NeSy Learning
Day 1: Introduction

Efthymia Tsamoura
Huawei Labs

Emile van Krieken
VU, Amsterdam

ESSLLI 2025

About this Course

- ✓Day 1: Introduction to NeSy
- ✓Day 2: Learnability
- ✓Day 3: Learning Imbalances in NeSy
- ✓Day 4: Reasoning Shortcuts
- ✓Day 5: Other Aspects

Outline of Today's Lecture

- ✓ **Basic notions**
- ✓ **Motivation**
- ✓ **Types of Integration**
- ✓ **Enforcing Boolean Constraints During Training**
  - ✓ **Losses Based on Probabilistic Logic**
  - ✓ **Application in Computer Vision**
  - ✓ **Losses Based on Fuzzy Logic**

About Neural Networks

Neural networks are **functions of (unknown) parameters** from an input space $\mathscr{X}$ to an output one $\mathscr{Y}$

✓ Examples of **inputs:** images, video, text

About Neural Networks

Depending on their outputs, we have two types of networks:

✓ **Classifiers:** They output the **class** the input belongs to

✓ **Generative models:** They output new content, e.g., images, text, videos

Example: Classifiers

Each **prediction** is associated with a **probability of being** true, called the **softmax probability**

Neural network

x_{raven}

x_{crow}

$x_{blackbird}$

$x_{panther}$

Output neurons, one per possible class

Given: An image

Output: Its class, e.g., **raven**, **panther**

Example: Generative Models

Describe me the image

Neural network

The images shows a kitchen with a table in the middle. There is a bowl of fruits on the top.

About Neural Networks

How to learn the parameters of the networks?

- ✓ We **present them examples.** Two options:
  - ✓ Present the **gold** output (supervised learning)
  - ✓ Do not present the **gold** output (unsupervised learning)

Example: Classifiers

Each **prediction** is associated with a **probability of being** true, called the **softmax probability**

Neural network

x_{raven}

x_{crow}

$x_{blackbird}$

$x_{panther}$

Given: An image and its target class

Goal: Learn the parameters of the network

Learning depends on a deferentiable function called **loss**. The loss measures how "far" are the current predictions from the target class

About Logic

Logic is a mathematical formalism for

✓ **Representing knowledge**

✓ **Deriving new knowledge** given existing knowledge

About Logic

What kind of knowledge can we represent?

- ✓ **Positive vs Negative**
- ✓ **Static vs Temporal**
- ✓ **Certain vs Uncertain**
- ✓ **Lack of knowledge**

Logic Examples: Boolean Logic

Knowledge: The bird is a **raven or crow or blackbird**

Logic representation: $X_{raven} \vee X_{crow} \vee X_{blackbird}$

Logic Examples: Temporal Logic

Knowledge: How to make tea

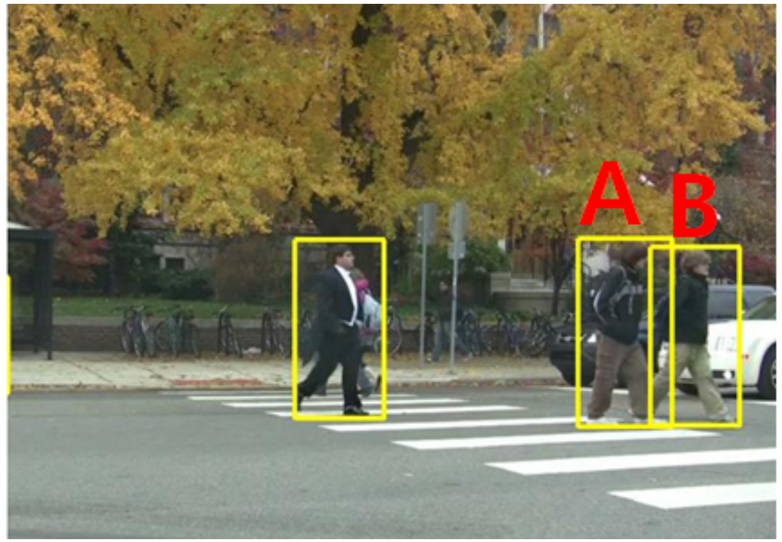
```
pour_water must after take_cup;
add_teabag must after take_cup;
never pour_juice if add_teabag;
```

Logic representation:

$$\psi = (\neg \text{pour_water}\ W\ \text{take_cup})$$
$$\wedge (\neg \text{add_teabag}\ W\ \text{take_cup})$$
$$\wedge (F\ \text{add_teabag} \rightarrow \neg F\ \text{pour_juice})$$

Video frames taken from Ziwei Xu et al., NeurIPS 2022

Logic Examples: First-Order Uncertain Logic



Task: Understand the activity of a group of actors in a video

Knowledge: If B is close to A, then B most likely do the same activity with A

Logic representation:

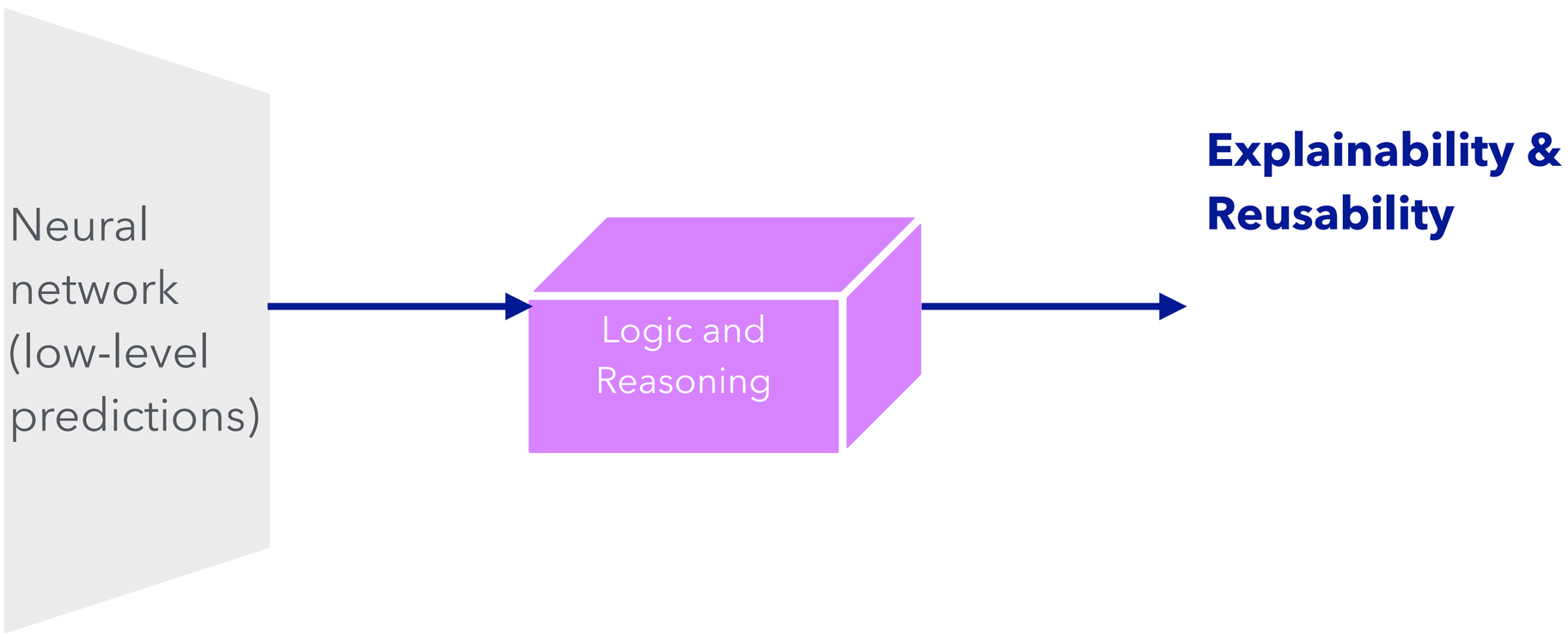
$$\text{DOING}(A, \text{activity}) \wedge \text{CLOSE}(A,B) \xrightarrow{0.75} \text{DOING}(B, \text{activity})$$

