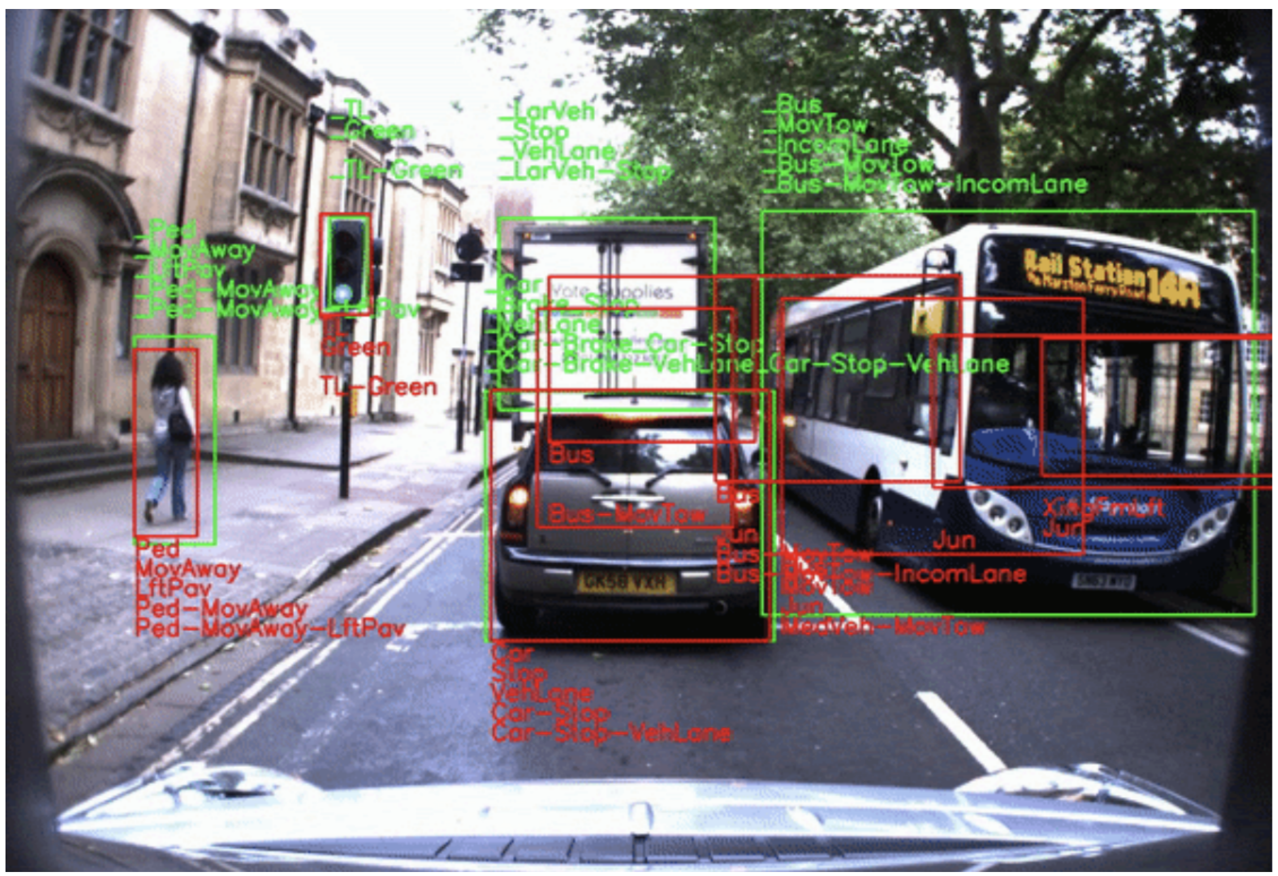
How to make logics neurosymbolic

Luc De Raedt

Giuseppe Marra & Vincent Derkinderen & Sebastijan Dumancic & Robin Manhaeve & Thomas Winters & Angelika Kimmig & Jaron Maene

AI MODEL = DATA + KNOWLEDGE



Natural Language Explanations
If an agent pushes an object then it is a pedestrian
A pedestrian can only push objects, move away, etc.
Only pedestrains, cars, cyclists, etc. can cross from left
Only pedestrians and cyclists can wait to cross
Only pedestrians, cars, cyclists, etc can stop
Only pedestrians, cars, cyclists, etc can move
Only pedestrians, cars, cyclists, etc can move towards
Only pedestrians, cars, cyclists, etc can move away
An emergency vehicle can only overtake, move away etc.
Only emergency vehicles, cars etc. can have hazards lights on
A bus can only overtake, move away move towards etc.
A medium vehicle can only overtake, move away, move towards etc.

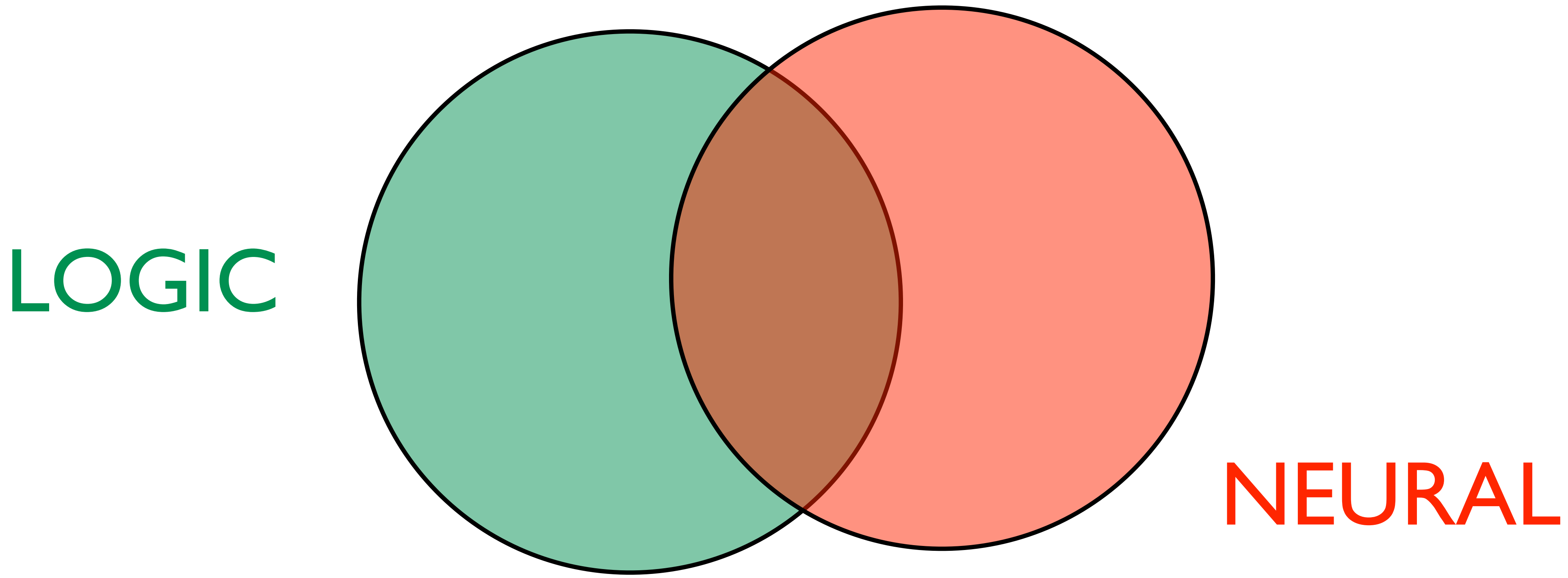
Giunchiglia, Eleonora, Mihaela Cătălina Stoian, Salman Khan, Fabio Cuzzolin, and Thomas Lukasiewicz. "ROAD-R: The autonomous driving dataset with logical requirements." *Machine Learning* (2023): 1-31.

The NeuroSymbolic alphabet-soup

Neural PSL
A-NeSI
NTP
SBR
DiffLog
NLog
LTN
SL
αILP
NLProlog
Neural MLNs
Scallop
Slash
δILP
RNM
DL2
NeurASP
TensorLog
DeepSeaLog
DeepProbLog
Neural ILP
SBR
LRNNs
DeepStochLog

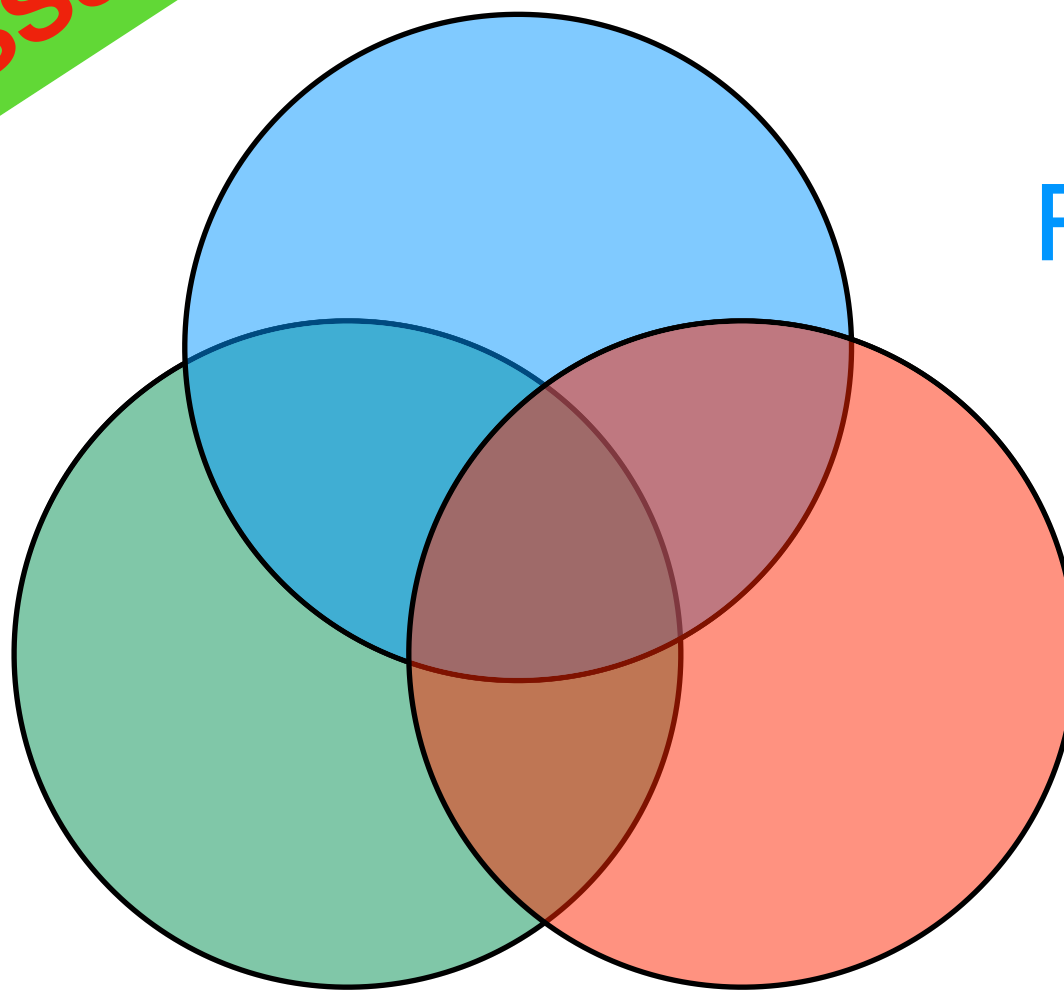
check our survey on AIJ — Marra, Dumancic, Manhaeve & De Raedt, 23

Neurosymbolic =
Neuro + Logic



Neurosymbolic =
Neuro + Logic + Probability

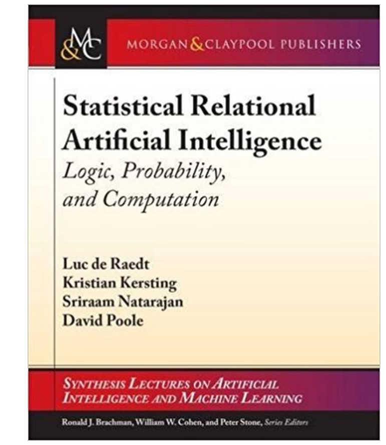
Key Message 1

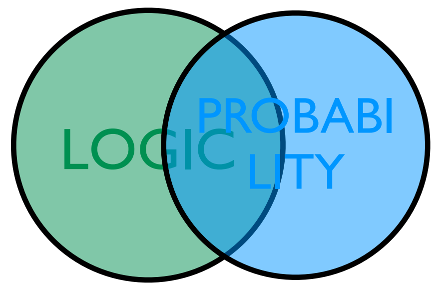


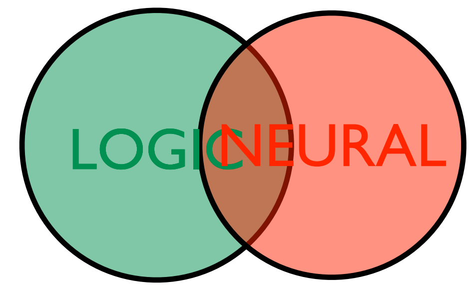
see Manhaeve et al. NeSy Book

interpret PROBABILITY broadly (including fuzzy)

Key Message 2







StarAI and NeSy share similar problems and thus similar solutions apply

See also [De Raedt et al., IJCAI 20; Marra et al, AIJ 24]

Key Message 3

Provide recipe for

Kautz

Neural : : Symbolic

Key Message 3

Provide recipe for

Kautz

Neural : : Symbolic

“an interface layer (<> pipeline) between neural & symbolic components”

Part 1: NeSy AI - a little Survey

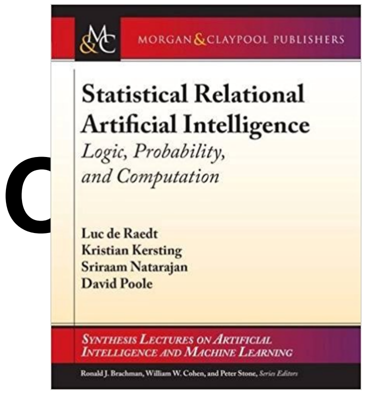
Part 2: The Recipe

Part 3: DeepStochLog and DeepProbLog

Part 1: NeSy AI - a little survey

check our survey on AIJ — Marra, Dumancic, Manhaeve & De Raedt, 23

Two types of probabilistic graphic models and StarAI systems



0.1 :: burglary.

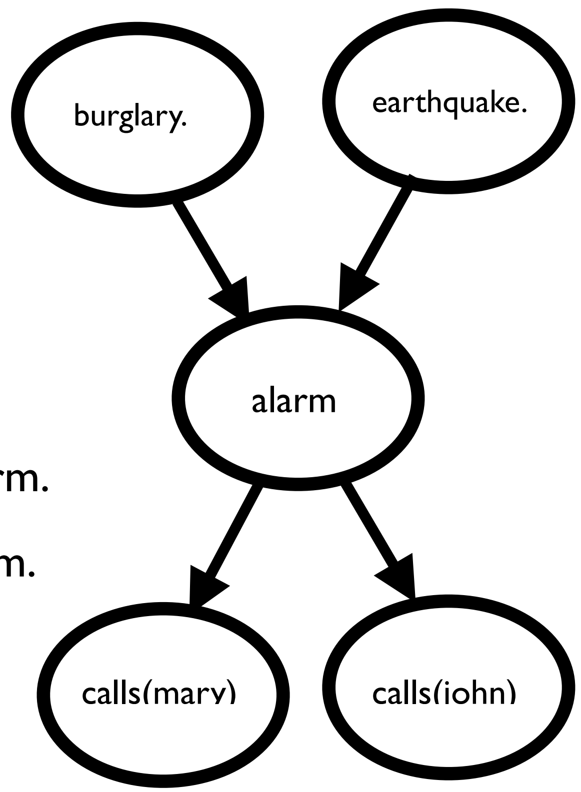
0.05 :: earthquake.

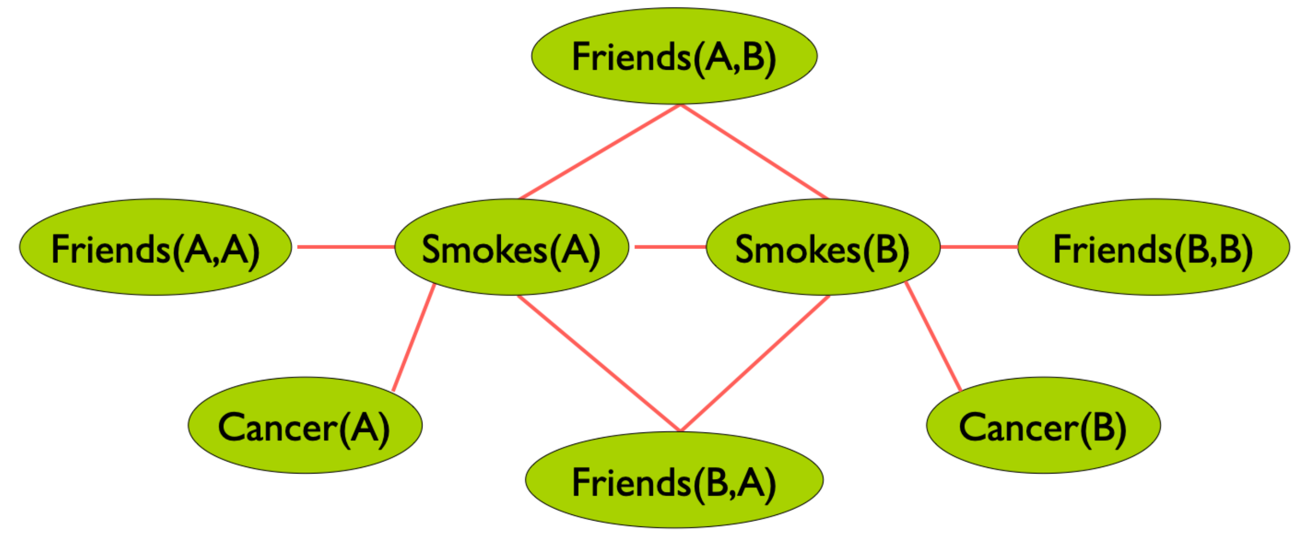
alarm :– earthquake.

alarm :– burglary.

0.7::calls(mary) :– alarm.

