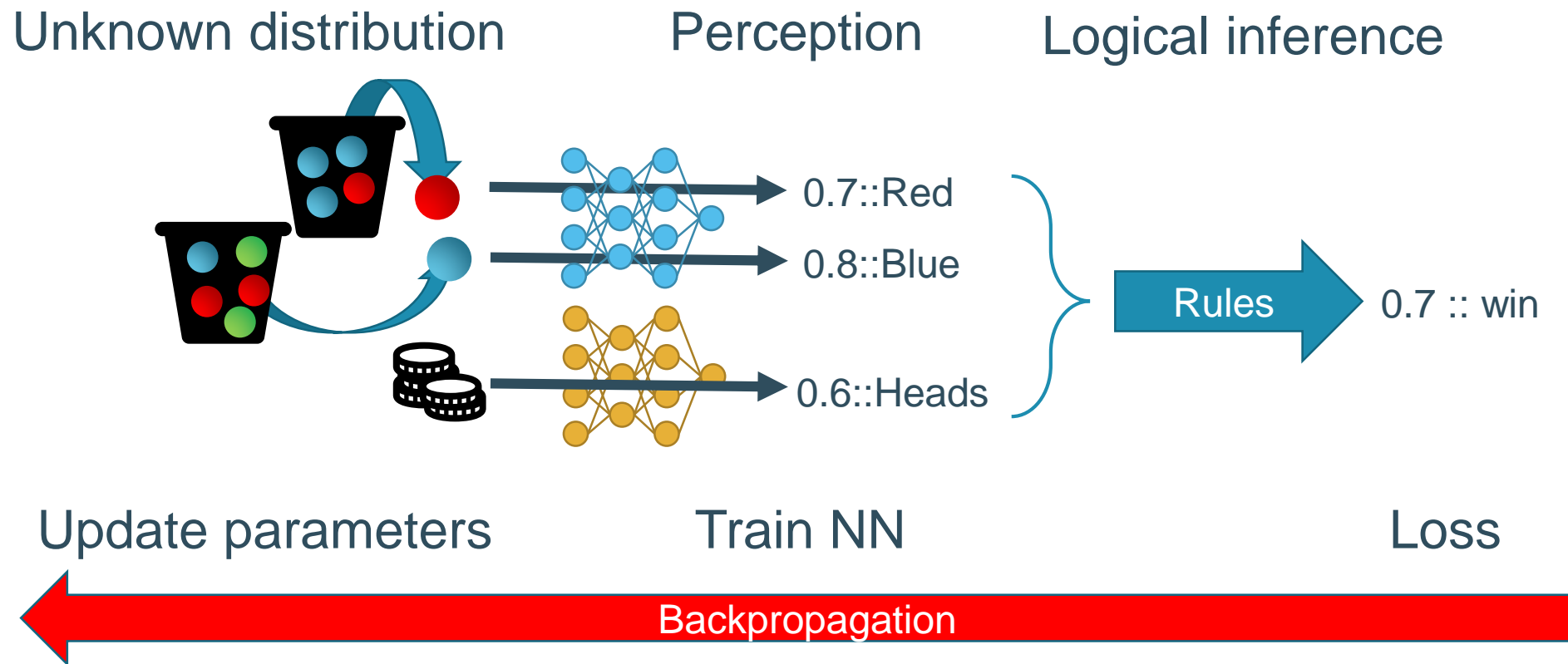
Perception



Combined reasoning



Combined reasoning



Conclusion

DeepProbLog: Neural Probabilistic Logic Programming

- Integration of DL and PLP
- Probabilistic
- Clean semantics, clear separation
- Retain power of both worlds
- Power of ProbLog

Code is available at:

<https://bitbucket.org/problog/deepproblog>

Poster #118

KU LEUVEN
DEPARTMENT OF COMPUTER SCIENCE

DeepProbLog: Neural Probabilistic Logic Programming

Robin Manhaeve¹, Sebastijan Dumančić², Angelika Kimmig³, Thomas Demeester³, Luc De Raedt¹
¹KU Leuven, ²Cardiff University, ³Ghent University, imec, ⁴Joint last authors
 rs robin.manhaeve@cs.kuleuven.be

Integrating reasoning and perception

- Integrating low-level perception with high-level reasoning is one of the oldest, and yet most active open challenges in AI.
- Low-level perception is typically handled by deep learning.
- High-level reasoning is typically addressed using (probabilistic) logical representations and inference.
- Joining the full flexibility of high-level probabilistic reasoning with the representational power of deep neural networks is still an open problem.