These prerequisites may be uncertain themselves

Outline of Today's Lecture

- ✓ **Basic notions**
- ✓ **Motivation**
- ✓ **Types of Integration**
- ✓ **Enforcing Boolean Constraints During Training**
  - ✓ **Losses Based on Probabilistic Logic**
  - ✓ **Application in Computer Vision**
  - ✓ **Losses Based on Fuzzy Logic**

Why


Neural network (low-level predictions)
Logic and Reasoning
Explainability & Reusability

Why: Building Simpler Models

Task: Understand the activity of an actor in a video

```
pour_water must after take_cup;
add_teabag must after take_cup;
never pour_juice if add_teabag;
```

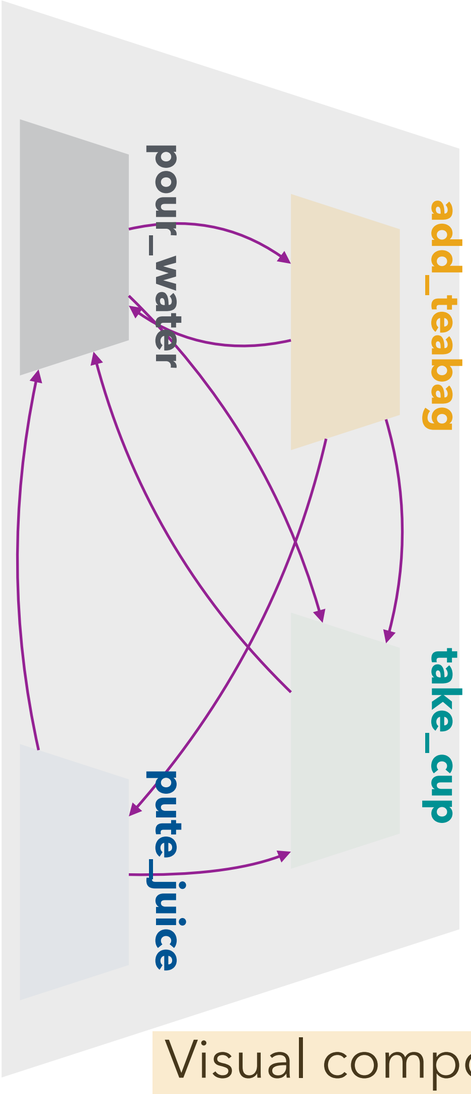
$$\psi = (\neg \text{pour_water}\ W\ \text{take_cup}) \\ \wedge (\neg \text{add_teabag}\ W\ \text{take_cup}) \\ \wedge (F\ \text{add_teabag} \rightarrow \neg F\ \text{pour_juice})$$

Questions:

1. How could we use this logic to build a simpler network
2. Why the overall framework becomes more trustworthy?

Video frames taken from Ziwei Xu et al., NeurIPS 2022

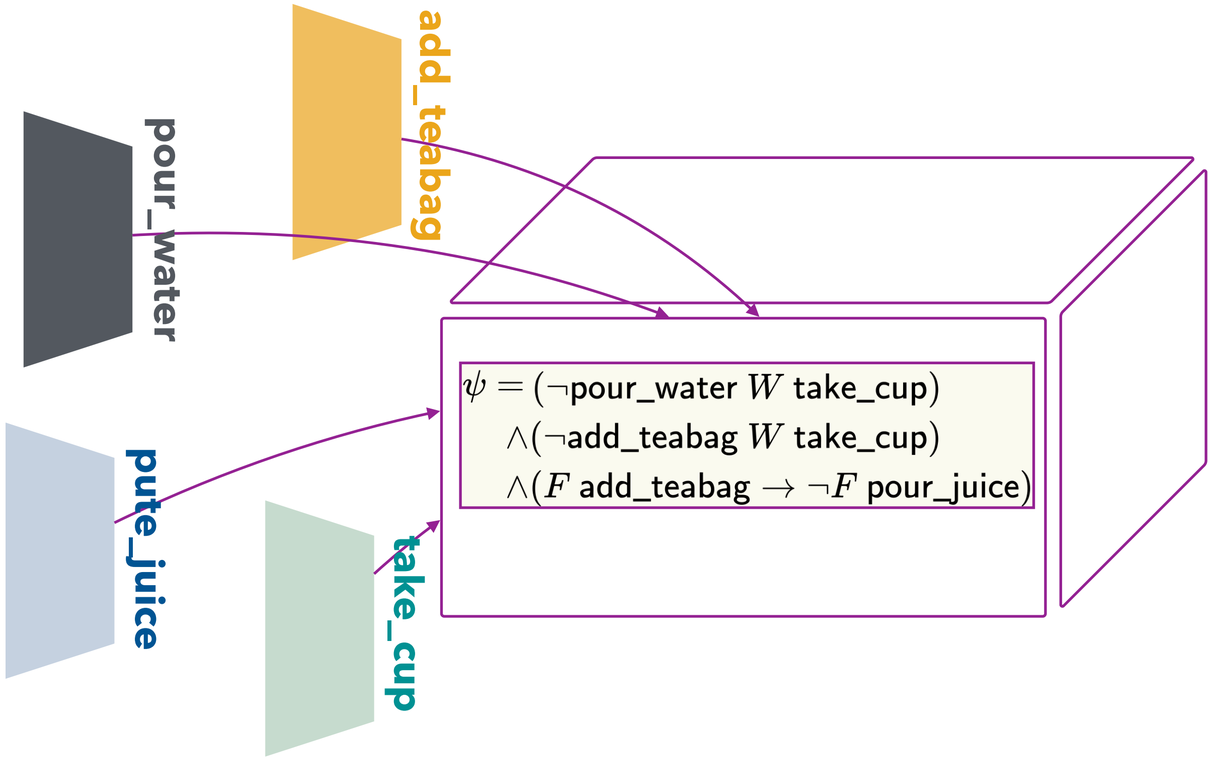
Why: Building Simpler Models



Visual components & logic are **baked in**

✓ Logic **unknown**

✓ Visual components **not extractable**

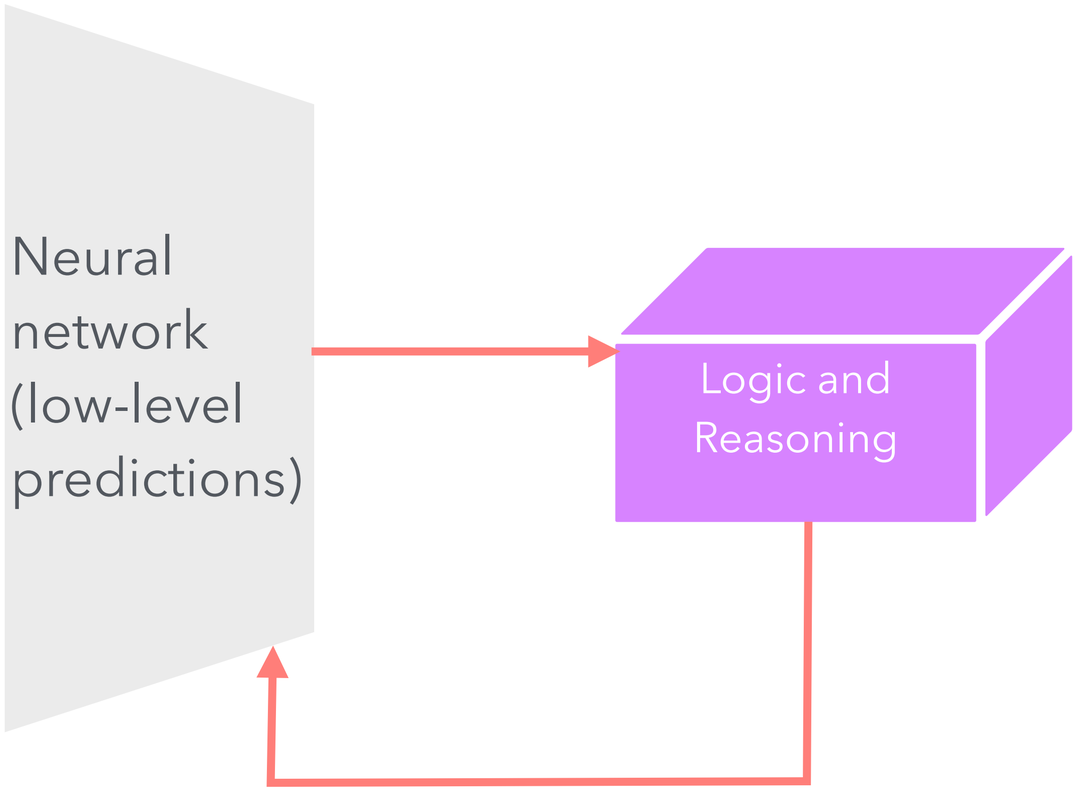


Visual components & logic are **separated**

✓ Logic **known**

✓ Visual components **extractable**

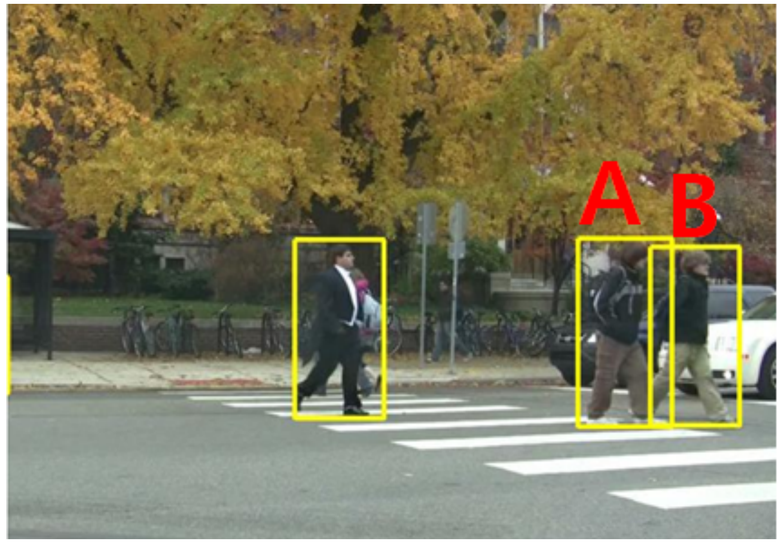
Why



Efficiency

Provide additional supervision signal by enforcing the predictions to satisfy certain constraints at training time

Why: Guiding the Training



Task: Understand the activity of a group of actors in a video

Knowledge: If B is close to A, then B most likely do the same activity with A

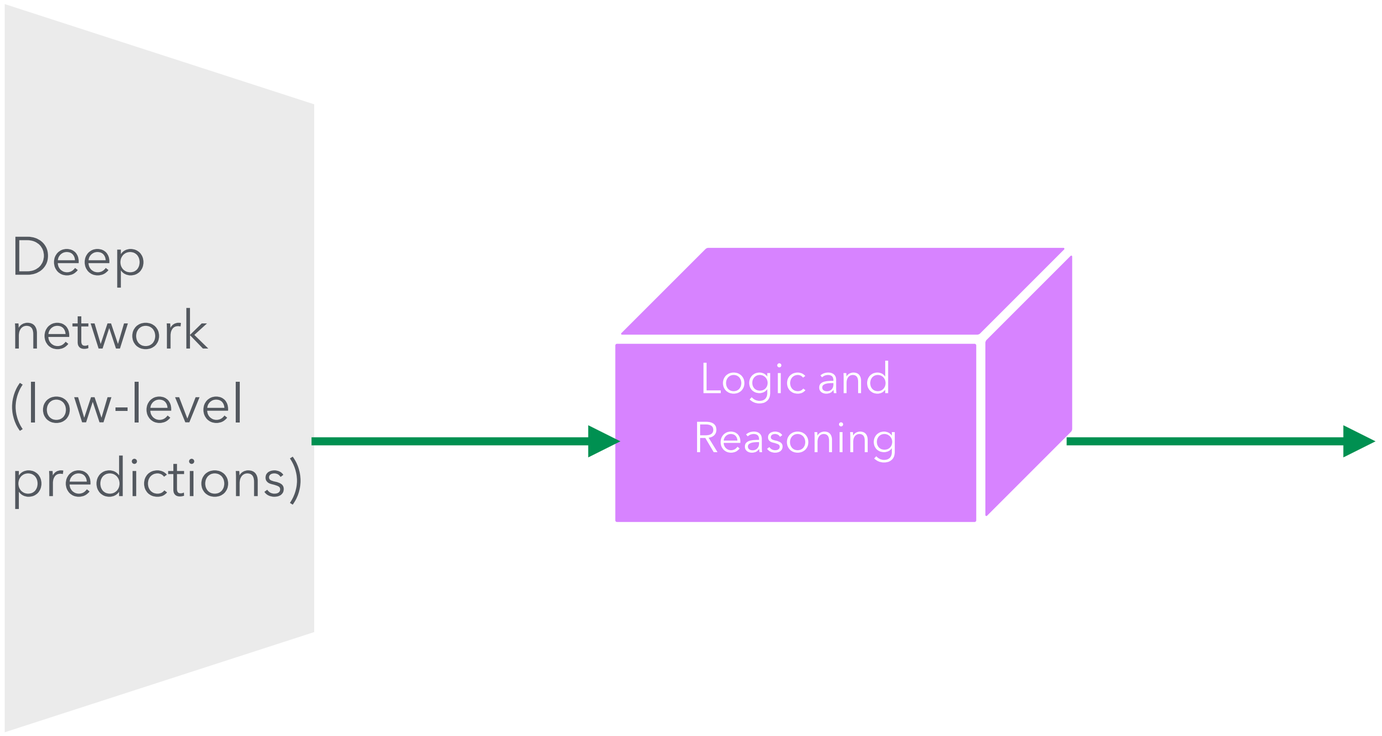
Logic representation:

$$\text{DOING}(A, \text{activity}) \wedge \text{CLOSE}(A,B) \xrightarrow{0.75} \text{DOING}(B, \text{activity})$$

We can **penalize the network** whenever the predictions of A and B are different from each other, even when the target activities of A and B are missing

Leon Jonathan Feldstein, Jurcius Modestas and Efthymia Tsamoura. Parallel neurosymbolic integration with Concordia. In ICML, 2023.

Why



Trustworthiness

Enforce the predictions to satisfy certain constraints at testing time

Why: Trustworthiness

Task: Autonomous driving

Knowledge: Stop if you see a pedestrian is in danger

Logic representation:

pedestrian_crossing ∧ car_is_close → pedestrian_is_in_danger

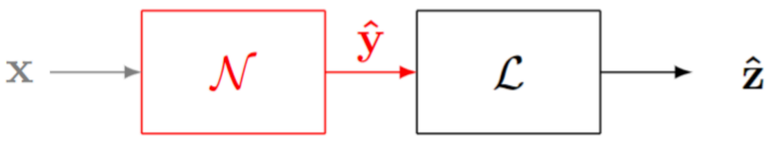
pedestrian_is_in_danger→ stop

Outline of Today's Lecture

- ✓ **Basic notions**
- ✓ **Motivation**
- ✓ **Types of Integration**
- ✓ **Enforcing Boolean Constraints During Training**
  - ✓ **Losses Based on Probabilistic Logic**
  - ✓ **Application in Computer Vision**
  - ✓ **Losses Based on Fuzzy Logic**