0.6::calls(john) :– alarm.





$$1.5 \quad \forall x\ Smokes(x) \Rightarrow Cancer(x)$$

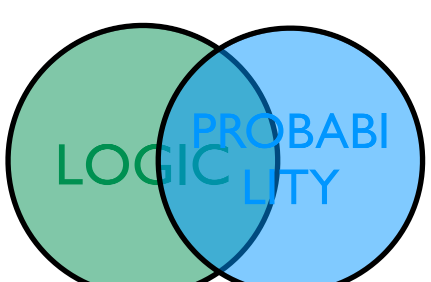
$$1.1 \quad \forall x, y\ Friends(x, y) \Rightarrow \big(Smokes(x) \Leftrightarrow Smokes(y)\big)$$

Probabilistic Logic Programs
ProbLog

directed
Bayesian Net

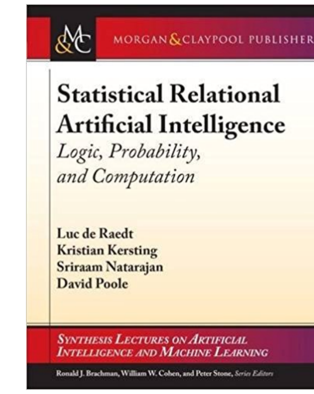
Markov Logic

undirected
Markov Net
model theoretic



key representatives

Two types of Neural Symbolic Systems



Just like in StarAI

Logic as a kind of ***neural program***

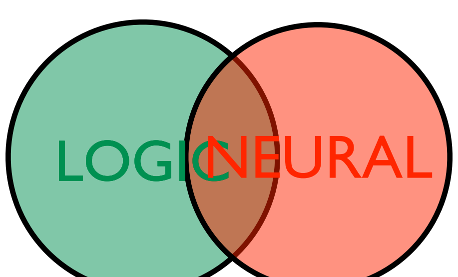
directed StarAI approach and logic programs

Logic as the ***regularizer*** ***(reminiscent of Markov Logic Networks)***

undirected StarAI approach and (soft) constraints

Also, many NeSy systems are doing
knowledge based model construction KBMC
where logic is used as a template

Just like in StarAI



Logic as a neural program

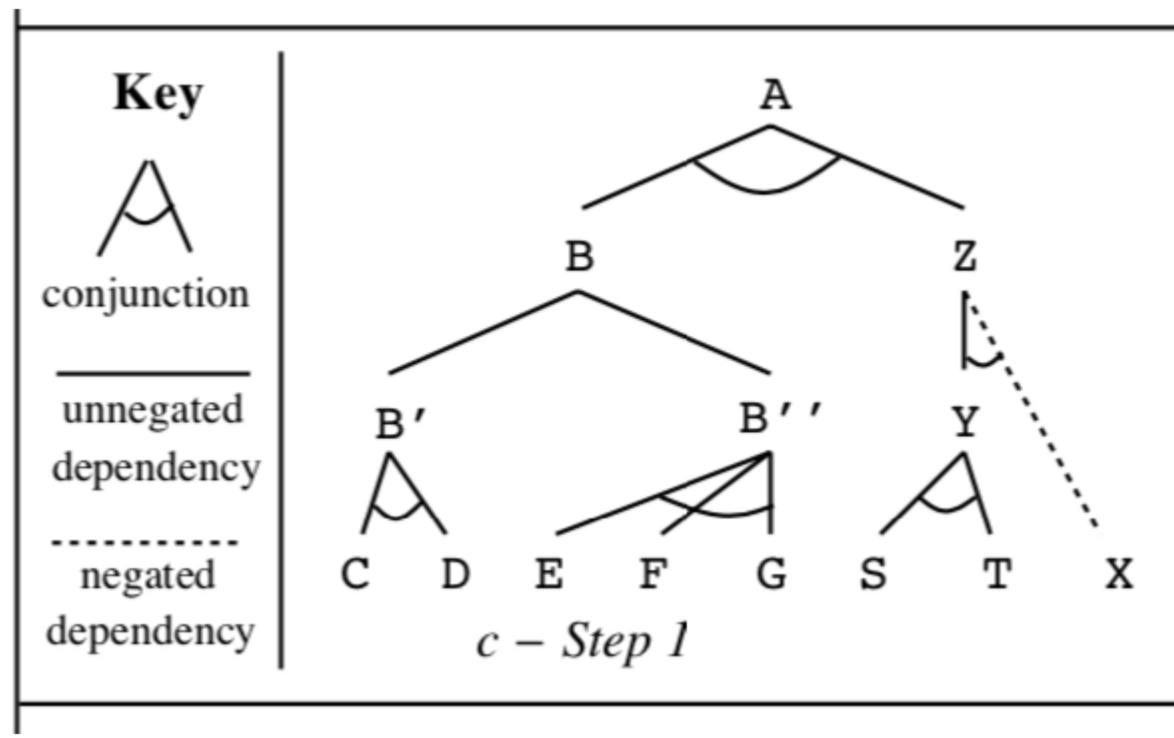
directed StarAI approach and logic programs

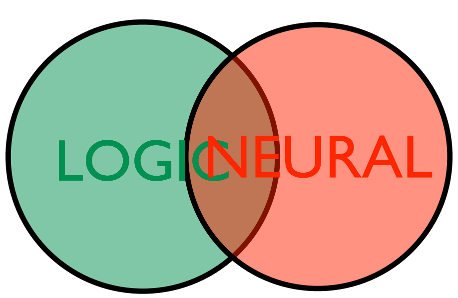
- KBANN (Towell and Shavlik AIJ 94)
- Turn a (propositional) Prolog program into a neural network and learn

```
A :- B, Z.
B :- C, D.
B :- E, F, G.
Z :- Y, not X.
Y :- S, T.
```

REWRITE

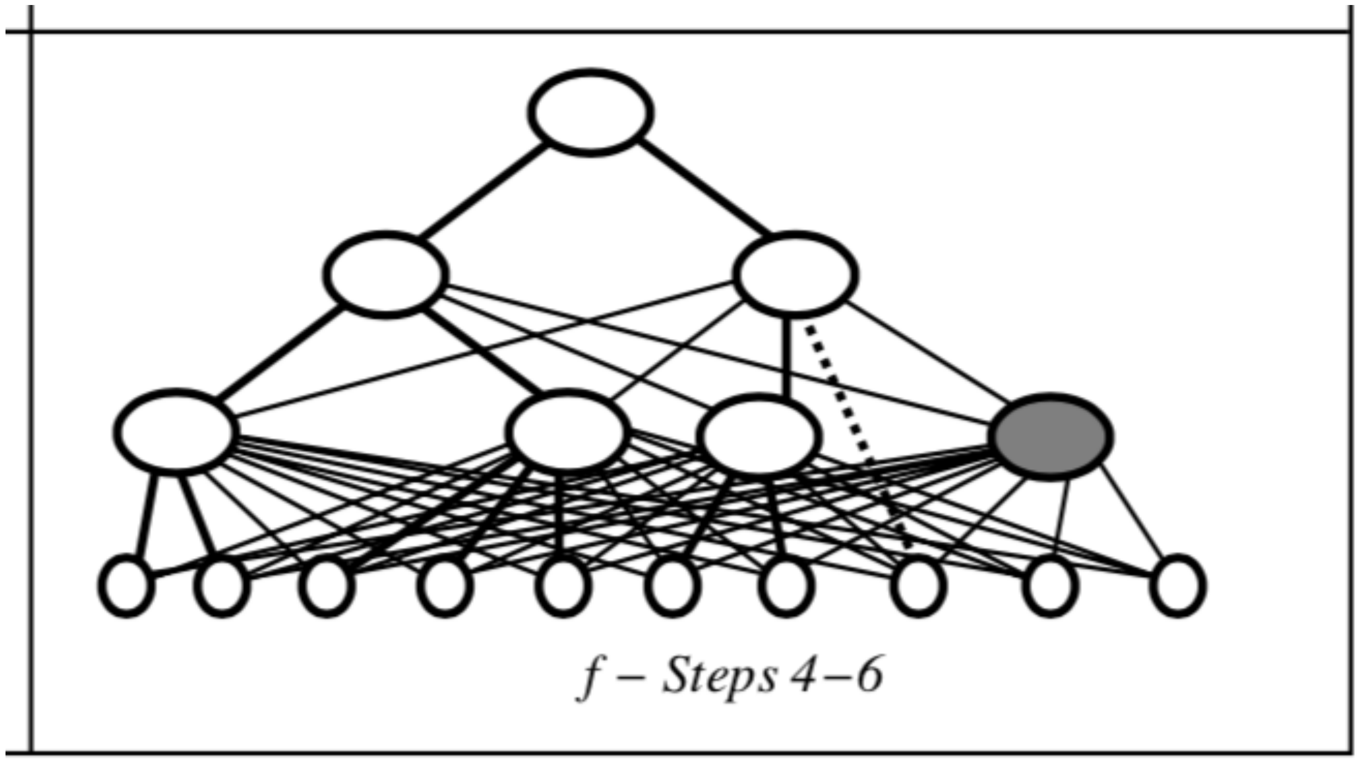
```
A    :- B, Z.
B    :- B'.
B    :- B''.
B'   :- C, D.
B''  :- E, F, G.
Z    :- Y, not X.
Y    :- S, T.
```

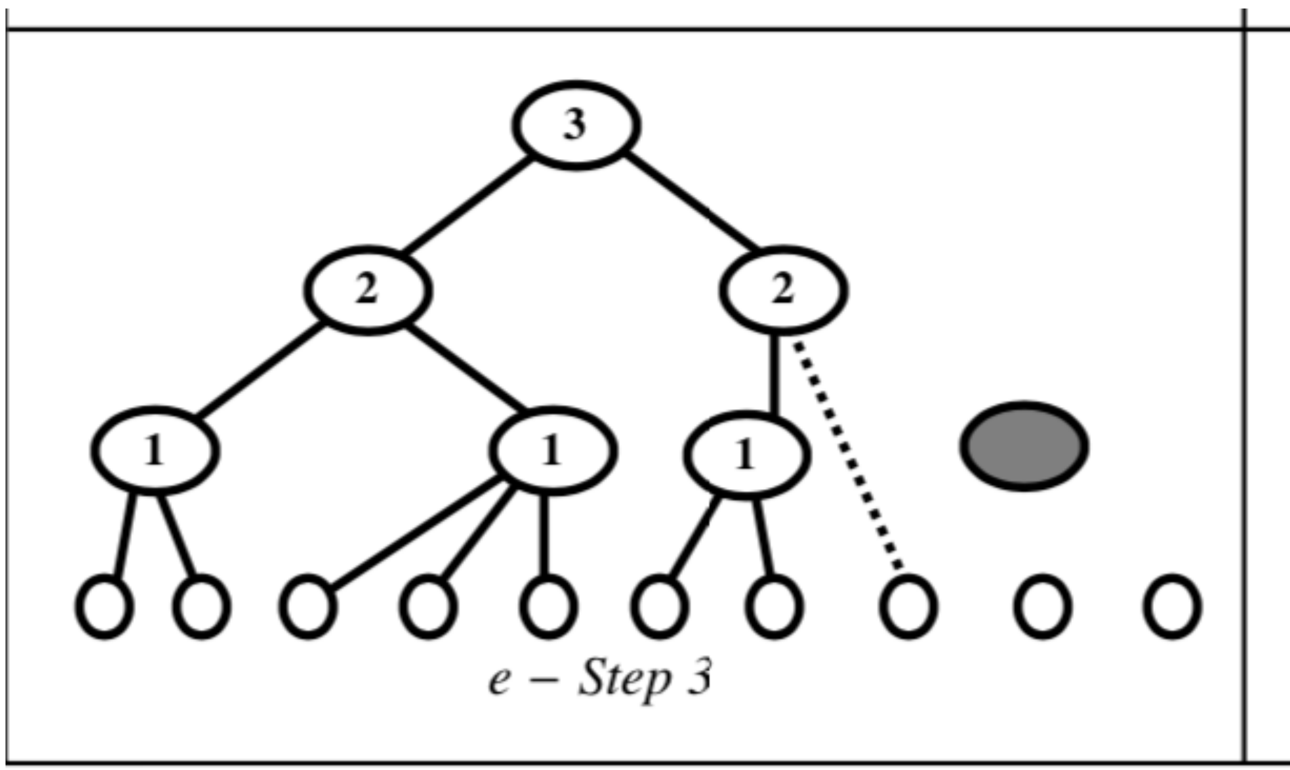




Logic as a neural program

directed StarAI approach and logic programs



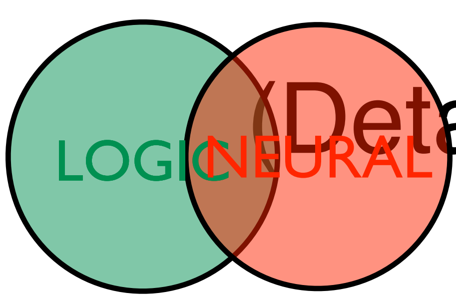


ADD LINKS — ALSO SPURIOUS ONES

HIDDEN UNIT

and then learn

(Details of activation & loss functions not mentioned)



directed StarAI approach and logic programs

Neural Theorem Prover

Towards Neural Theorem Proving at Scale

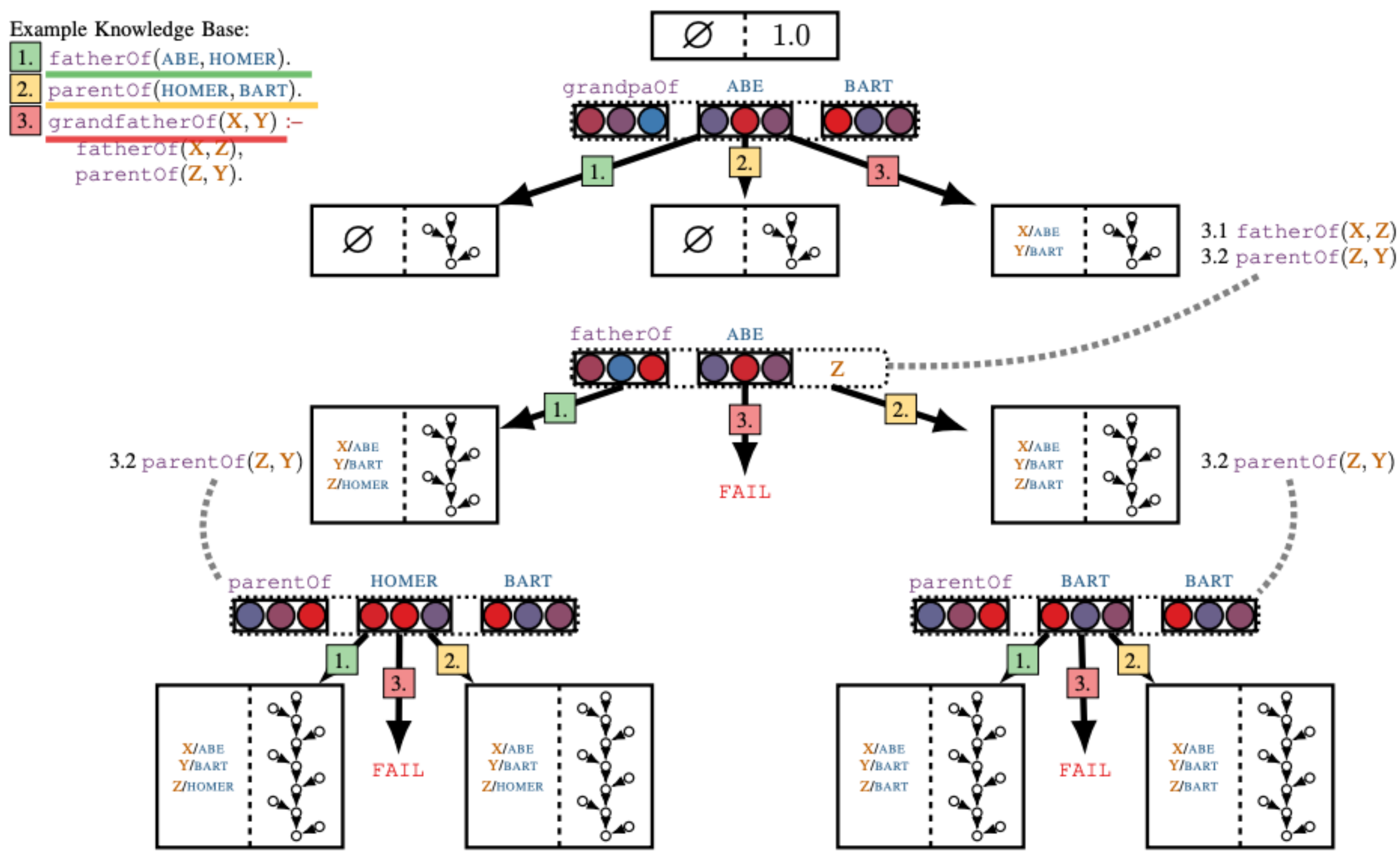
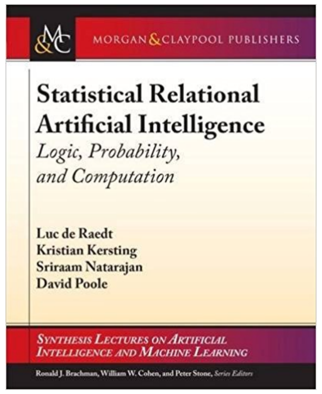


Figure 1. A visual depiction of the NTP' recursive computation graph construction, applied to a toy KB (top left). Dash-separated rectangles denote proof states (left: substitutions, right: proof score -generating neural network). All the non-FAIL proof states are aggregated to obtain the final proof success (depicted in Figure 2). Colours and indices on arrows correspond to the respective KB rule application.

LOGIC NEURAL

[Rocktäschel Riedel, NeurIPS 17; Minervini et al.]

Two types of Neural Symbolic Systems



Just like in StarAI

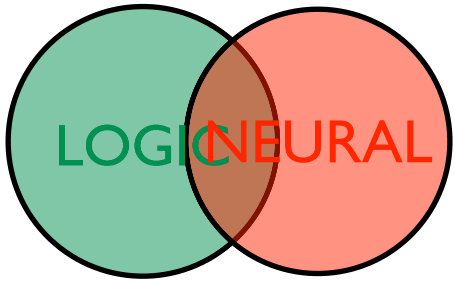
Logic as a kind of ***neural program***

directed StarAI approach and logic programs

Logic as the ***regularizer*** ***(reminiscent of Markov Logic Networks)***

undirected StarAI approach and (soft) constraints

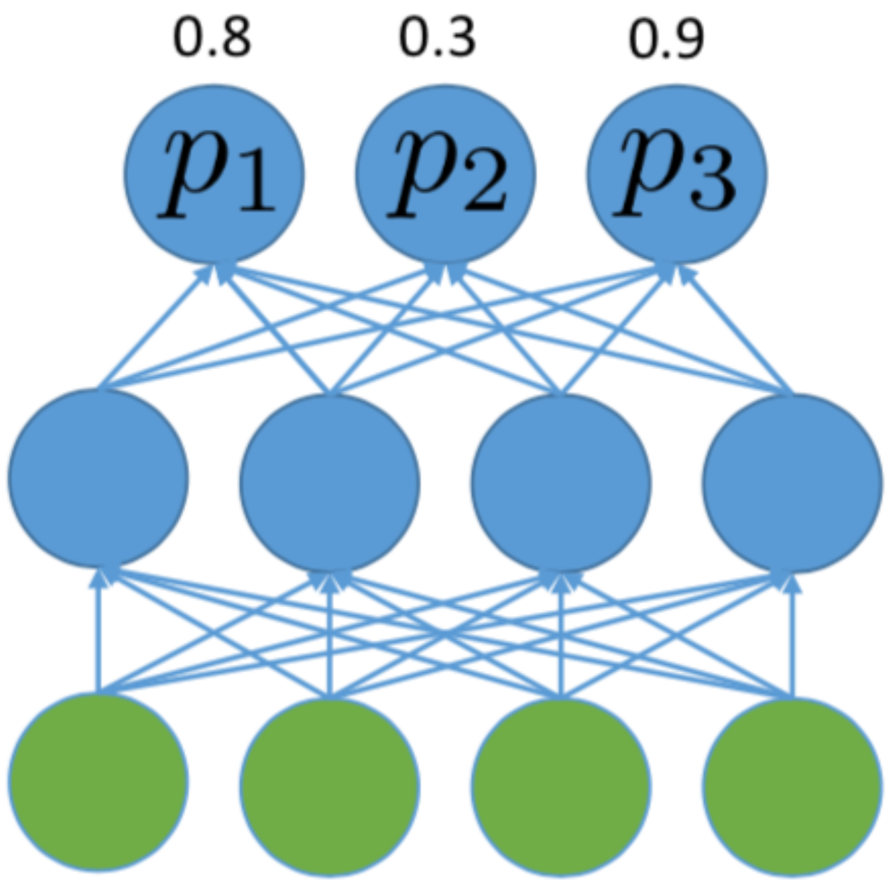
Also, many NeSy systems are doing *knowledge based model construction KBMC where logic is used as a template*