Our approach

- Instead of integrating reasoning capabilities into a complex neural network architecture, we proceed the other way round.
- DeepProbLog is a probabilistic logic programming language incorporating deep learning
 - It contains expressive probabilistic-logical modeling and reasoning
 - It encapsulates general-purpose neural networks
 - It can be trained end-to-end on examples
- Our approach has:
 - The expressiveness and strengths of both worlds
 - A clean separation between both sides
 - A clear semantics

Learning in DeepProbLog

- aProbLog: ProbLog + Semirings
- Gradient semiring: elements of the form $(p, \frac{\partial p}{\partial \theta})$

$$(a_1, \dots, a_N) \otimes (b_1, \dots, b_N) = (a_1 + b_1, \dots, a_N + b_N) \quad \text{for } p := f \text{ with fixed } \theta$$

$$(a_1, \dots, a_N) \otimes (b_1, \dots, b_N) = (a_1 b_1, \dots, a_N b_N) \quad \text{for } f(p) := f_1 \text{ with learnable } \theta$$

$$\mathcal{L}(\neg f) = (1 - p, -\nabla p) \quad \text{with } \mathcal{L}(f) = (p, \nabla p)$$

```

flip(coin), flip(coin),
write_atom(C,beads,init): write(C,1),
(L,0):write(L,0):write,
beads: flip(1), write(L,beads),
write - beads, read,
query(write).
                    
```

Experimental evaluation

- T1: addition (5, 7, 8): Classify the sum of pair of MNIST digits
- T2: addition (17, 21, 12, 1, 63): Classify the sum of two multi-digit numbers.

- T3, T4, T5: Learning to perform addition, sorting and WAP problems[1]

Task Length	Sorting (T4): Training length						Addition (T5): Training length					
	2	3	4	5	6	7	2	3	4	5	6	
400	8	100	100	100	100	100	8	100	100	100	100	
DeepProbLog	8	100	100	100	100	100	8	100	100	100	100	

Training length on T4: 2 3 4 5 6
 0% on CPU: 42 s 150 s - - -
 0% on GPU: 61 s 209 s - - -
 DeepProbLog on CPU: 11 s 14 s 32 s 114 s 265 s

T5: Accuracy = 96 - 97%

- T6: A task in which we model a probabilistic game in which:
 - Neural networks are trained from scratch to recognize colours and heads / tails
 - The input is coins (MNIST images), noisy RGB triplets and the outcome
 - Probabilistic parameters need to be trained jointly with the neural networks

We show that DeepProbLog can achieve all this jointly, achieving 100% accuracy.

DeepProbLog

- Prolog: ProbLog + Probabilities
- (Probabilistic) facts:


```

0.1 : burglary. 0.2 : earthquake. 0.5 : alarm_alarm(sary). 0.4 : alarm_alarm(john).
                    
```
- Rules:


```

alarm :- earthquake. alarm :- burglary. calls :- alarm, alarm(X).
                    
```
- DeepProbLog: ProbLog + Neural predicates
- Neural Annotated Disjunction (nAD):

$$nn(m_1, \dots, m_n) :: q(\vec{E}, u_1), \dots, q(\vec{E}, u_n) :- b_1, \dots, b_n.$$
- Evaluates a neural network m_i on input \vec{E}
- It defines a probability distribution over u_1, \dots, u_n .

Example

- Classify the sum of pairs of MNIST images, e.g. 5 + 7 = 8
- Encode the background knowledge of the sum.
 - Define the neural predicate for classifying the digits.


```

nn(m_digit_X, [0, ..., 9]) :: digit(X, 0); ...; digit(X, 9).
                    
```
 - Define the addition.


```

addition(X, Y, Z) :- digit(X, X2), digit(Y, Y2), Z is X2 + Y2.
                    
```
- The neural predicate evaluates the neural network


```

... ; 0.2 :: digit(5, 2) ; 0.8 :: digit(5, 3) ; ...
... ; 0.7 :: digit(5, 5) ; 0.3 :: digit(5, 6) ; ...
                    
```
- query(addition(5, 7, X)).


```

addition(5, 7, 12) : 0.14
addition(5, 7, 8) : 0.62
addition(5, 7, 9) : 0.24
                    
```

KU Leuven
Department of Computer Science
Celestijnenlaan 200A box 2402
3001 Leuven, Belgium

DTAI
Department of Data Science and Artificial Intelligence

Cardiff University
PROBLOG
CARDIFF

Ghent University

fwo

DTAI reserach group

KU LEUVEN