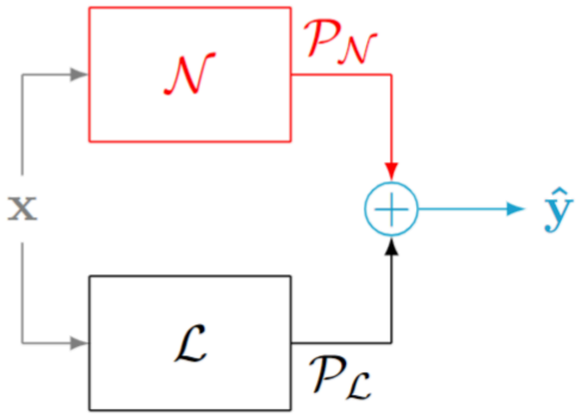
Types of Integration

Stratified



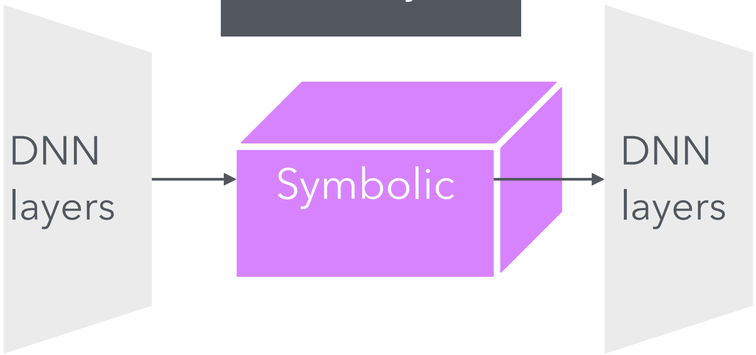
- ✓DeepProbLog [NeurIPS 2018]
- ✓ABL [NeurIPS 2019]
- ✓NeurASP [IJCAI 2020]
- ✓NeuroLog [AAAI 2021]
- ✓Scallop [NeurIPS 2021]
- ✓ENT [ICLR 2023]
- ✓DeepSoftLog [NeurIPS 2023]
- ✓ISED [NeurIPS 2023]
- ✓Dolphin [arXiv 2024]

Teacher-Student



- ✓T-S, ACL [EMNLP 2016]
- ✓DPL [EMNLP 2018]
- ✓Concordia [ICML 2023]

As a layer



- ✓MIPaaL [AAAI 2020]
- ✓BB-backprop [ICLR 2020]
- ✓CombOptNet [ICML 2021]
- ✓SurCo [ICML 2023]
- ✓GenCO [ICML 2024]

(Non-Exhaustive List of) Applications

- ✓Activity detection (Feldstein et al., 2023)
- ✓**Scene graph generation (Buffelli & Tsamoura, 2023)**
- ✓Fine-tuning LLMs (Li et al., 2023)
- ✓Visual question answering (Li et al, 2023, Tsamoura et al., 2023)
- ✓Spatio-temporal scene graph generation (Huang et al., 2025)
- ✓Learning knowledge graph embeddings (Maene & Tsamoura, 2025)

On the Power of Logic

Logic can represent different learning settings

Outline of Today's Lecture

- ✓ **Basic notions**
- ✓ **Motivation**
- ✓ **Types of Integration**
- ✓ **Enforcing Boolean Constraints During Training**
  - ✓ **Losses Based on Probabilistic Logic**
  - ✓ **Application in Computer Vision**
  - ✓ **Losses Based on Fuzzy Logic**

Enforcing Boolean Constraints During Training

Given

✓ a network N

✓ a Boolean formula φ over the predictions of N

Train N so that its predictions adhere to Σ

Example

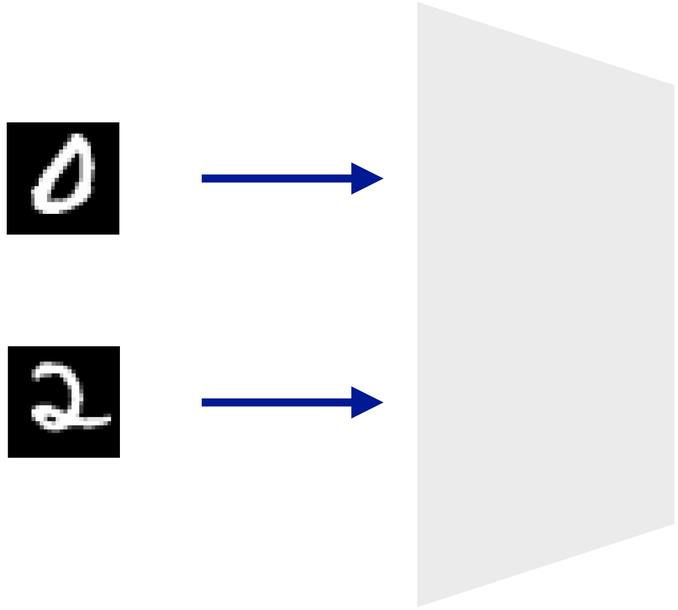
Typical when there are multiple annotators or the classes are difficult to differentiate

Neural network

Goal: Train the DNN knowing that the input shows a **raven or crow or blackbird**

Reduces to: Train the DNN subject to the Boolean constraint $\mathbf{X}_{\text{raven}} \vee \mathbf{X}_{\text{crow}} \vee \mathbf{X}_{\text{blackbird}}$

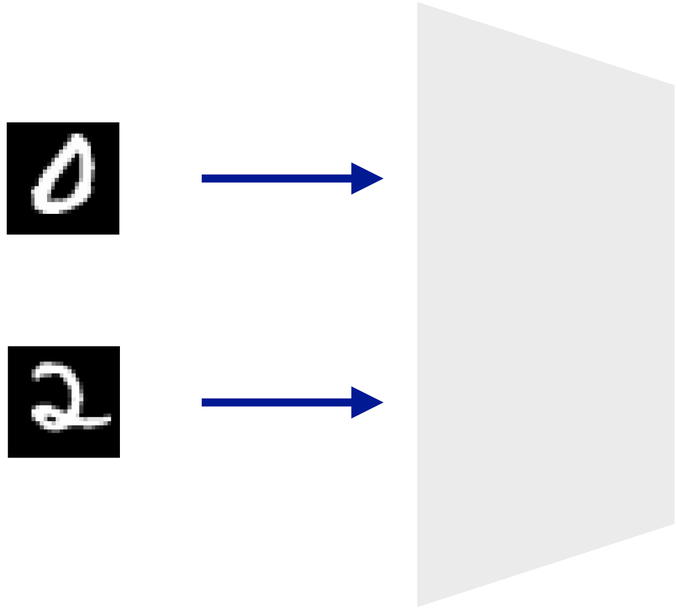
Trickier Example



Goal: Train the DNN knowing that the input MNIST images sum up to **2**. The gold labels are **unknown**

Question: Can you write the Boolean constraint?

Trickier Example



Goal: Train the DNN knowing that the input MNIST images sum up to **2**. The gold labels are **unknown**

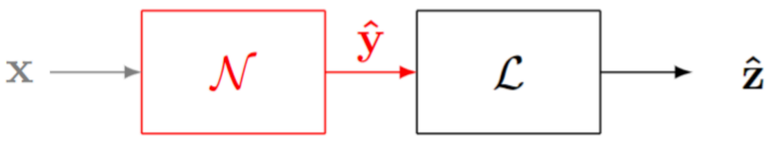
1st image is **1**, 2nd image is **1**

Reduces to: Train the DNN subject to the Boolean constraint $(\mathbf{X}_{1,0} \wedge \mathbf{X}_{2,2}) \vee (\mathbf{X}_{1,1} \wedge \mathbf{X}_{2,1}) \vee (\mathbf{X}_{1,2} \wedge \mathbf{X}_{2,0})$

1st image is **0**, 2nd image is **2**

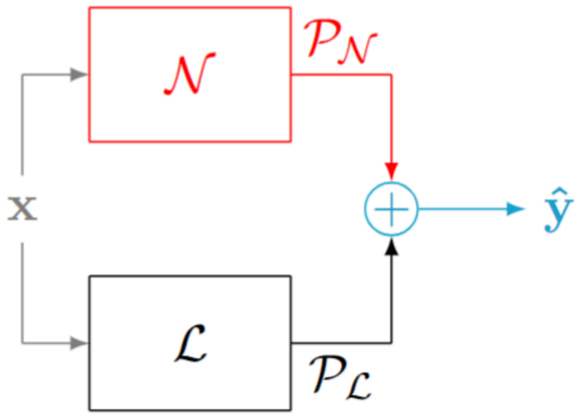
Types of Integration

Stratified



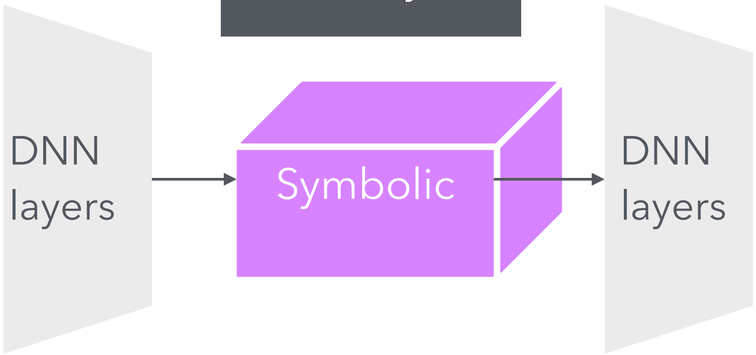
- ✓DeepProbLog [NeurIPS 2018]
- ✓ABL [NeurIPS 2019]
- ✓NeurASP [IJCAI 2020]
- ✓NeuroLog [AAAI 2021]
- ✓Scallop [NeurIPS 2021]
- ✓ENT [ICLR 2023]
- ✓DeepSoftLog [NeurIPS 2023]
- ✓ISED [NeurIPS 2023]
- ✓Dolphin [arXiv 2024]