Logic as constraints

undirected StarAI approach and (soft) constraints

multi-class classification

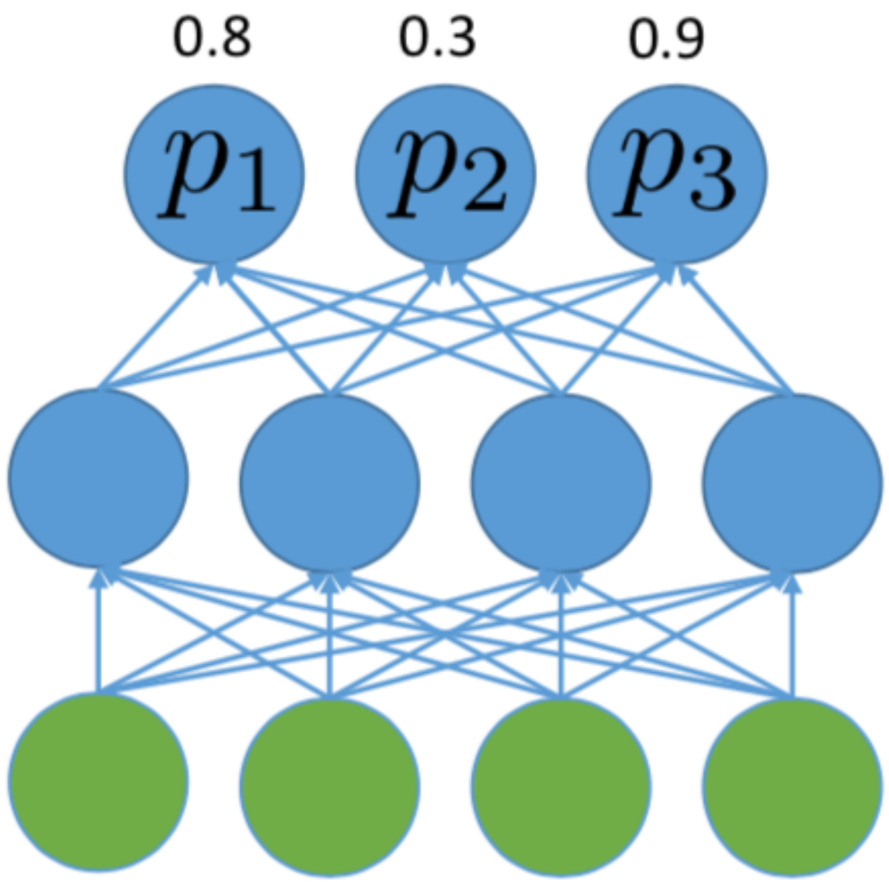


figures and example from Xu et al., ICML 2018

Logic as constraints

undirected StarAI approach and (soft) constraints

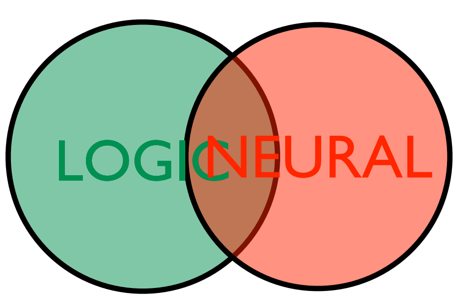
multi-class classification



This constraint should be satisfied

$$
\begin{aligned}
&(\neg x_1 \wedge \neg x_2 \wedge x_3) \vee \\
&(\neg x_1 \wedge x_2 \wedge \neg x_3) \vee \\
&(x_1 \wedge \neg x_2 \wedge \neg x_3)
\end{aligned}
$$

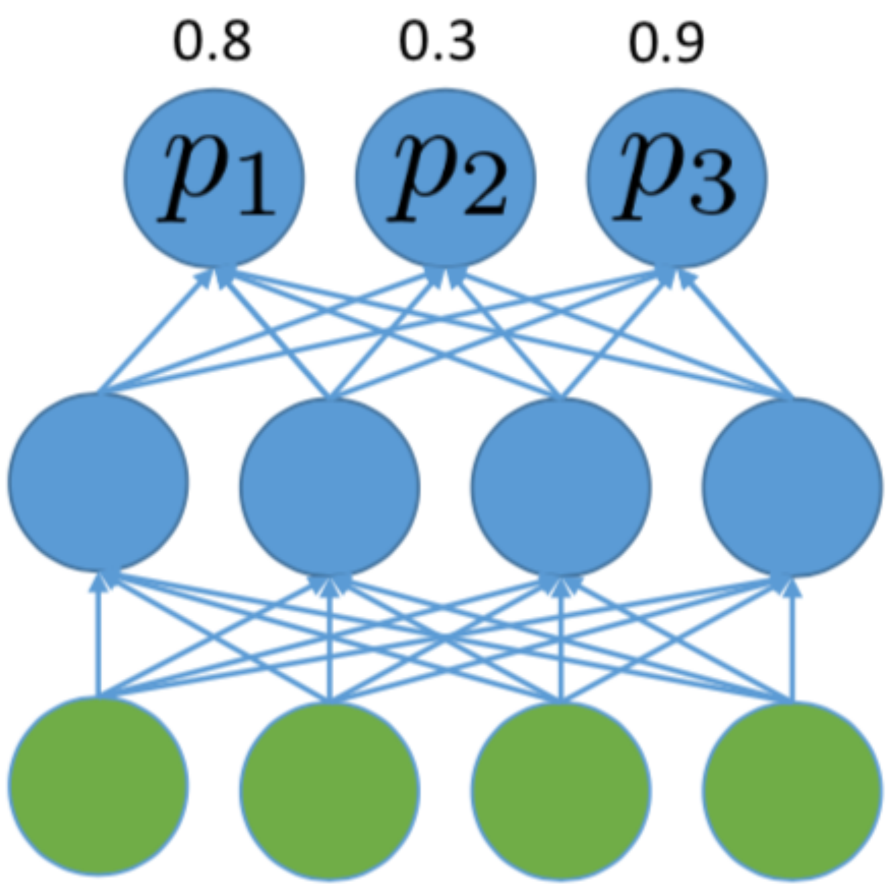
figures and example from Xu et al., ICML 2018



Logic as constraints

undirected StarAI approach and (soft) constraints

multi-class classification

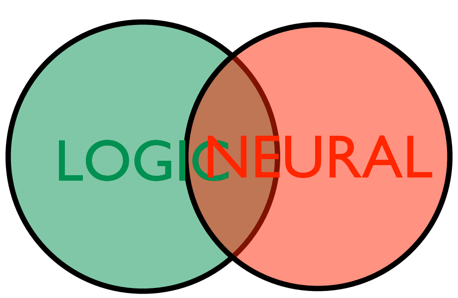


Probability that constraint is satisfied

$$(1-x_1)(1-x_2)x_3+$$
$$(1-x_1)x_2(1-x_3)+$$
$$x_1(1-x_2)(1-x_3)$$

basis for SEMANTIC LOSS

(weighted model counting)



Logic as a regularizer

undirected StarAI approach and (soft) constraints

Semantic Loss:

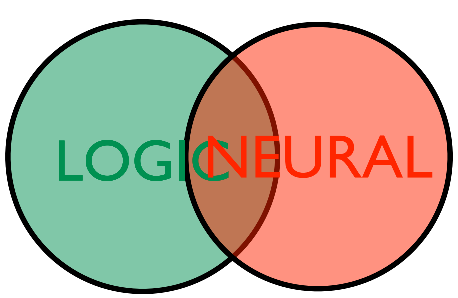
- Use logic as constraints (very much like "propositional MLNs)
- Semantic loss

$$SLoss(T) \propto -\log \sum_{X \models T} \prod_{x \in X} p_i \prod_{\neg x \in X} (1 - p_i)$$

- Used as regulariser

$$Loss = TraditionalLoss + w.SLoss$$

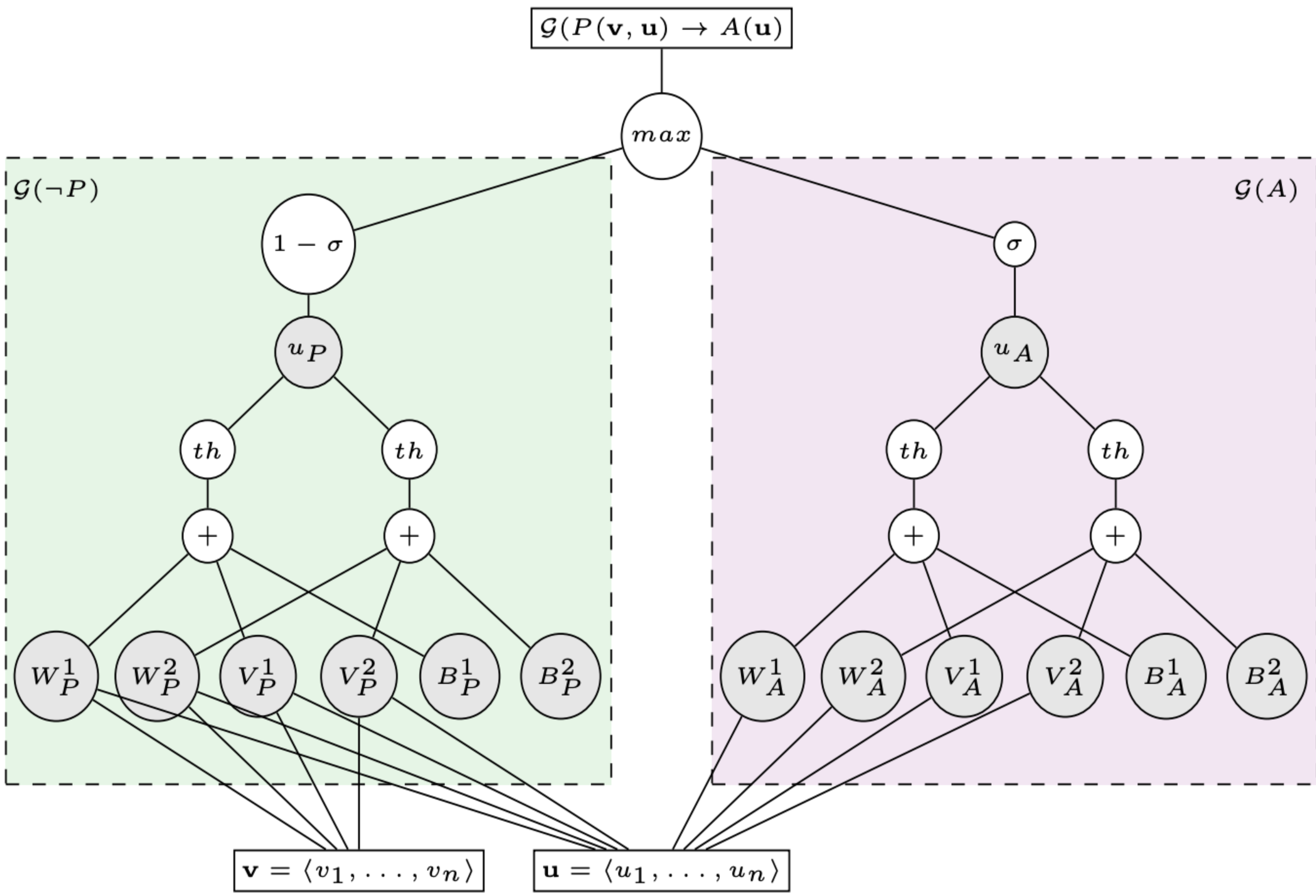
- Use weighted model counting , close to StarAI

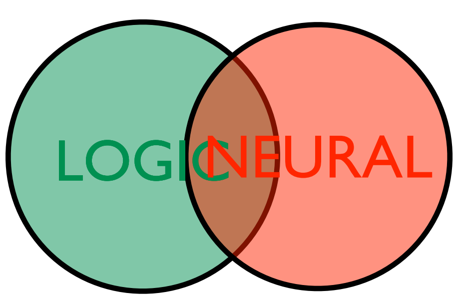


Logic Tensor Networks

undirected StarAI approach and (soft) constraints

$$P(x, y) \to A(y), \text{ with } \mathcal{G}(x) = \mathbf{v} \text{ and } \mathcal{G}(y) = \mathbf{u}$$





Serafini & Garcez

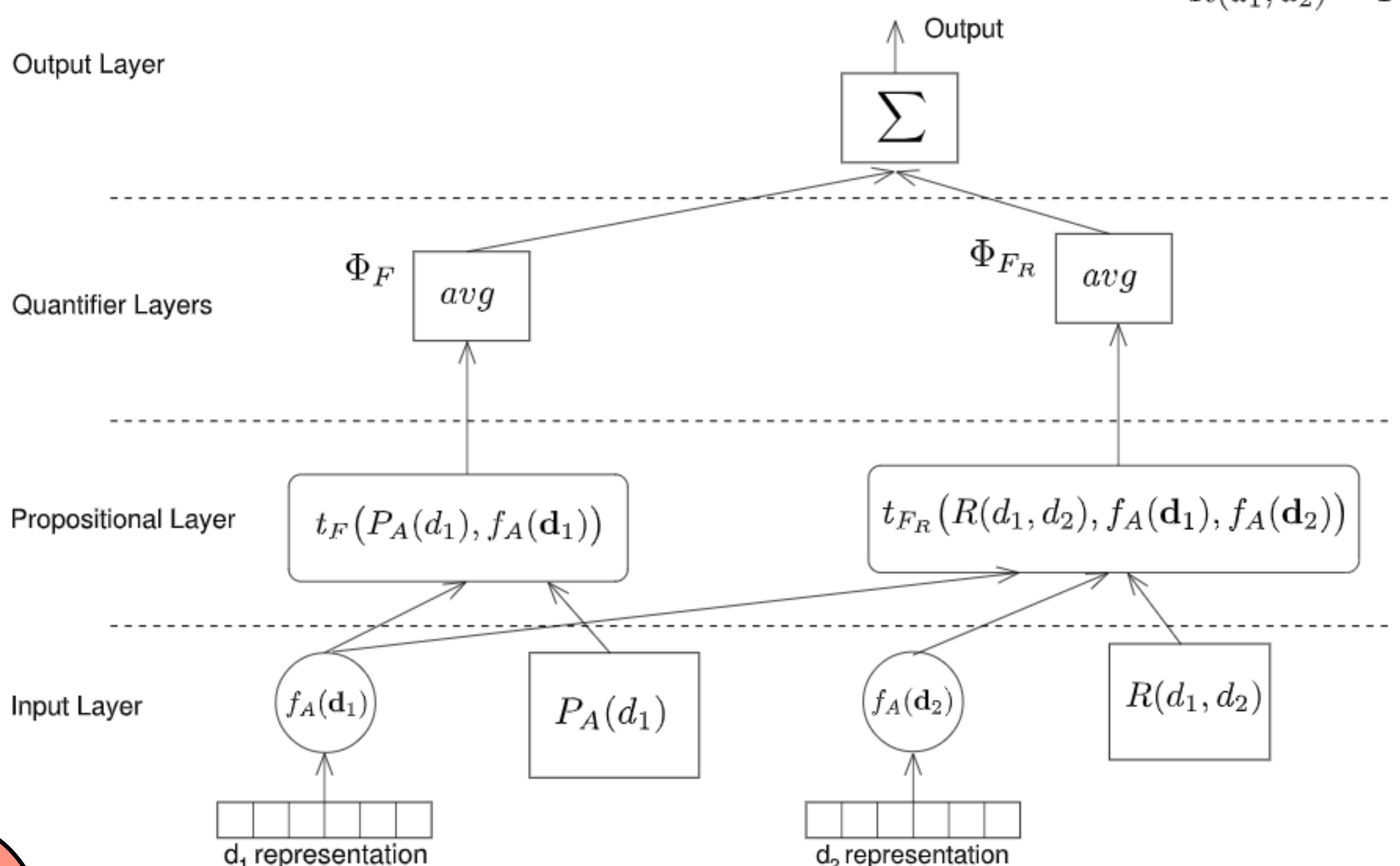
Semantic Based Regularization

undirected StarAI approach and (soft) constraints

$$
\begin{aligned}
F &:= \forall d\ P_A(d) \Rightarrow A(d) \\
F_R &:= \forall d\ \forall d'\ R(d,d') \Rightarrow \big((A(d) \wedge A(d')) \vee (\neg A(d) \wedge \neg A(d'))\big) \\
C &= \{d_1, d_2\}
\end{aligned}
$$

Evidence Predicate Groundings

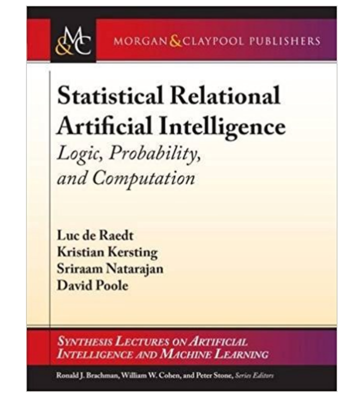
$$
\begin{aligned}
P_A(d_1) &= 1 \\
R(d_1, d_2) &= 1
\end{aligned}
$$



LOGIC NEURAL

Diligenti et al. AIJ

Two types of Neural Symbolic Systems



Just like in StarAI

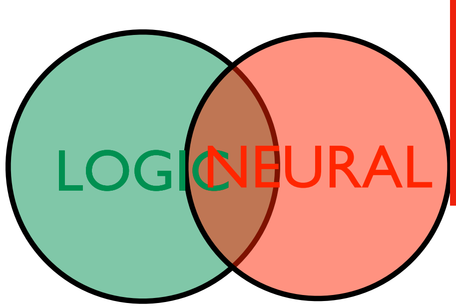
Logic as a kind of ***neural program***

directed StarAI approach and logic programs

Logic as the ***regularizer*** ***(reminiscent of Markov Logic Networks)***

undirected StarAI approach and (soft) constraints

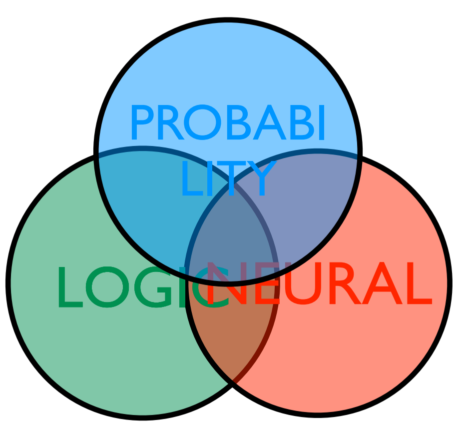
Consequence :
the logic is encoded in the network
the ability to logically reason is lost
logic is not a special case



A different approach

A true integration T of X and Y should allow to reconstruct X and Y as special cases of T

Thus, Neural Symbolic approaches should have both logic and neural networks as special cases



Part 3 of the talk — illustration with DeepProbLog [NeurIPS 2018] and DeepStochLog [AAAI 2022]

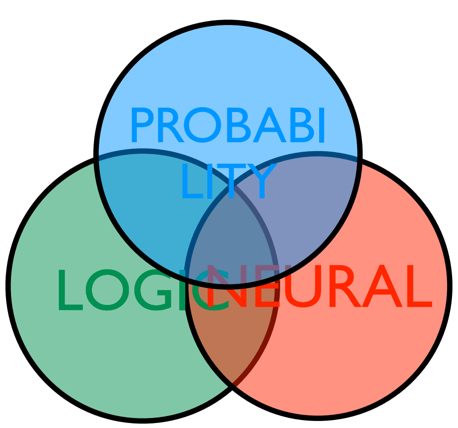
A different approach

A true integration T of X and Y should allow to reconstruct X and Y as special cases of T

Thus, Neural Symbolic approaches should have both logic and neural networks as special cases

Our approach: “an interface layer (<> pipeline) between neural & symbolic components” will be illustrated with DeepProbLog

See also [Manhaeve et al., NeurIPS 18; arXiv: 1907.08194]



Part 3 of the talk — illustration with DeepProbLog [NeurIPS 2018] and DeepStochLog [AAAI 2022]

Part 2: The Recipe

Turning any logic into a neurosymbolic one

check our survey on AIJ — Marra, Dumancic, Manhaeve & De Raedt, 24

Neurosymbolic functions

& primitives defining semantics and computational graph

Neural nets

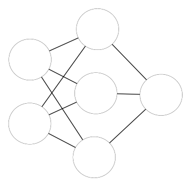
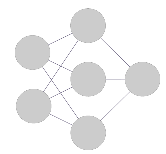
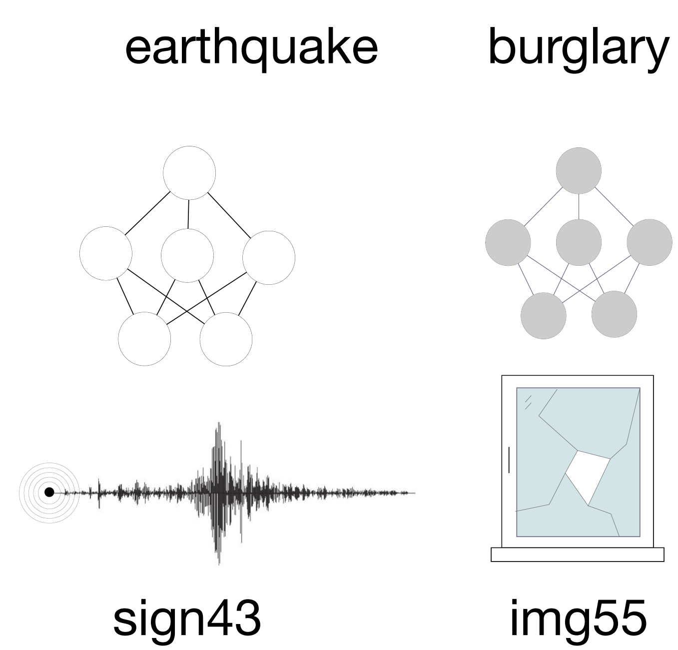
signal-
analysis 

image-
analysis 



Neurosymbolic functions

& primitives defining semantics and computational graph

Neural nets

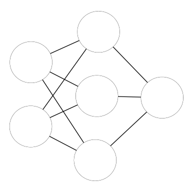
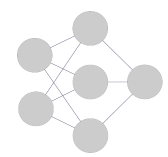
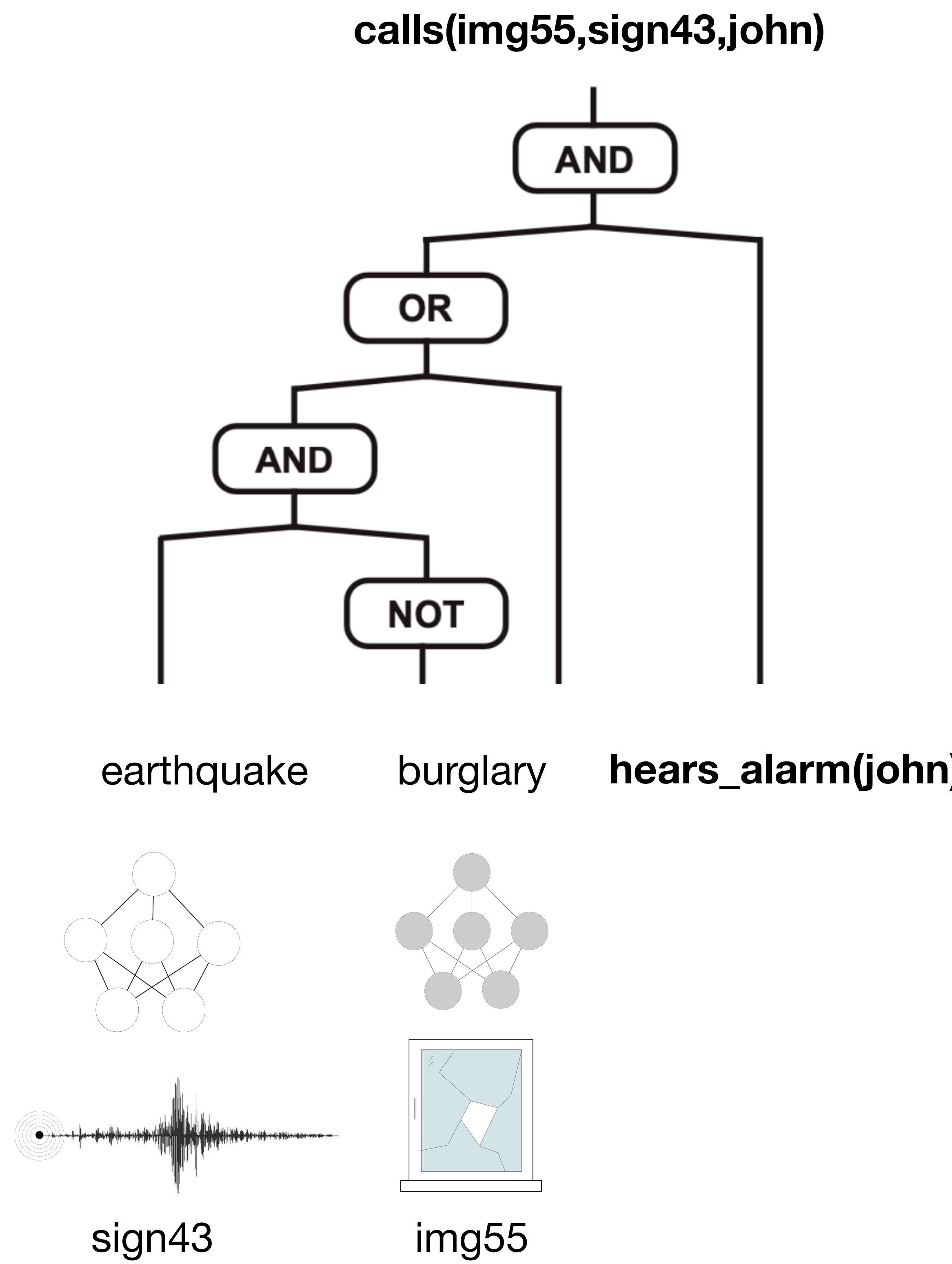
signal-
analysis 

image-
analysis 

Logic

```
alarm(B,E)    IF earthquake(E) OR burglary(B)
calls(B,E,P) IF alarm(B,E) AND hears_alarm(P)

hears_alarm(john)
hears_alarm(mary)
```



Neurosymbolic functions

Can we learn
calls END TO END ?

& primitives defining semantics and computational graph

Neural nets

signal-
analysis

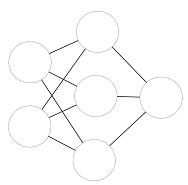
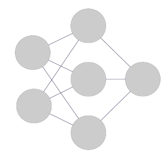


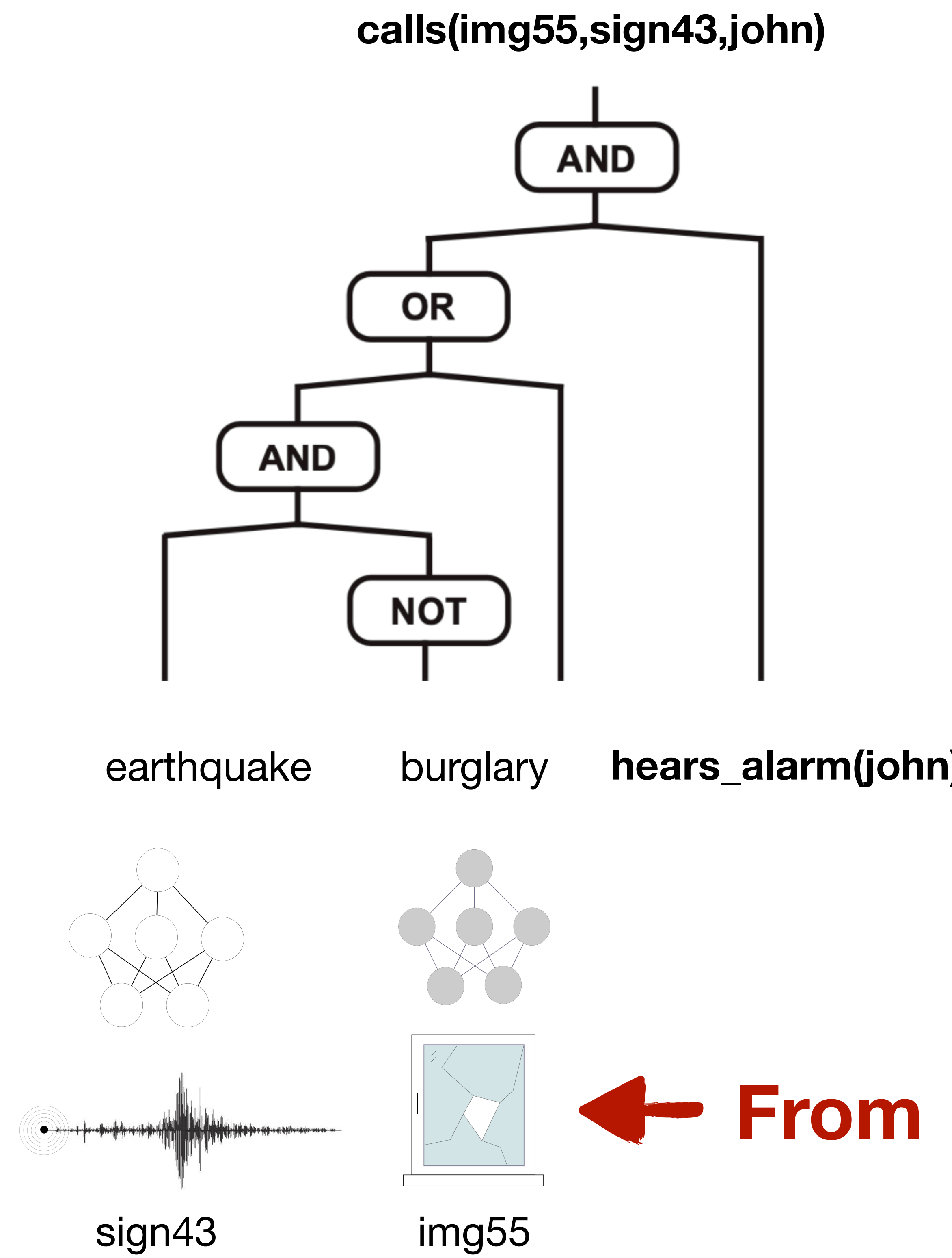
image-
analysis



Logic

```
alarm(B,E)    IF earthquake(E) OR burglary(B)
calls(B,E,P) IF alarm(B,E) AND hears_alarm(P)

hears_alarm(john)
hears_alarm(mary)
```



Neurosymbolic functions

& primitives defining semantics and computational graph

Neural nets

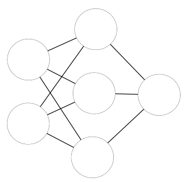
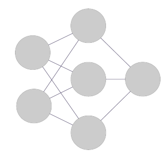
signal-analysis 

image-analysis 

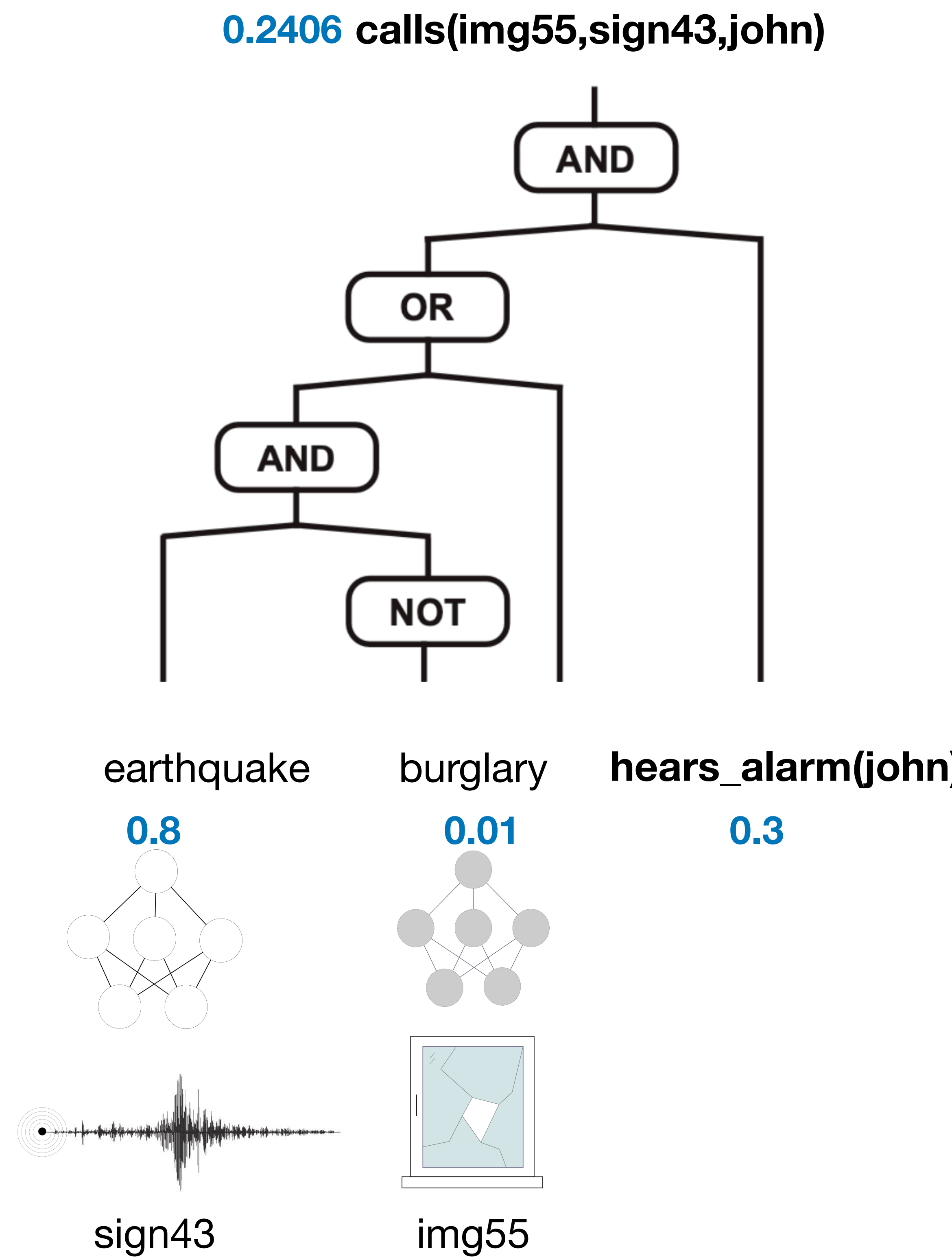
Logic

```
alarm(B,E)    IF earthquake(E) OR burglary(B)
calls(B,E,P) IF alarm(B,E) AND hears_alarm(P)

hears_alarm(john)
hears_alarm(mary)
```

Probability (or fuzzy)

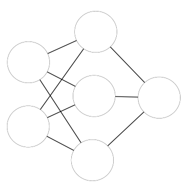
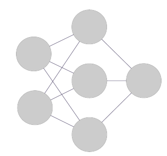
```
0.8::earthquake
0.01::burglary
0.3::hears_alarm(john)
```



Neurosymbolic functions

& primitives defining semantics and computational graph

Neural nets

signal-analysis  image-analysis 

Logic

```
alarm(B,E)   IF earthquake(E) OR burglary(B)
calls(B,E,P) IF alarm(B,E) AND hears_alarm(P)

hears_alarm(john)
hears_alarm(mary)
```

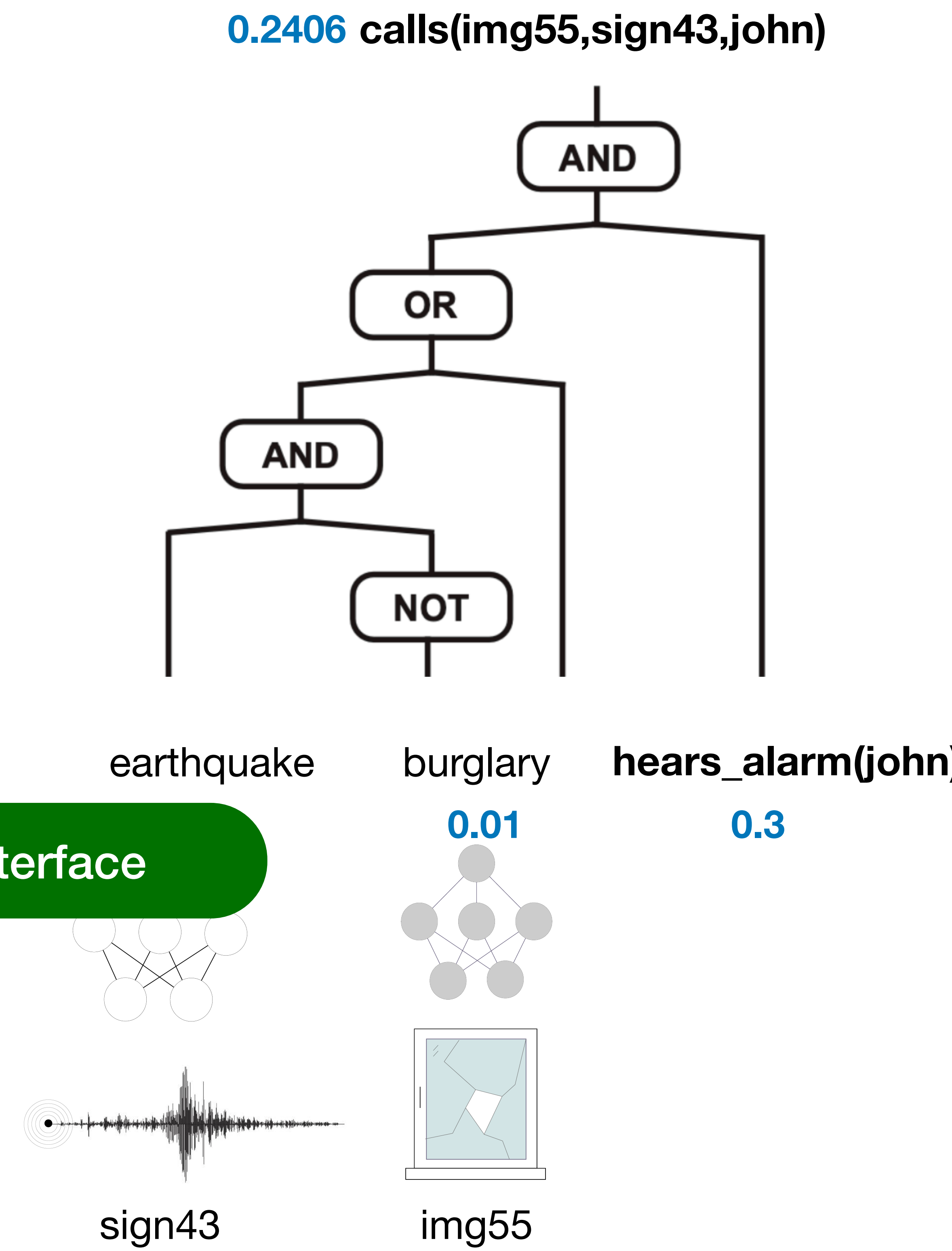
Probability (or fuzzy)

```
0.8::earthquake
0.01::burglary
0.3::hears_alarm(john)
```