Teacher-Student



- ✓T-S, ACL [EMNLP 2016]
- ✓DPL [EMNLP 2018]
- ✓Concordia [ICML 2023]

As a layer



- ✓MIPaaL [AAAI 2020]
- ✓BB-backprop [ICLR 2020]
- ✓CombOptNet [ICML 2021]
- ✓SurCo [ICML 2023]
- ✓GenCO [ICML 2024]

RoadMap

- ✓Boolean formulas – basic syntax
- ✓Boolean satisfiability
- ✓Boolean formulas & probability theory
- ✓Logical loss

Boolean Formulas

Expressions composed over

- ✓Variables in {0,1} – {True, False}
- ✓Logical connectives
  - ✓AND $\wedge$
  - ✓OR $\vee$
  - ✓NOT $\neg$

Boolean Formulas

Example

$\phi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

Boolean Satisfiability

In **classical logic**, we seek to answer whether the formula can become true for some variable assignment.

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

X	Y	Z	φ
FALSE	FALSE	FALSE	FALSE
FALSE	FALSE	TRUE	FALSE
FALSE	TRUE	FALSE	FALSE
FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	FALSE
TRUE	FALSE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE

Boolean Formulas & Probability Theory

Question: What if each Boolean variable **X** becomes true with some **probability** π**(X)** and the Boolean variables are **independent**?

Boolean Formulas

Example

$\phi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

X becomes **true** with probability **0.3** and **false** with probability **0.7**

Y becomes **true** with probability **0.4** and **false** with probability **0.6**

Z becomes **true** with probability **0.9** and **false** with probability **0.1**

Boolean Formulas & Probability Theory

Question: What if each Boolean variable **X** becomes true with some **probability** π**(X)** and the Boolean variables are **independent**?

What **semantics** should we give to satisfiability?

Boolean Satisfiability

In **classical logic**, we seek to answer whether the formula can become true for some variable assignment.

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

X	Y	Z	φ
FALSE	FALSE	FALSE	FALSE
FALSE	FALSE	TRUE	FALSE
FALSE	TRUE	FALSE	FALSE
FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	FALSE
TRUE	FALSE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE

Probabilities of Formulas

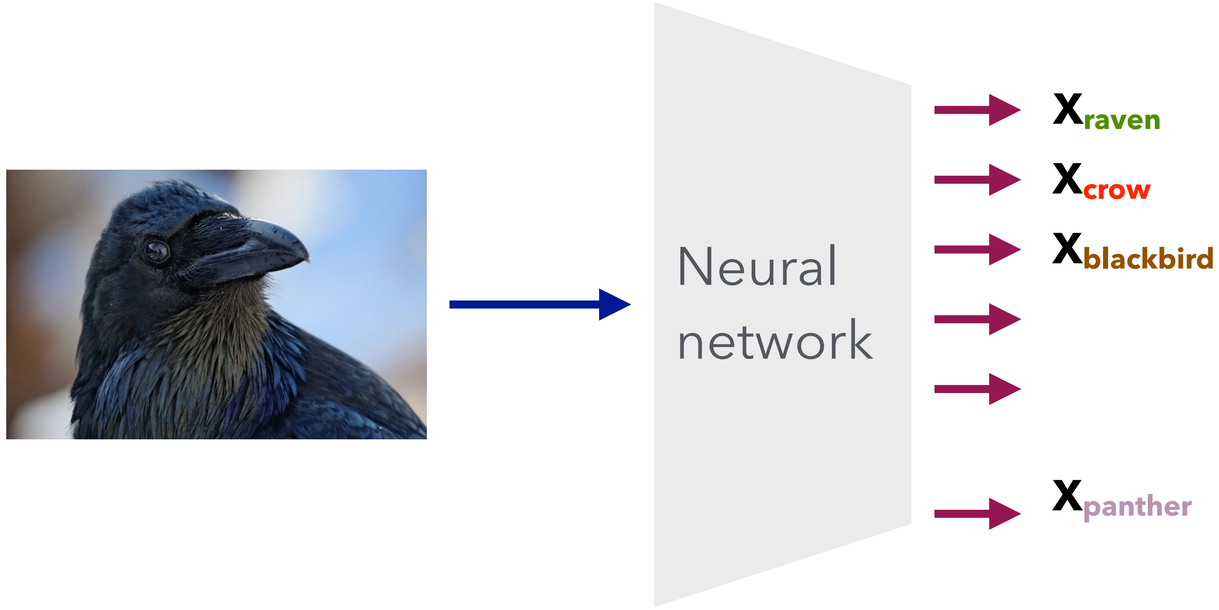
Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

X	Y	Z	φ	Probabilities of assignments
FALSE	FALSE	FALSE	FALSE	
FALSE	FALSE	TRUE	FALSE	
FALSE	TRUE	FALSE	FALSE	
FALSE	TRUE	TRUE	TRUE	$(1 - \pi(\mathbf{X}))\pi(\mathbf{Y})\pi(\mathbf{Z})$
TRUE	FALSE	FALSE	FALSE	
TRUE	FALSE	TRUE	TRUE	$\pi(\mathbf{X})(1 - \pi(\mathbf{Y}))\pi(\mathbf{Z})$
TRUE	TRUE	FALSE	TRUE	$\pi(\mathbf{X})\pi(\mathbf{Y})(1 - \pi(\mathbf{Z}))$
TRUE	TRUE	TRUE	TRUE	$\pi(\mathbf{X})\pi(\mathbf{Y})\pi(\mathbf{Z})$

Semantics: The probability $P(\varphi)$ is the probability of drawing a variable assignment in which φ becomes true

The probability of each variable assignment is the product of the probabilities the variables are true or false

Training DNNs Under Boolean Constraints



Goal: Train the DNN knowing that the input shows a **raven or crow or blackbird**

Reduces to: Train the DNN subject to the Boolean constraint $\mathbf{x}_{raven} \vee \mathbf{x}_{crow} \vee \mathbf{x}_{blackbird}$