Deep bidirectional interface

Neurosymbolic primitive = the neural predicate

```
burglary(B)    =  neural(image-analysis(B))
earthquake(E)  =  neural(signal-analysis(E)))
```



Neurosymbolic functions

& primitives defining semantics and computational graph

Semantics = Computational Graph

Neural nets

signal-analysis

image-analysis

Logic

```
alarm(B,E)    IF earthquake(E) OR burglary(B)
calls(B,E,P) IF alarm(B,E) AND hears_alarm(P)

hears_alarm(john)
hears_alarm(mary)
```

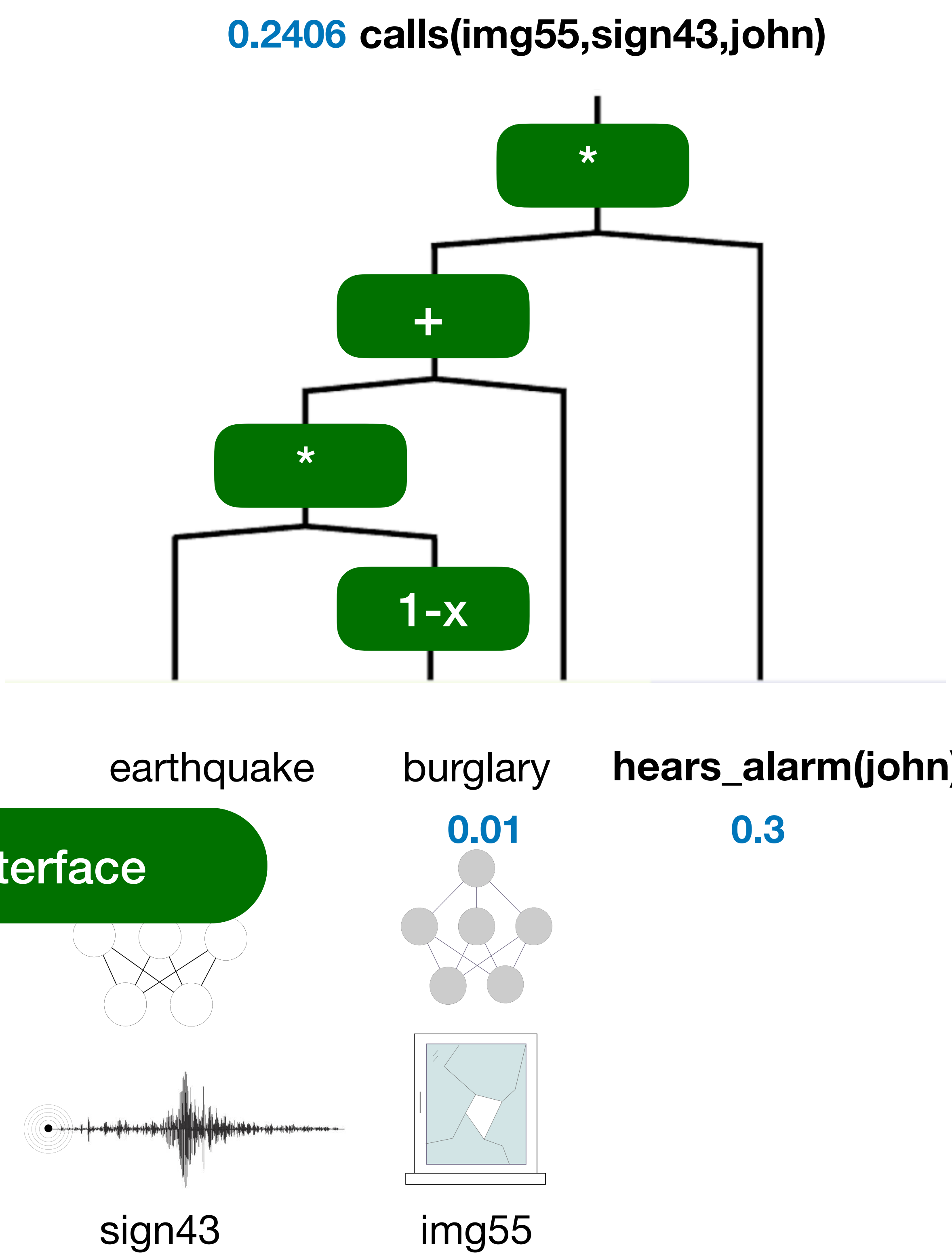
Probability (or fuzzy)

```
0.8::earthquake
0.01::burglary
0.3::hears_alarm(john)
```

Deep bidirectional interface

Neurosymbolic primitive = the neural predicate

```
burglary(B)     =  neural(image-analysis(B))
earthquake(E)   =  neural(signal-analysis(E)))
```



Inference and learning (end-to-end)

Many variations, many challenges
Standard gradient descent applies

Explanations

due to the use of a probabilistic logic, quite natural
probabilistic abduction (cf David Poole)

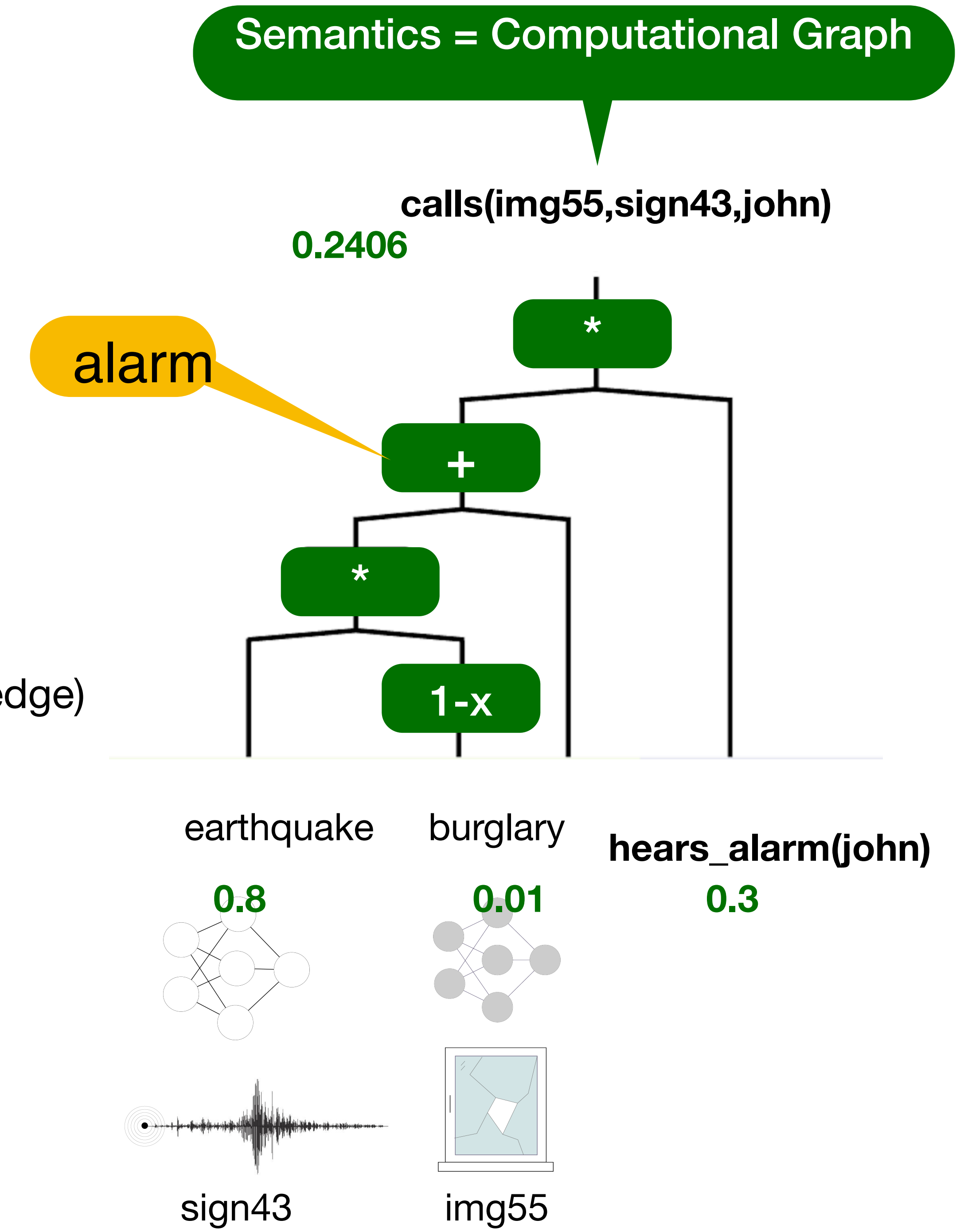
if john calls then it is because
alarm
earthquake (and no burglary)
hears_alarm(john)

you can event edit these programs (knowledge)

alarm

Semantics

How to define semantics ?
Role of components ?
(logic, fuzzy / probability, neural nets)?
Many variations



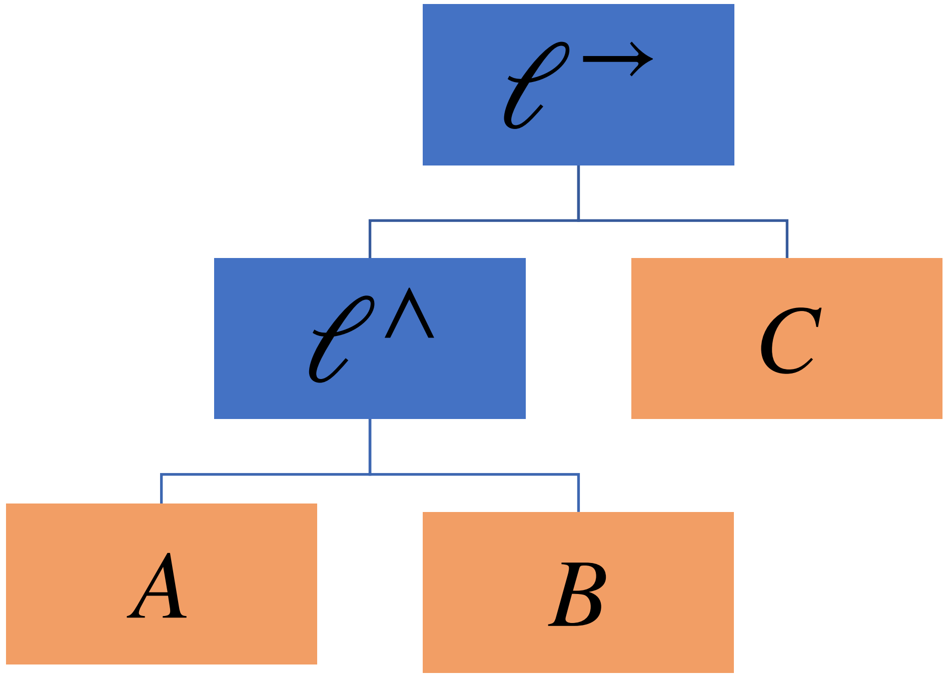
A recipe for NeSy

Where do the numbers come from ?

From logic formulae to circuits

$\ell((A \wedge B) \to C)$

$\ell(Q)$



The query Q determines the structure

A recipe for NeSy

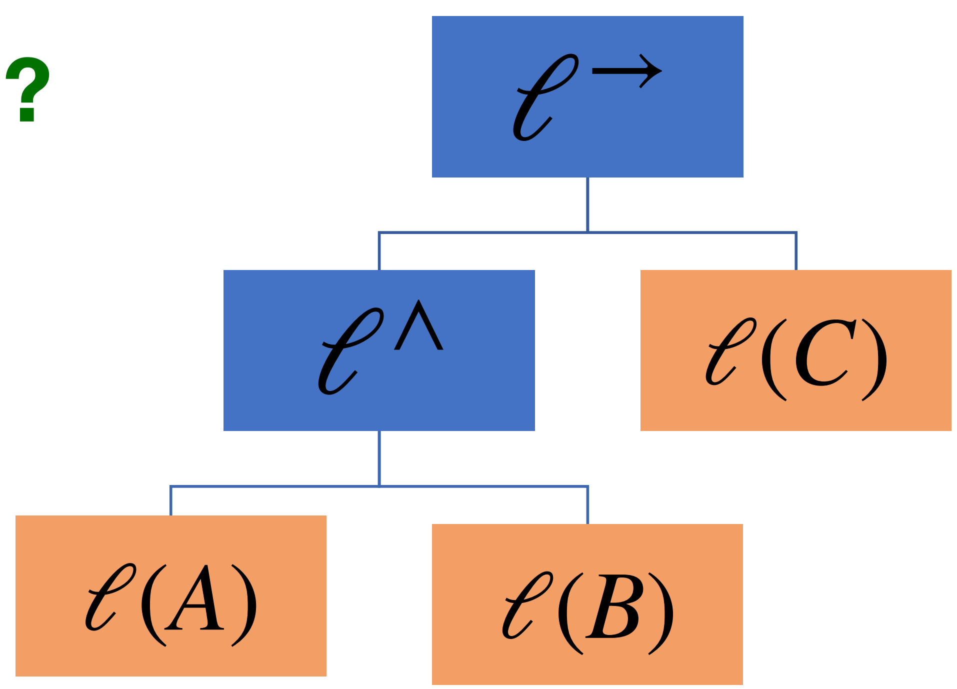
Where do the numbers come from ?

From logic formulae to circuits

$$\ell_F((A \wedge B) \rightarrow C) \qquad \ell(Q)$$

What is the algebraic structure ? = Parametric circuit

What operators ?



The query Q determines the structure (potentially after knowledge compilation)

What labeling functions ?

A recipe for NeSy

Where do the numbers come from ?

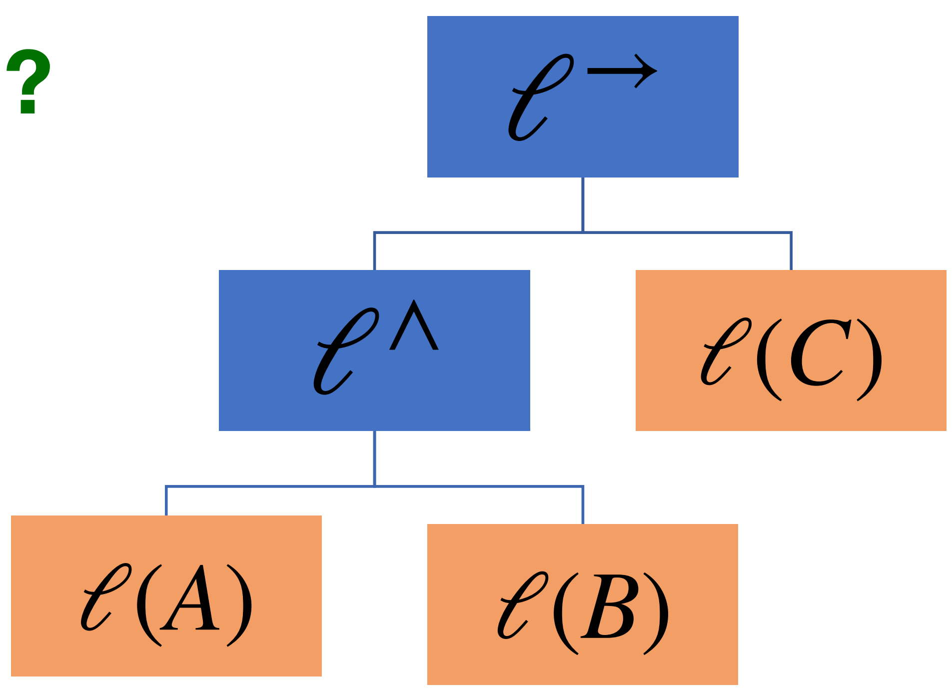
Boolean

$$\ell_F((A \wedge B) \rightarrow C)$$

p	q	p∧q
T	T	T
T	F	F
F	T	F
F	F	F

p	q	p→q
T	T	T
T	F	F
F	T	T
F	F	T

What operators ?



The query Q determines the structure (potentially after knowledge compilation)

What labeling functions ?

A recipe for NeSy

Where do the numbers come from ?

Fuzzy

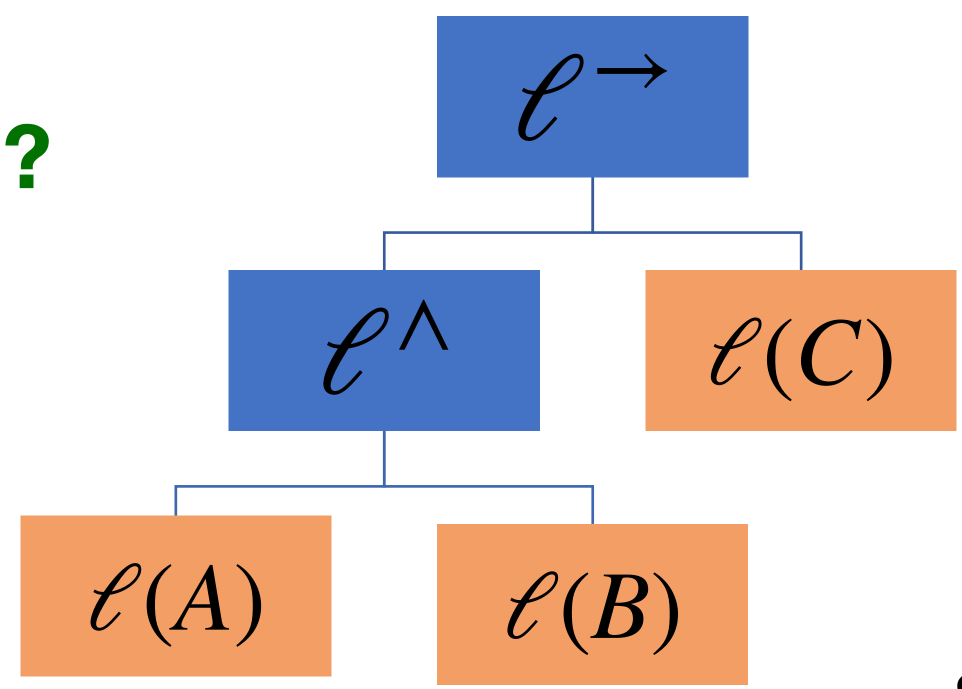
- t-norm extends conjunction to [0,1] interval
- Three fundamental t-norms:
  - Lukasiewicz t-norm: $t_L(x, y) = \max(0, x + y - 1)$
  - Goedel t-norm: $t_G(x, y) = \min(x, y)$
  - Product t-norm: $t_P(x, y) = x \cdot y$

Other operators derived from the t-norm

	Product	Łukasiewicz	Gödel
$x \wedge y$	$x \cdot y$	$\max(0, x + y - 1)$	$\min(x, y)$
$x \vee y$	$x + y - x \cdot y$	$\min(1, x + y)$	$\max(x, y)$
$\neg x$	$1 - x$	$1 - x$	$1 - x$
$x \Rightarrow y \ (x > y)$	y/x	$\min(1, 1 - x + y)$	y

What operators ?

What labeling functions ?



continuous and differentiable

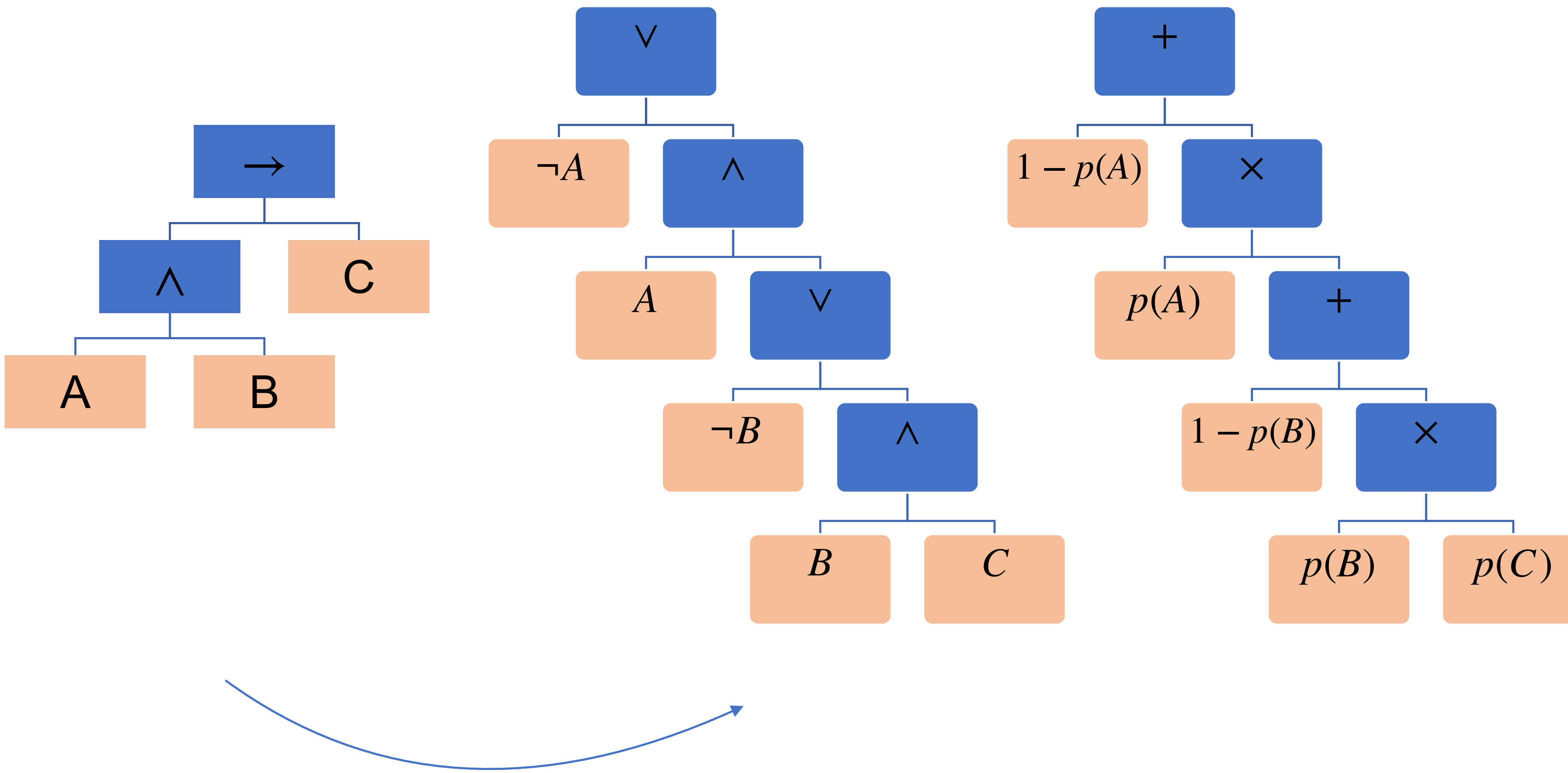
but a measure of **vagueness** not of uncertainty

Many problems
See [Van Krieken et al AIJ]

A recipe for NeSy

Where do the numbers come from ?

Probability



Knowledge Compilation **(computationally expensive)**

Probabilistic structure is explicit in compiled formula.

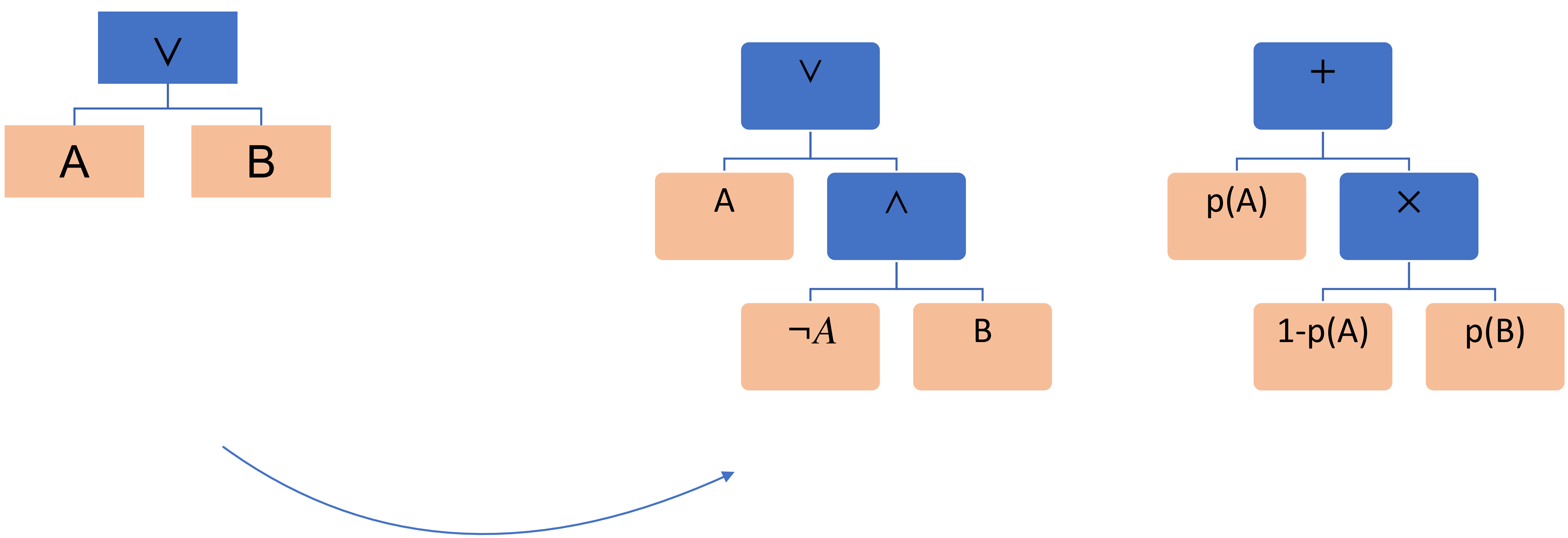
A recipe for NeSy

Where do the numbers come from ?

Probability

Why Compile

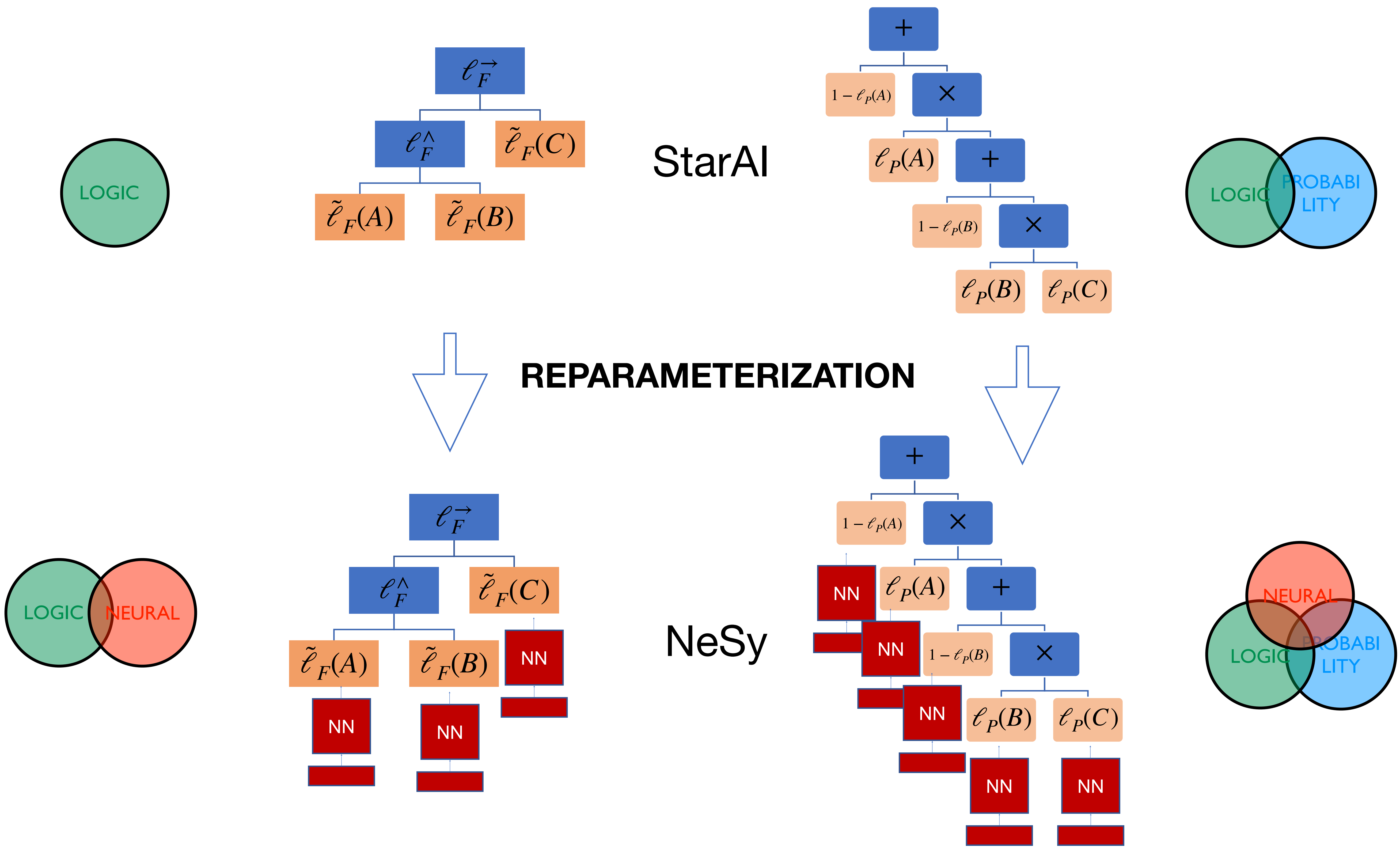
$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$



Knowledge Compilation **(computationally expensive)**

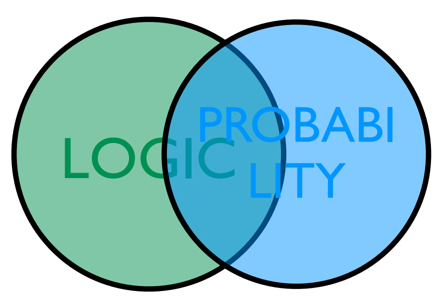
Probabilistic structure is explicit in compiled formula.

From StarAI to NeSy

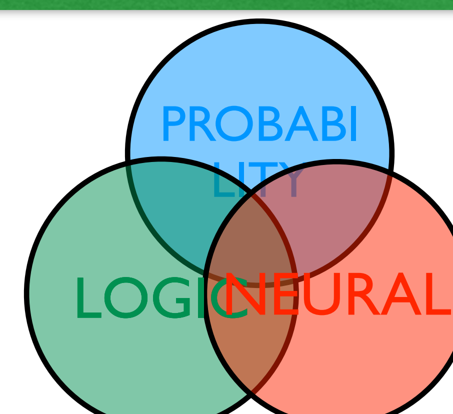


Part 3: DeepStochLog and DeepProbLog

FROM



TO



Two types of probabilistic models / programs

- Based on a *random graph model*
  - Bayesian Nets and ProbLog **-> DeepProbLog [AIJ 21]**
- Based on a *random walk model*
  - Probabilistic grammars and Stochastic Logic Programs [Muggleton] **-> DeepStochLog [AAAI 22]**

Our method/recipe:

Take an existing probabilistic logic and

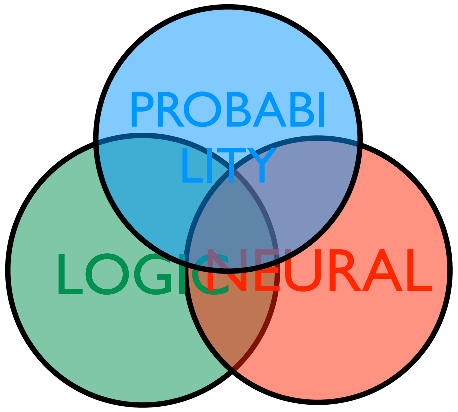
inject neural predicates that act ako interface

DeepLog

DeepStochLog = SLPs + Neural Network

DeepProbLog = ProbLog + Neural Network

Related work in NeSy	DeepProbLog and DeepStochLog
Logic is made less expressive	Full expressivity is retained
Logic is pushed into the neural network	Maintain both logic and neural networ
Fuzzy logic	Probabilistic logic programming
Language semantics unclear	Clear semantics



DeepStochLog

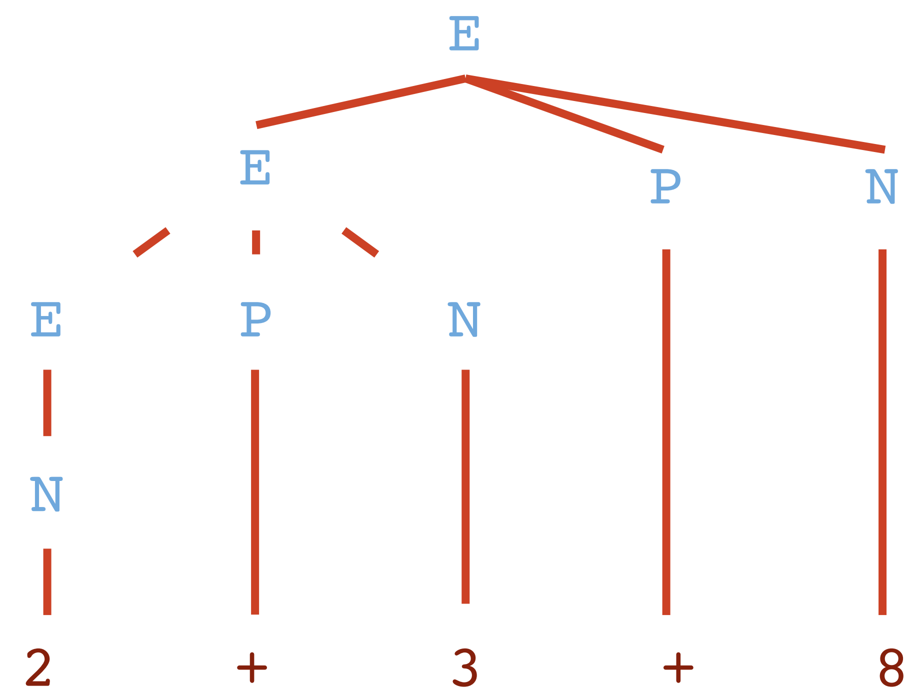
- Little sibling of DeepProbLog [Winters, Marra, et al AAAI 22]
- Based on a different semantics
  - **probabilistic graphical models vs grammars**
  - **random graphs vs random walks**
- Underlying StarAI representation is **Stochastic Logic Programs** (Muggleton, Cussens)
  - close to Probabilistic Definite Clause Grammars, ako probabilistic unification based grammar formalism
  - again the idea of neural predicates
- Scales better, is faster than DeepProbLog

CFG: Context-Free Grammar

```
E --> N
E --> E, P, N

P --> ["+"]

N --> ["0"]
N --> ["1"]
...
N --> ["9"]
```



Useful for:

- Is sequence an element of the specified language?
- What is the *"part of speech"*-tag of a terminal
- Generate all elements of language

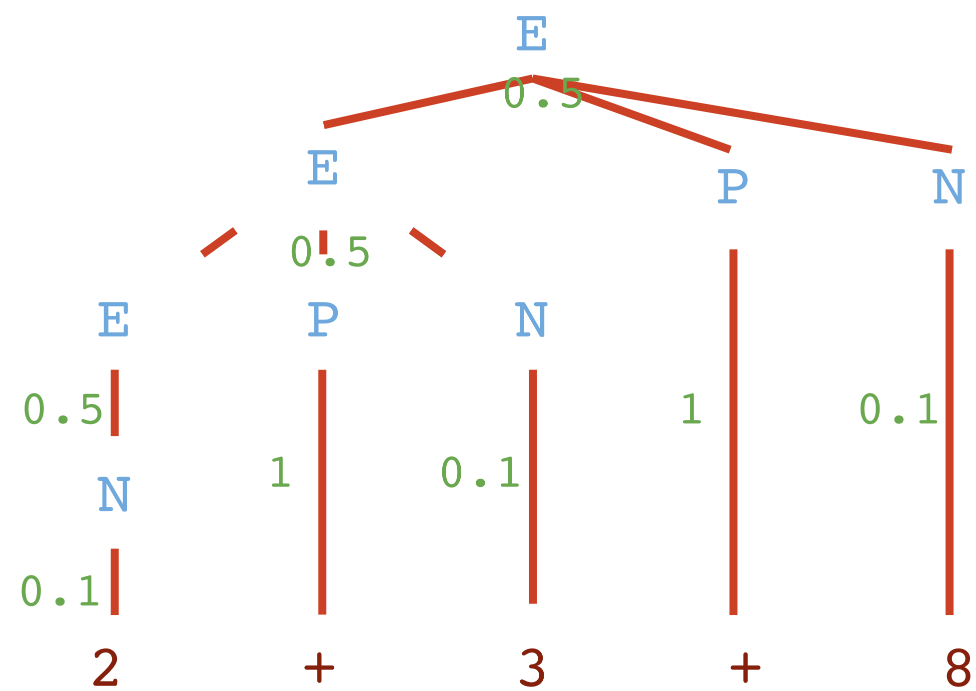
PCFG: Probabilistic Context-Free Grammar

```
0.5 :: E --> N
0.5 :: E --> E, P, N

1.0 :: P --> ["+"]

0.1 :: N --> ["0"]
0.1 :: N --> ["1"]
        ...
0.1 :: N --> ["9"]
```

Always sums to 1 per non-terminal



*Probability of this parse = 0.5*0.5*0.5*0.1*1*0.1*1*0.1*
= 0.000125

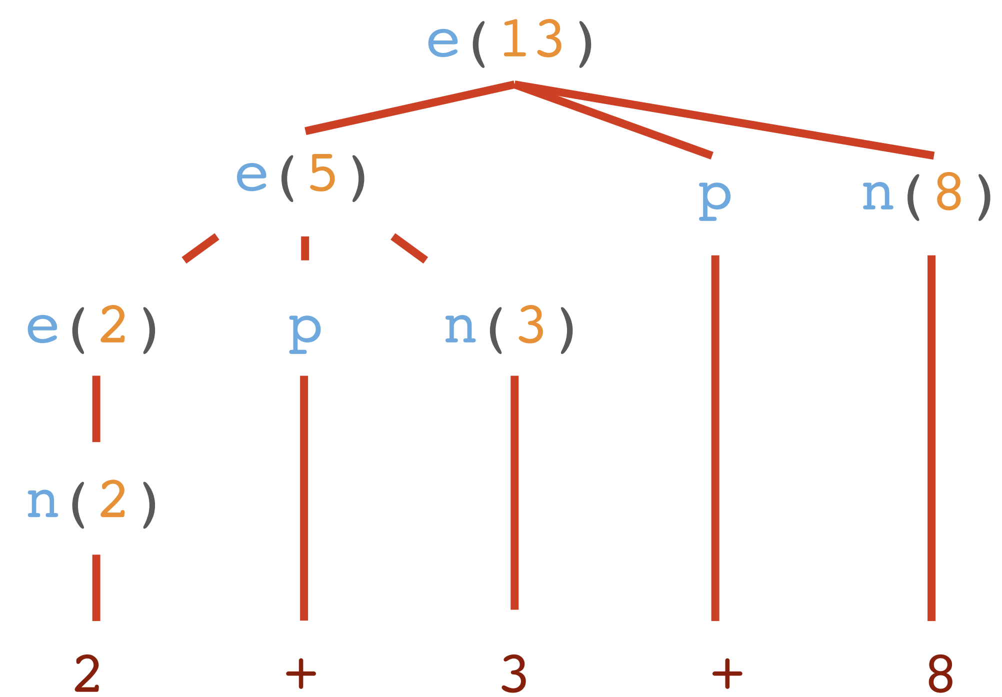
Useful for:

- What is the most likely parse for this sequence of terminals? *(useful for ambiguous grammars)*
- What is the probability of generating this string?