On Computing Probabilities

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

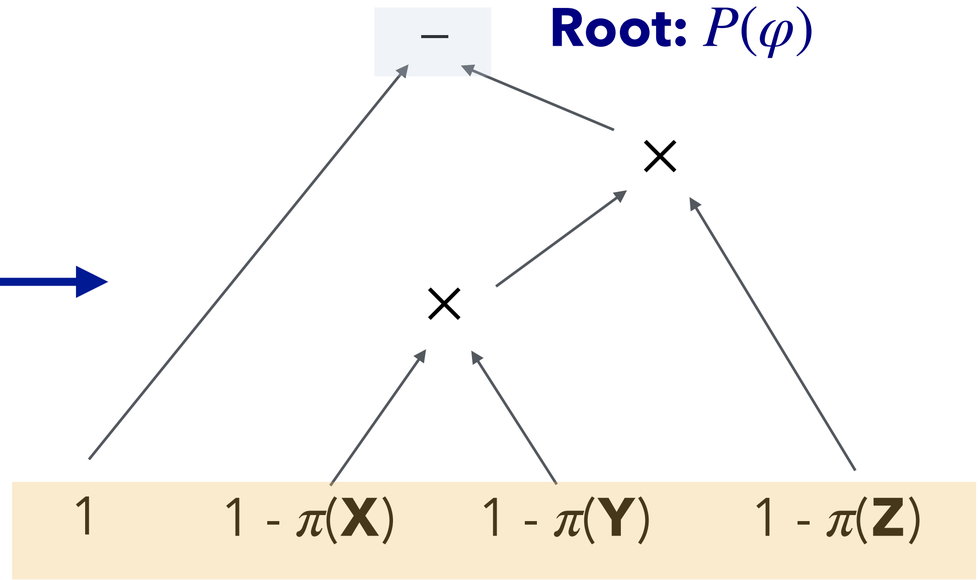
X	Y	Z	φ	Probabilities of assignments
FALSE	FALSE	FALSE	FALSE	
FALSE	FALSE	TRUE	FALSE	
FALSE	TRUE	FALSE	FALSE	
FALSE	TRUE	TRUE	TRUE	$(1 - \pi(\mathbf{X}))\pi(\mathbf{Y})\pi(\mathbf{Z})$
TRUE	FALSE	FALSE	FALSE	
TRUE	FALSE	TRUE	TRUE	$\pi(\mathbf{X})(1 - \pi(\mathbf{Y}))\pi(\mathbf{Z})$
TRUE	TRUE	FALSE	TRUE	$\pi(\mathbf{X})\pi(\mathbf{Y})(1 - \pi(\mathbf{Z}))$
TRUE	TRUE	TRUE	TRUE	$\pi(\mathbf{X})\pi(\mathbf{Y})\pi(\mathbf{Z})$

Enumerating the variable assignments is not the way we compute the probability

On Computing Probabilities

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ → Circuit construction →



Leaves: Variable probabilities

These techniques will explained later!

On Computing Probabilities

- ✓#P-hard already for positive bipartite formulas in disjunctive normal form (Provan & Ball, 1983)
- ✓Counting the number of satisfying instantiations is #P-complete (Valiant, 1979)
- ✓Exact probabilistic inference is #P-complete (Roth 1996)
- ✓PTIME if formula is in specific forms, e.g., each variable occurs only once
- ✓Multiple polynomial-time approximations (Fink et al., 2013, Meel et al., 2019, Soos et al., 2023)
- ✓Multiple exact and approximate techniques that compute probabilities by translating the formula into a circuit (i.e., an arithmetic function over plus, minus, times operations) (Darwiche & Marquis, 2002, see other work from Darwiche)
- ✓Good read (Suciu et al., 2011)

On Circuit Construction

$$P(\neg\Phi) = 1 - P(\Phi)$$

Formulas Φ, Ψ don't share variables

$$P(\Phi \wedge \Psi) = P(\Phi) \cdot P(\Psi)$$

$$P(\Phi \vee \Psi) = 1 - (1 - P(\Phi)) \cdot (1 - P(\Psi))$$

Formulas Φ, Ψ cannot become true at the same time (inconsistent)

$$P(\Phi \wedge \Psi) = 0$$

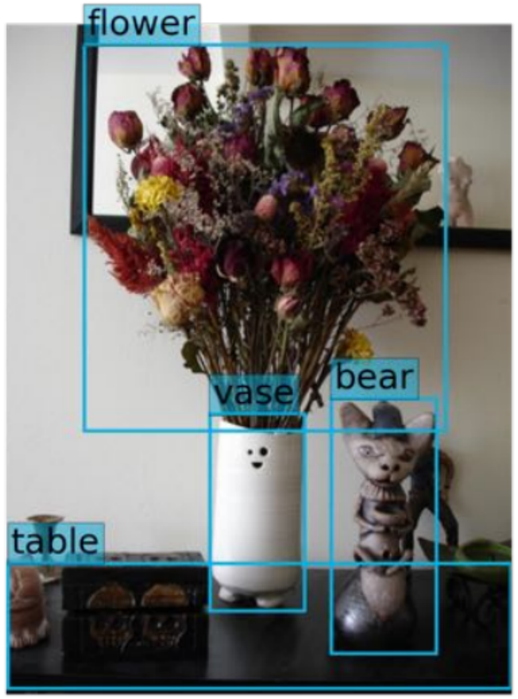
$$P(\Phi \vee \Psi) = P(\Phi) + P(\Psi)$$

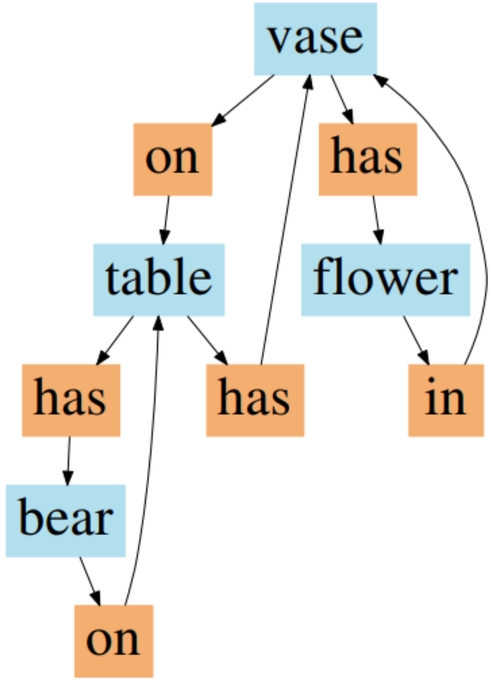
Techniques rely on making formulas inconsistent via Shannon decomposition and then apply these rules

Outline of Today's Lecture

- ✓ **Basic notions**
- ✓ **Motivation**
- ✓ **Types of Integration**
- ✓ **Enforcing Boolean Constraints During Training**
  - ✓ **Losses Based on Probabilistic Logic**
  - ✓ **Application in Computer Vision**
  - ✓ **Losses Based on Fuzzy Logic**

Applications in Computer Vision

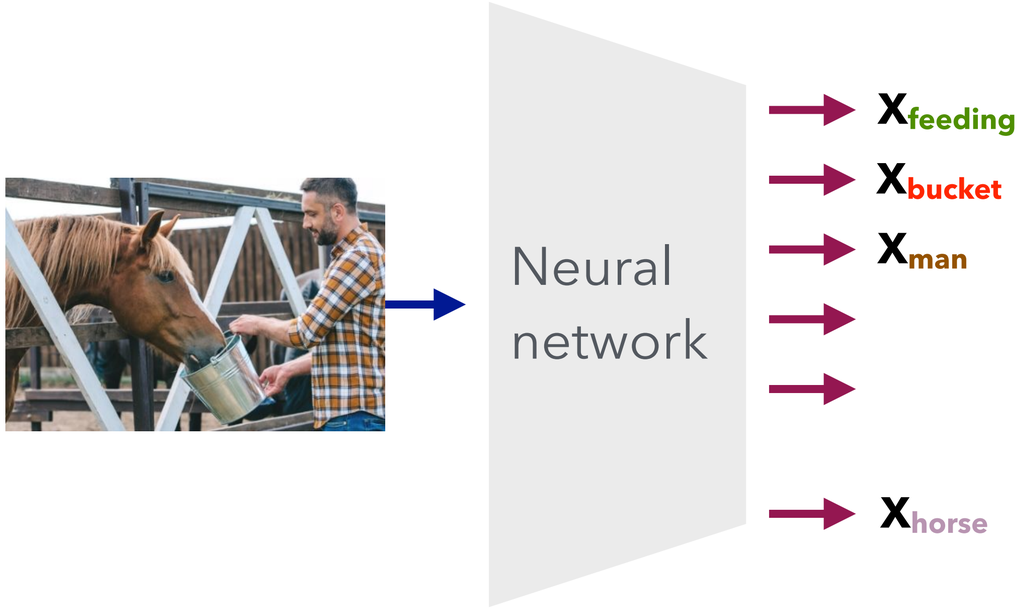




Task: Extract the scene graph from an input image

Davide Buffelli, and Efthymia Tsamoura. Scalable Theory-Driven Regularization of Scene Graph Generation Models. In AAAI, 2023.