DCG: Definite Clause Grammar

```
e(N) --> n(N).
e(N) --> e(N1), p, n(N2),
         {N is N1 + N2}.
p    --> ["+"].

n(0) --> ["0"].
n(1) --> ["1"].
…
n(9) --> ["9"].
```

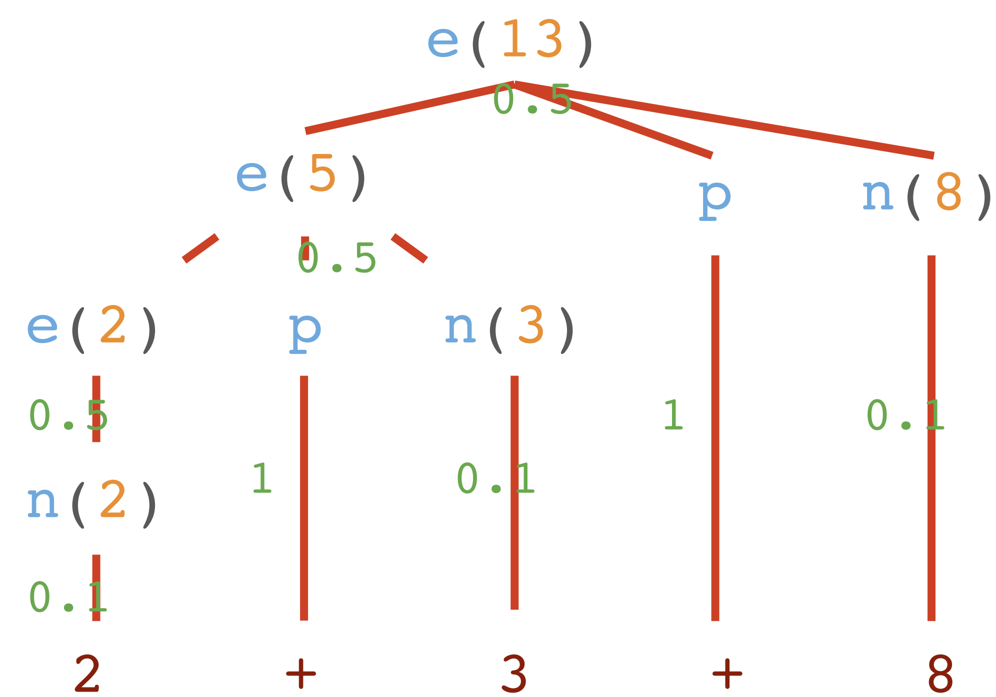


Useful for:
- Modelling more complex languages *(e.g. context-sensitive)*
- Adding constraints between non-terminals thanks to Prolog power *(e.g. through unification)*
- Extra inputs & outputs aside from terminal sequence *(through unification of input variables)*

SDCG: Stochastic Definite Clause Grammar

```
0.5 :: e(N) --> n(N).
0.5 :: e(N) --> e(N1), p, n(N2),
                {N is N1 + N2}.
1.0 :: p    --> ["+"].

0.1 :: n(0) --> ["0"].
0.1 :: n(1) --> ["1"].
        …
0.1 :: n(9) --> ["9"].
```

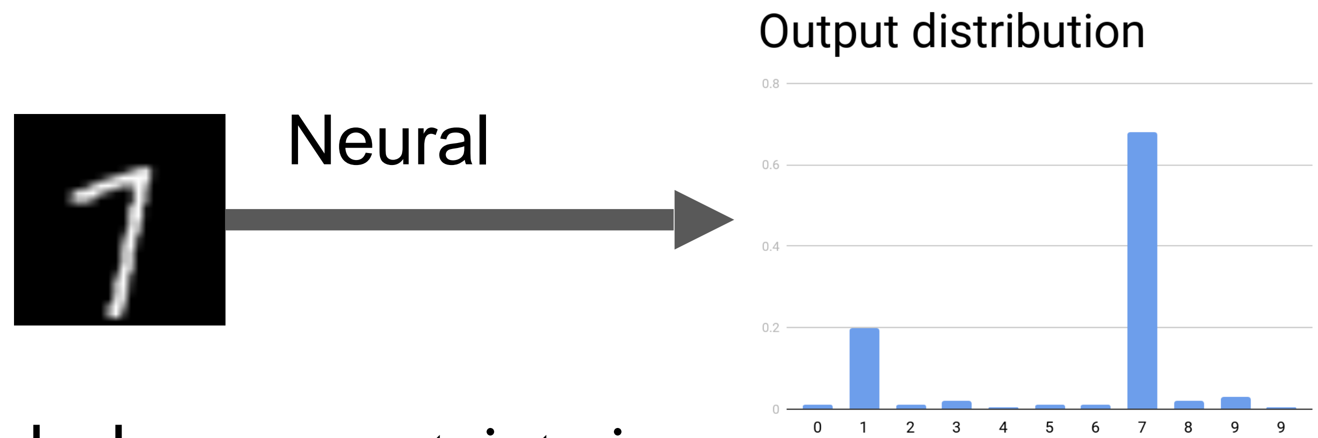


*Probability of this parse = 0.5*0.5*0.5*0.1*1*0.1*1*0.1*
= 0.000125

Useful for:

- Same benefits as PCFGs give to CFG *(e.g. most likely parse)*
- But: loss of probability mass possible due to failing derivations

Neural predicate



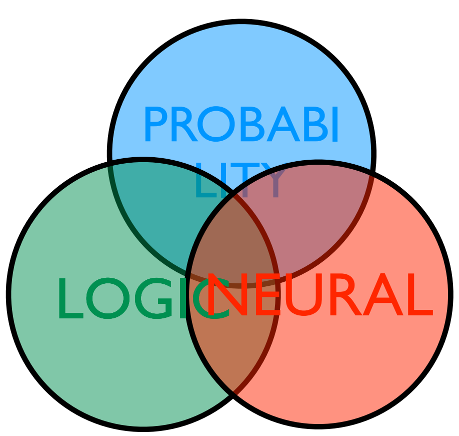
- Neural networks have uncertainty in their predictions
- A normalized output can be interpreted as a probability distribution
- Neural predicate models the output as probabilistic facts
- No changes needed in the probabilistic host language

Key Idea DeepProbLog

unify the basic concepts in logic and neural networks:

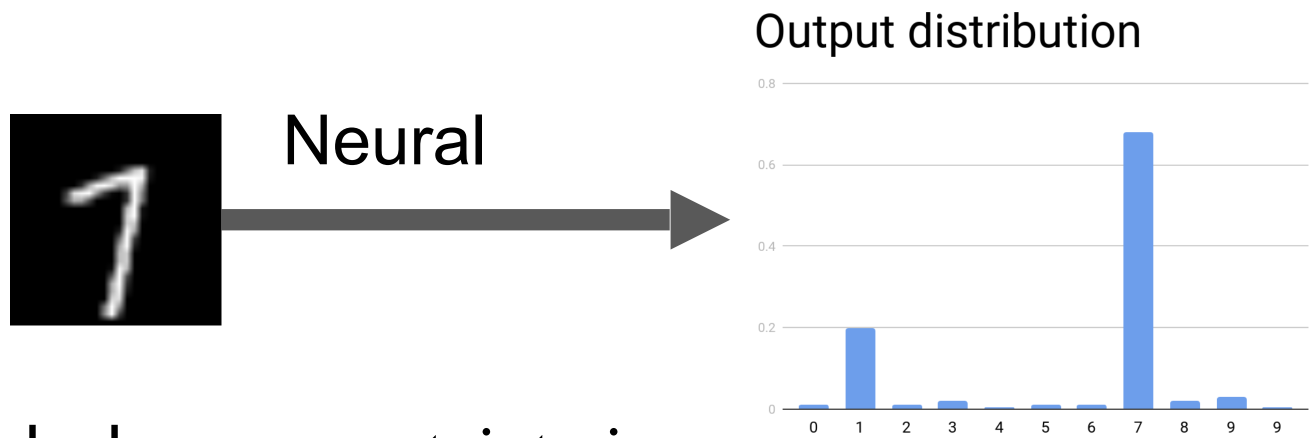
neural predicate ~ neural net

an interface between logic and neural nets




```
0.04::digit( ,0) XOR 0.35::digit( ,1) XOR ... XOR
0.53::digit( ,7) XOR ... XOR 0.014::digit( ,9).
```

Neural predicate



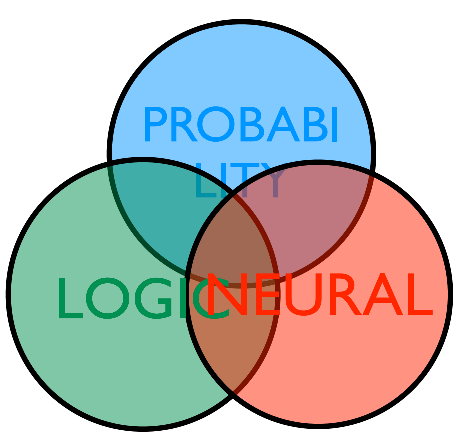
- Neural networks have uncertainty in their predictions
- A normalized output can be interpreted as a probability distribution
- Neural predicate models the output as probabilistic facts
- No changes needed in the probabilistic host language

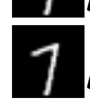
Key Idea DeepProbLog

unify the basic concepts in logic and neural networks:

neural predicate ~ neural net

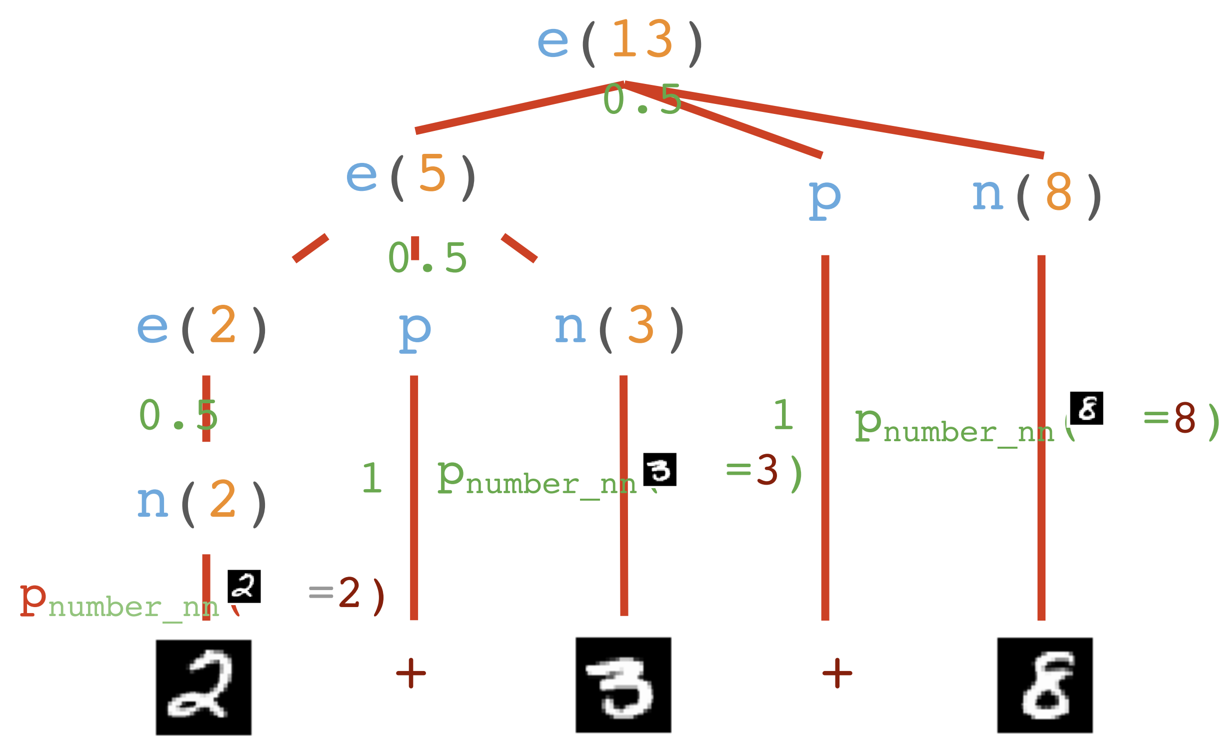
an interface between logic and neural nets



0.04::digit(,0) XOR 0.35::digit(,1) XOR ... XOR
0.53::digit(,7) XOR ... XOR 0.014::digit(,9).

NDCG: Neural Definite Clause Grammar (= DeepStochLog)

```
0.5 :: e(N) --> n(N).
0.5 :: e(N) --> e(N1), p, n(N2),
                {N is N1 + N2}.
1.0 :: p    --> [“+”].
nn(number_nn,[X],[Y],[digit])::
        n(Y) –> [X].
digit(Y) :-
      member(Y,[0,1,2,3,4,5,6,7,8,9]).
```



Probability of this parse =

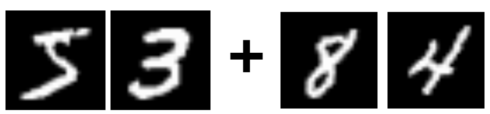
0.5*0.5*0.5*p_{number_nn}(=2)*1*p_{number_nn}(=3)*1*p_{number_nn}(=8)

Useful for:

- Subsymbolic processing: e.g. tensors as terminals
- Learning rule probabilities using neural networks

Mathematical expression outcome

T1: Summing MNIST numbers with pre-specified # digits



= 137

Table 1: The test accuracy (%) on the MNIST addition (**T1**).

	Number of digits per number (N)			
Methods	1	2	3	4
NeurASP	97.3 ± 0.3	93.9 ± 0.7	timeout	timeout
DeepProbLog	97.2 ± 0.5	95.2 ± 1.7	timeout	timeout
DeepStochLog	97.9 ± 0.1	96.4 ± 0.1	94.5 ± 1.1	92.7 ± 0.6

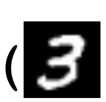
Table 2: The accuracy (%) on the HWF dataset (**T2**).

	Expression length			
Method	1	3	5	7
NGS	90.2 ± 1.6	85.7 ± 1.0	91.7 ± 1.3	20.4 ± 37.2
DeepProbLog	90.8 ± 1.3	85.6 ± 1.1	timeout	timeout
DeepStochLog	90.8 ± 1.0	86.3 ± 1.9	92.1 ± 1.4	94.8 ± 0.9

T2: Expressions with images representing operator or single digit number.

= 19

Rules of Addition Known — Impose Strong Constraints on Neural Nets

addition( ,  ,8) IF and only IF digit( N1), digit( ,N2), 8 = N1 + N2.

Citation networks

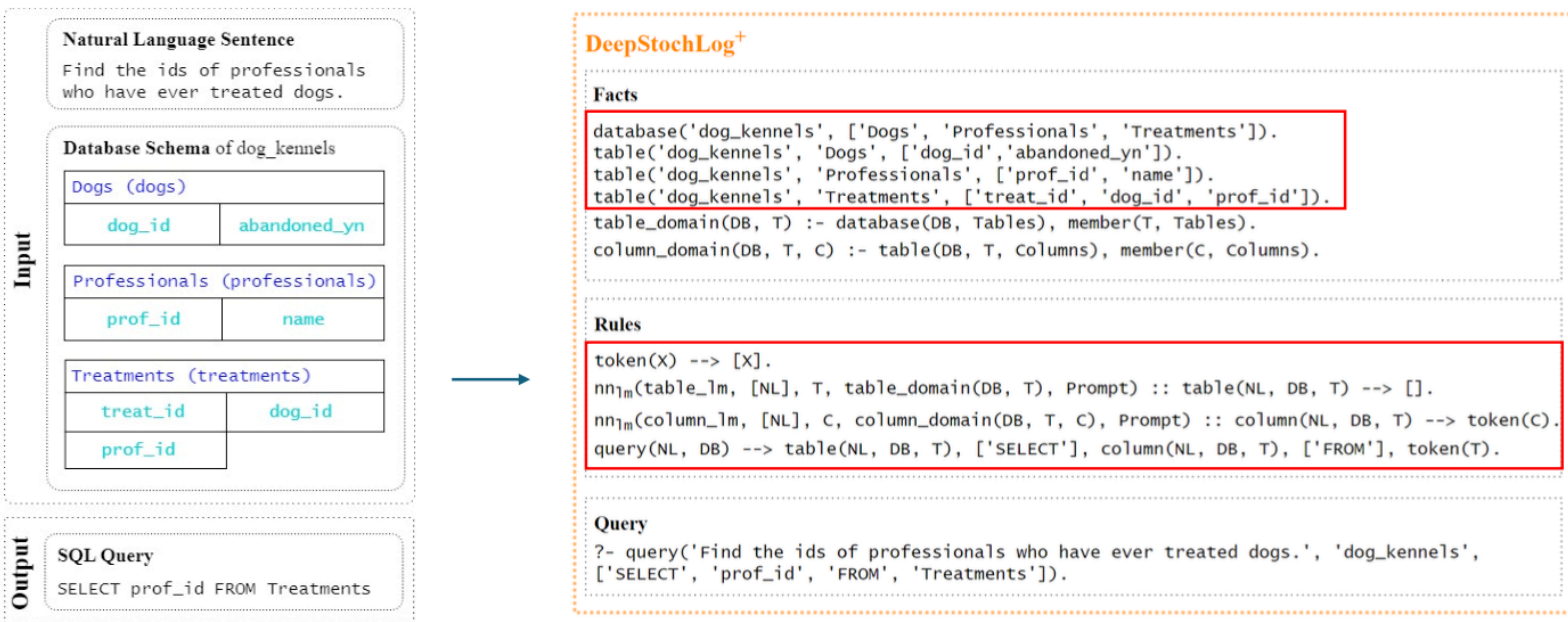
T5: Given scientific paper set with only few labels & citation network, find all labels

Table 5: **Q3** Accuracy (%) of the classification on the test nodes on task **T5**

Method	**Citeseer**	**Cora**
ManiReg	60.1	59.5
SemiEmb	59.6	59.0
LP	45.3	68.0
DeepWalk	43.2	67.2
ICA	69.1	75.1
GCN	70.3	81.5
DeepProbLog	timeout	timeout
DeepStochLog	65.0	69.4

Applied to NL to SQL

Training:



$$p(SQL_{gt}|NL, DB) = p([\text{'SELECT', 'prof\_id', 'FROM', 'Treatments'}] \mid \text{'Find ... dogs.', 'dog\_kennels'})$$

Ying Jiao et al, NeSy 24

Soft-unification in Deep Probabilistic Logic

How can we reason symbolically over distributed representations?

isIn( , france) ∧ isIn( , EU)

-> isIn(eiffel_tower, )

DeepSoftLog: Reasoning over embeddings in Problog with sound probabilistic semantics.