Logic in Computer Vision



Goal: Penalize the outputs of the network whenever the output facts violate the background knowledge

Knowledge: Integrity constraints: ¬predicate(subject, object)

Examples:

¬feeding(bucket, man)

¬feeding(man, bucket)

Logic in Computer Vision

Neural network

$x_{feeding}$

x_{bucket}

x_{man}

x_{horse}

Integrity constraints

$\neg feeding(bucket, man)$

$\neg feeding(man, bucket)$

Boolean constraints

$\phi_1 = \neg(x_{feeding} \wedge x_{bucket} \wedge x_{man})$

$\phi_2 = \neg(x_{feeding} \wedge x_{man} \wedge x_{bucket})$

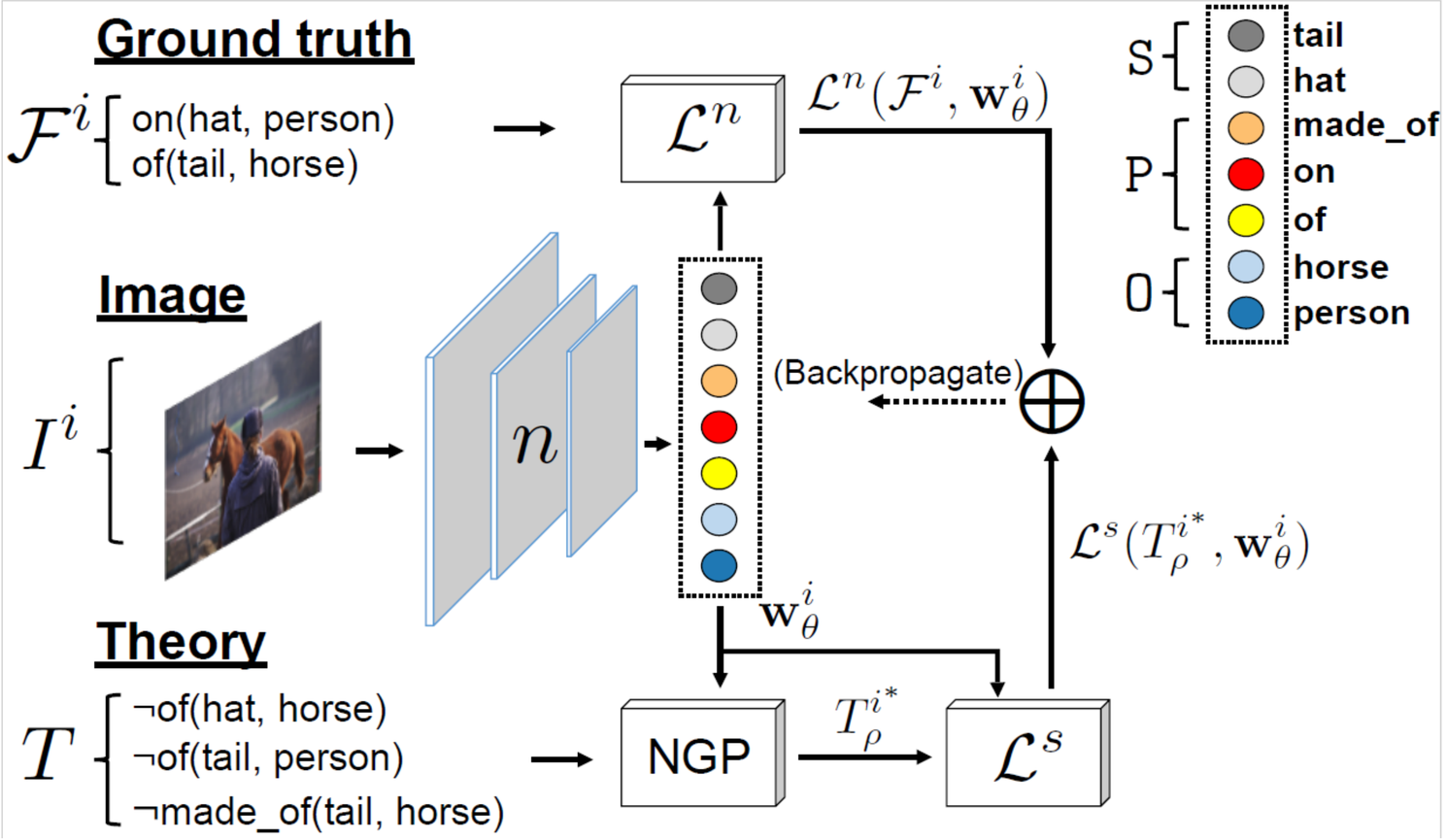
Probability computation

$$P(\phi_1 \wedge \phi_2)$$

Loss computation

$$-\log P(\phi_1 \wedge \phi_2)$$

Pipeline


Ground truth
$\mathcal{F}^i$
on(hat, person)
of(tail, horse)
$\mathcal{L}^n$
$\mathcal{L}^n(\mathcal{F}^i, \mathbf{w}_\theta^i)$
S
P
O
tail
hat
made_of
on
of
horse
person
Image
I^i
n
(Backpropagate)
$\mathcal{L}^s(T_\rho^{i*}, \mathbf{w}_\theta^i)$
$\mathbf{w}_\theta^i$
Theory
T
¬of(hat, horse)
¬of(tail, person)
¬made_of(tail, horse)
NGP
T_ρ^{i*}
$\mathcal{L}^s$

Benefits

- ✓ Allows to design a technique that **scales to hundreds of thousands of ICs**
  - ✓ The weights of the neural network are updated to minimize the maximum violation of the ICs
- ✓ **Easy for users** to (semi-)automatically create such ICs from existing knowledge bases or even from the training data itself

Pseudocode

Algorithm 1: NGP($I, \mathcal{F}, T, n_t$) $\rightarrow n_{t+1}$

$\mathbf{w} := n_t(I)$ Input image

% Find the ICs that maximally violate the network's predictions

$T^*_\rho := arg \max_{T_\rho \subseteq T} \mathcal{L}^s(T_\rho, \mathbf{w})$

$\ell := \beta_1 \cdot \mathcal{L}^n(\mathcal{F}, \mathbf{w}) + \beta_2 \cdot \mathcal{L}^s(T^*_\rho, \mathbf{w})$

% Logic

$n_{t+1} :=$ backpropagate$(n_t, \bigtriangledown \ell)$

return n_{t+1}

Note: β_1, β_2 and ρ are hyperparameters.

Algorithm 2: GREEDY(I, ρ, T, n_t) $\rightarrow T^*$

$\mathbf{w} := n_t(I)$ $T^* := \emptyset$ $j := 1$

while $|T^*| < \rho$ **do**

  get the j-th $\mathtt{p(s,o)}$ prediction maximizing $w(\mathtt{p}) \cdot w(\mathtt{s}) \cdot w(\mathtt{o})$