Jaron Maene & LDR

DeepSoftLog (NeurIPS 23)

Theorem: If we interpret the soft-unification as a probability, we and take a soft-unification function of the form $e^{-d(x,y)}$ with d a distance, we get:

(1) Well-defined proof scores
(2) No redundancy in proofs
(3) Connected embedding space
(4) Non-sparse gradients

+ a source transformation of this to DeepProbLog

DeepSeaProbLog

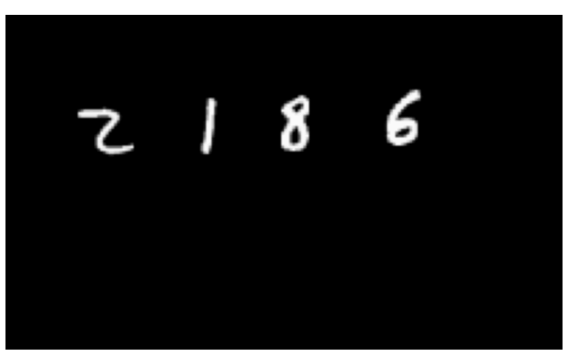
discrete and continuous distributions [De Smet UAI 23]

useful for robotics and perception

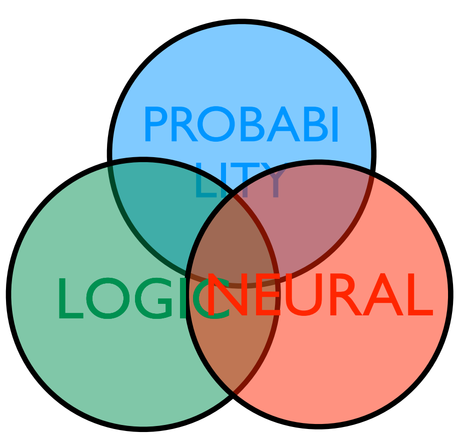
dim is neural net returning parameters of normal distribution.

```
length(Obj) ~ normal(dim(Obj,Image)).

large(Obj) :- length(Obj) > 100.
```



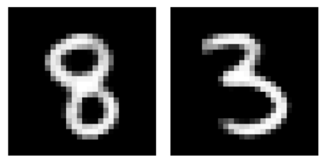
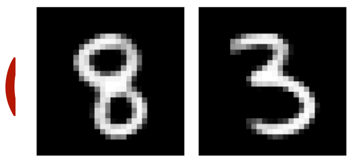
determining order digits to determine year

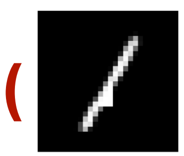


DeepSeaProbLog

discrete and continuous distributions [De Smet UAI 23]

generative model with variational autoencoders (see also [Misoni et al NeurIPS 22])

So far from input  to output 11 so that **SUM(**  **,11)** holds

In DeepSeaProblog, you can query **SUM(**  **, X, 5)**

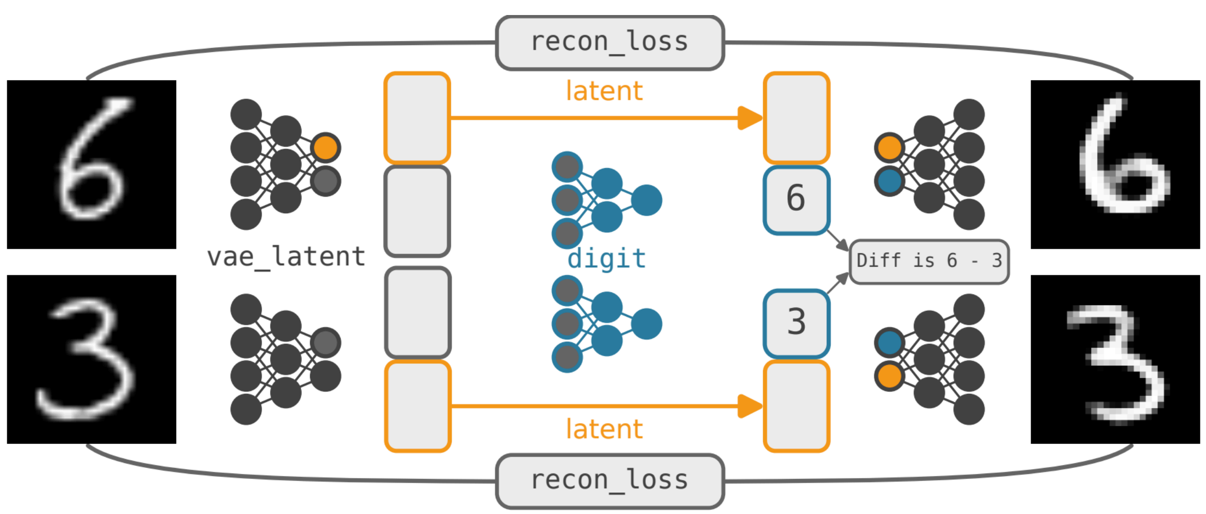
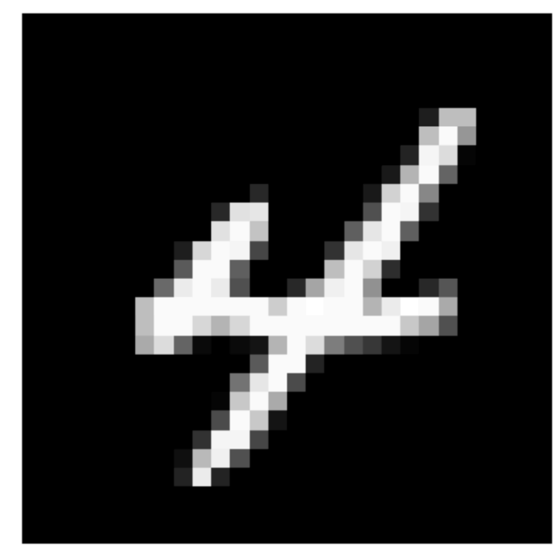
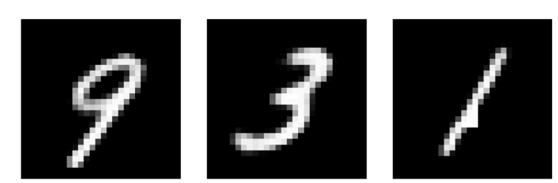


Figure 4: Given example pairs of images and the value of their subtraction, e.g., (,) and 3, the CVAE encoder `vae_latent` first encodes each image into a multivariate normal NDF (`latent`) and a latent vector. The latter is the input of a categorical NDF `digit`, completing the CVAE latent space. Supervision is dual; generated images are compared to the original ones in a probabilistic reconstruction loss, while both digits need to subtract to the given value.





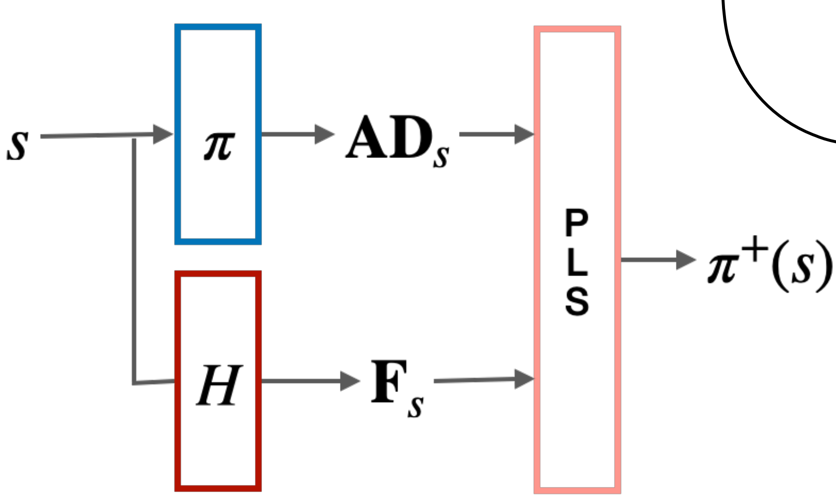
LOGIC NEURAL

Probabilistic Logic Shield for Reinforcement Learning

Wen-chi Yang et al, IJCAI 23 Distinguished paper award

Assuming noisy sensors

$$\begin{cases} \pi(\texttt{accelerate} \mid s) = 0.5 \\ \pi(\texttt{left} \mid s) = 0.3 \\ \pi(\texttt{right} \mid s) = 0.2 \end{cases}$$



Shield

```
0.8 :: obstc(front).
0.2 :: obstc(left).
0.5 :: obstc(right).

0.5 :: act(accel);
0.3 :: act(left);
0.2 :: act(right)

0.9 :: crash:- obstc(front), act(accel).
0.4 :: crash:- obstc(left), act(left).
0.4 :: crash:- obstc(right), act(right).
safe:- ¬crash.
```

DeepProbLog Theory (Manhaeve et al. AIJ)

Will stay undamaged?

$$\mathbf{P}(\texttt{safe} \mid a, s) = \begin{cases} \texttt{accelerate} & \rightarrow 0.28 \\ \texttt{left} & \rightarrow 0.92 \\ \texttt{right} & \rightarrow 0.8 \end{cases}$$

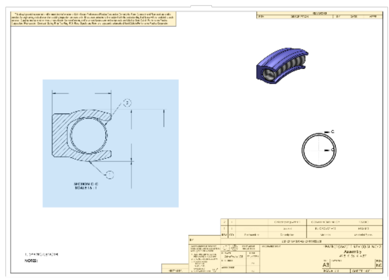
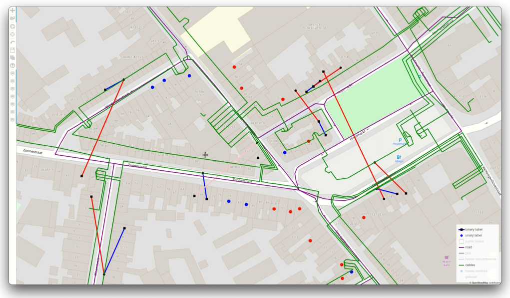
Probability of staying safe if following π?

$$\mathbf{P}_\pi(\texttt{safe} \mid s) = 0.576$$

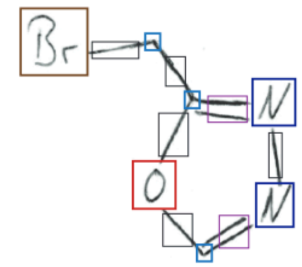
What is a safer policy π^+?

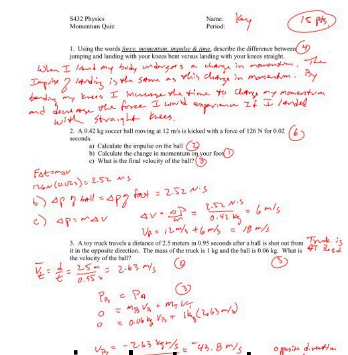
$$\begin{cases} \pi^+(\texttt{accelerate} \mid s) = 0.24 \\ \pi^+(\texttt{left} \mid s) = 0.48 \\ \pi^+(\texttt{right} \mid s) = 0.28 \end{cases}$$

Emerging applications

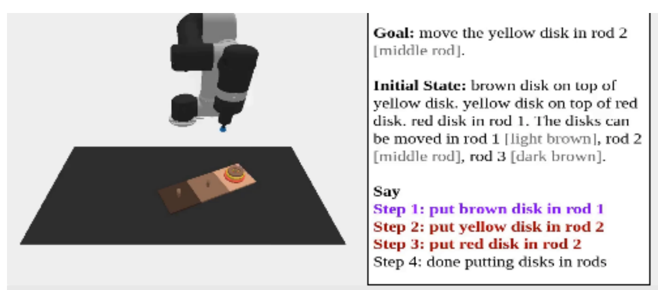



automated engineering assistant (IAAI 21)
interpret and correct designs and maps

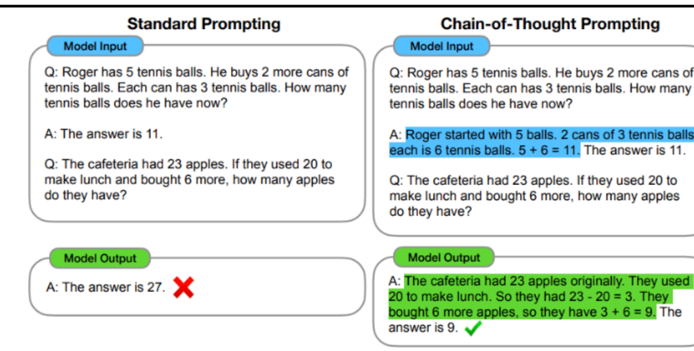




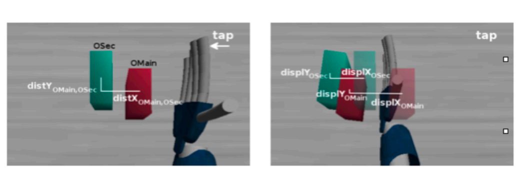
Intelligent OCR for chemical structures (ICLR 23)
and forms



planning, reinforcement learning and
shielding (AAAI 24, IJCAI 23)

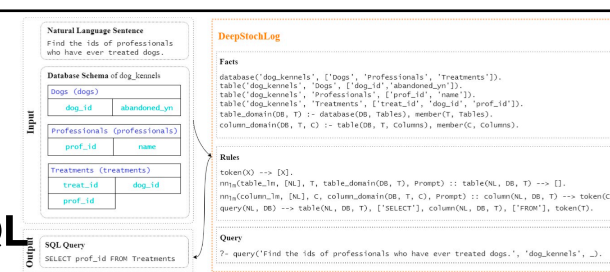


reasoning and mathematical problem
solving (JAIR 23, IJCAI 2017, EMNLP 21)



cognitive robotics (IJCAI 20, IEEE Trans)

NLP to SQL

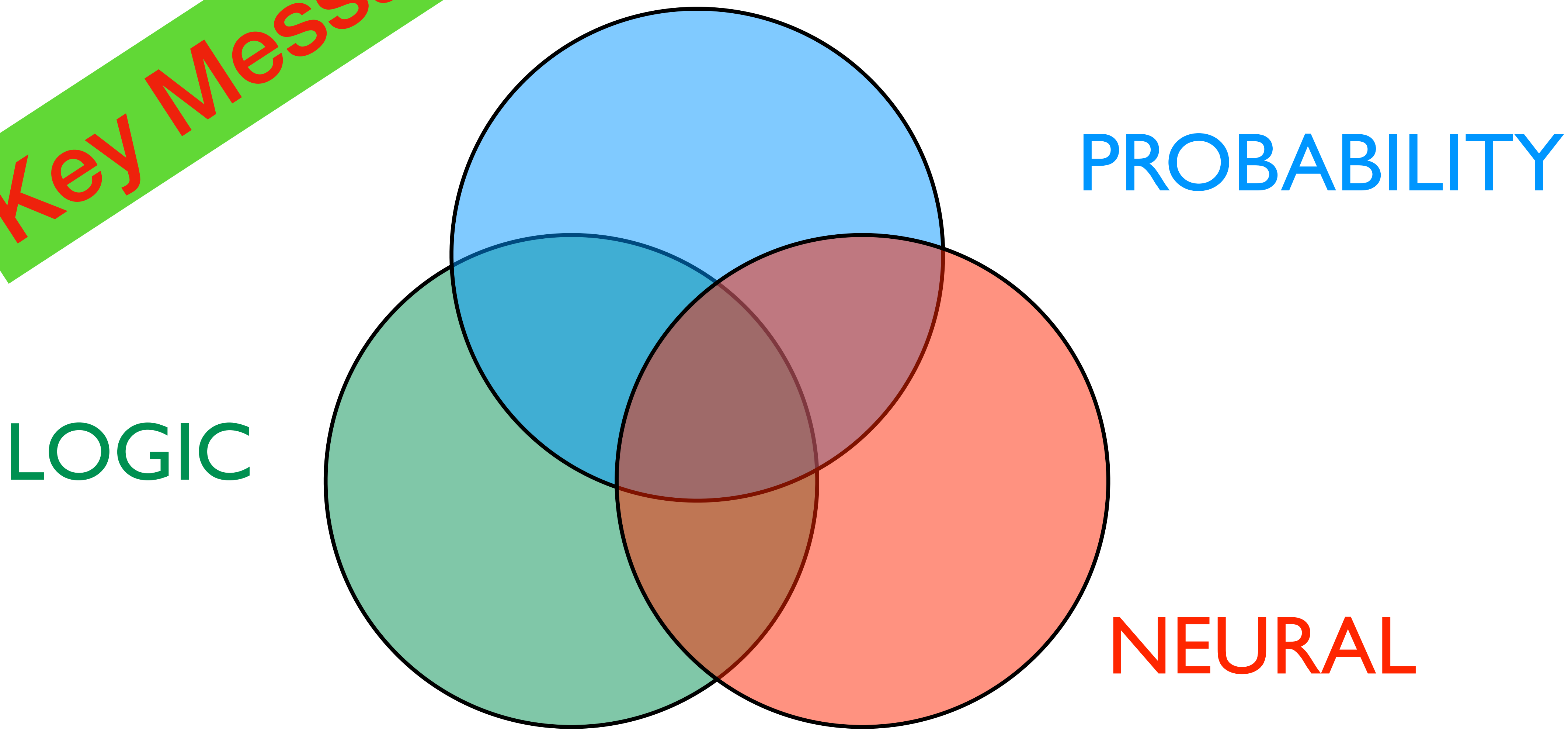


Challenges

- For NeSy,
  - scaling up (but serious progress !!)
  - which models and which knowledge to use
  - large scale life applications
  - peculiarities of neural nets & fuzzy logic
  - dynamics / continuous
  - **theory is largely missing !!!**
- This is an excellent area for starting researchers / PhDs

Neurosymbolic =
Neuro + Logic + Probability

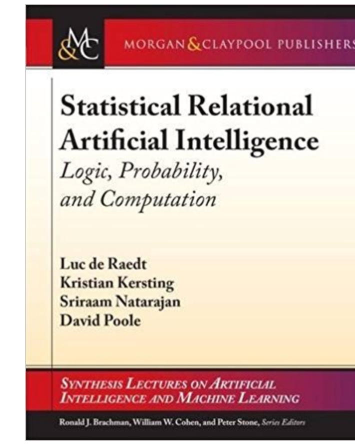
Key Message 1

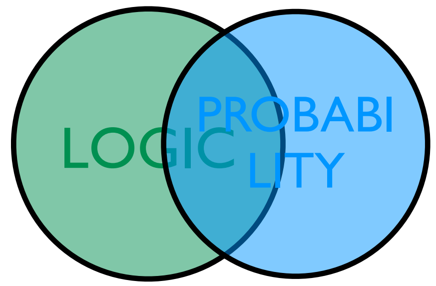


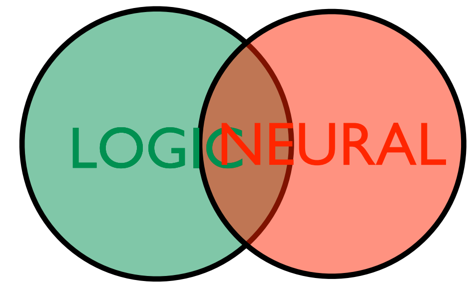
see Manhaeve et al. NeSy Book

interpret PROBABILITY broadly (including fuzzy)

Key Message 2







StarAI and NeSy share similar problems and thus similar solutions apply

See also [De Raedt et al., IJCAI 20; Marra et al, arxiv]

Key Message 3

Provide recipe for

Kautz

Neural : : Symbolic

“an interface layer (<> pipeline) between neural & symbolic components”

erc