  if $\neg\mathtt{p(s,o)}$ is in T, **then add** $\neg\mathtt{p(s,o)}$ to T^*

  $j := j + 1$

end while

return T^*

Empirical Results

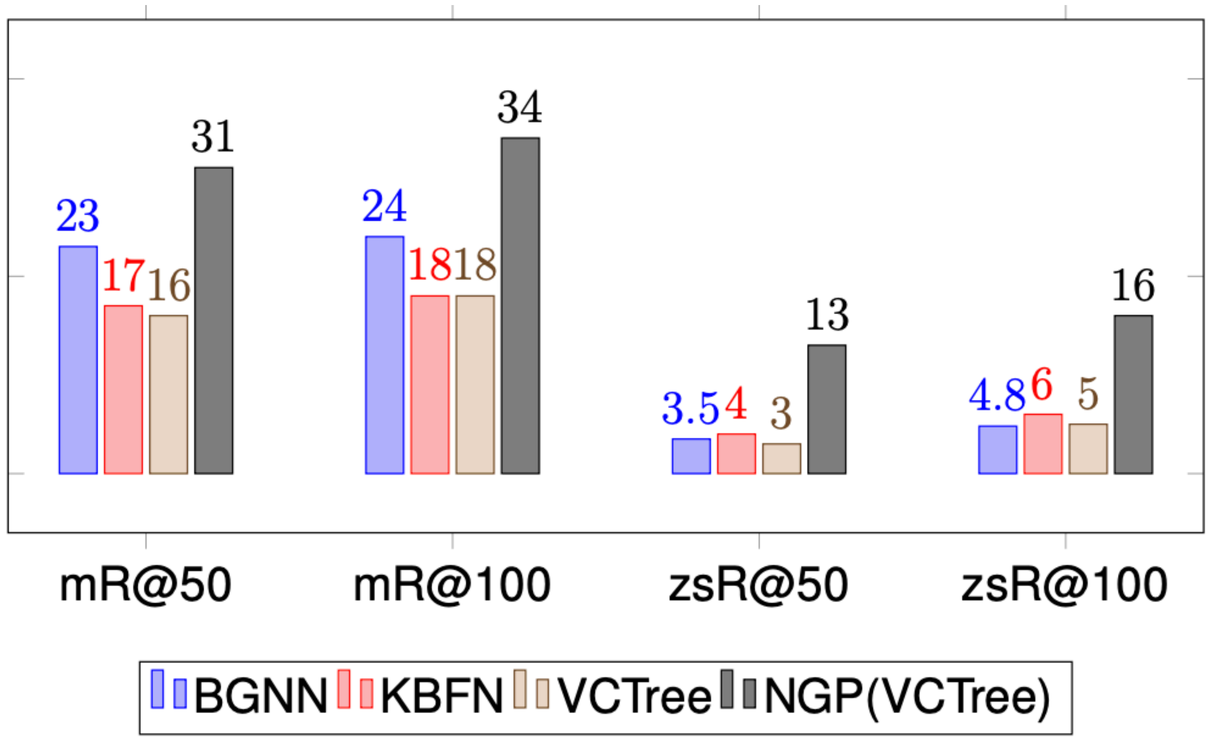
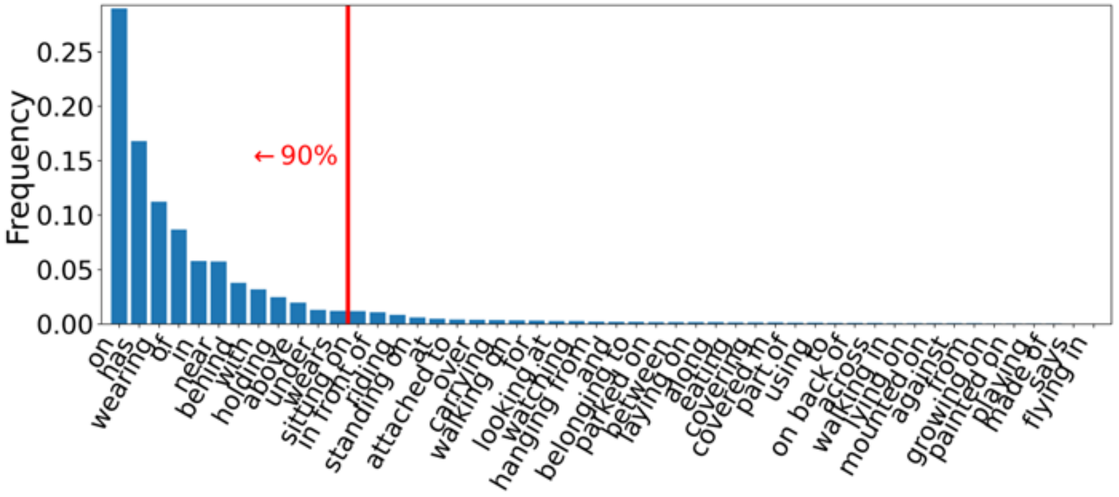


Figure. Mean recall @ k of different models: BGNN (Li et al. 2021), KBFN Gu et al. 2019), VCTree (Tang et al. 2019). NGP(VCTree): Our proposal.

Benchmark. Visual Genome

Empirical Results



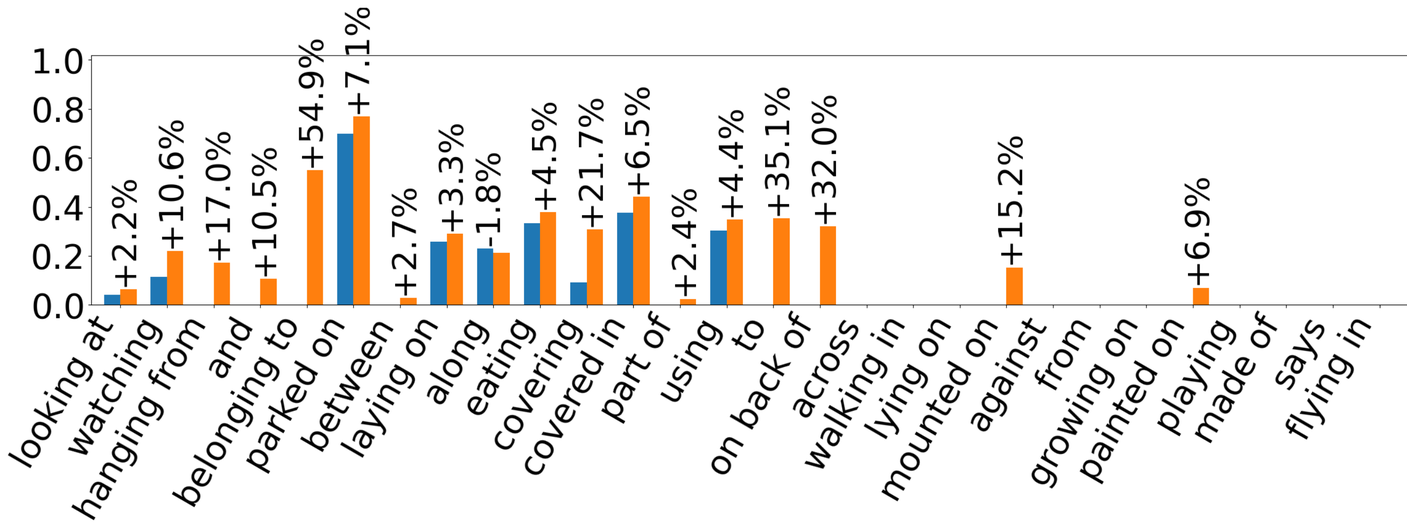
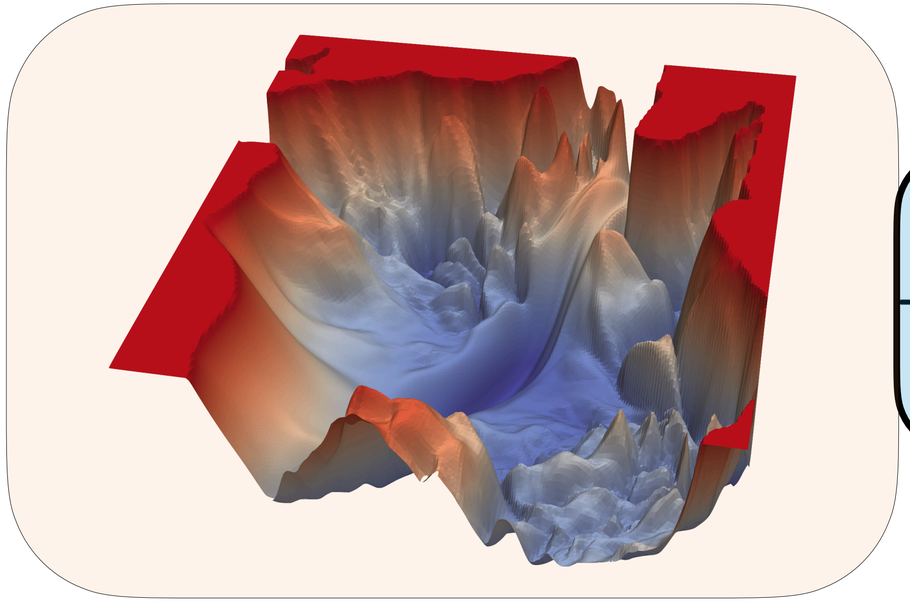
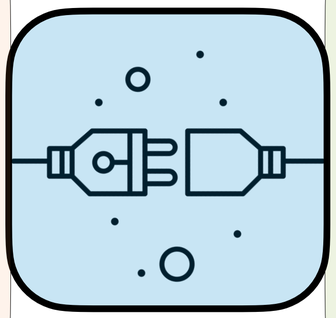
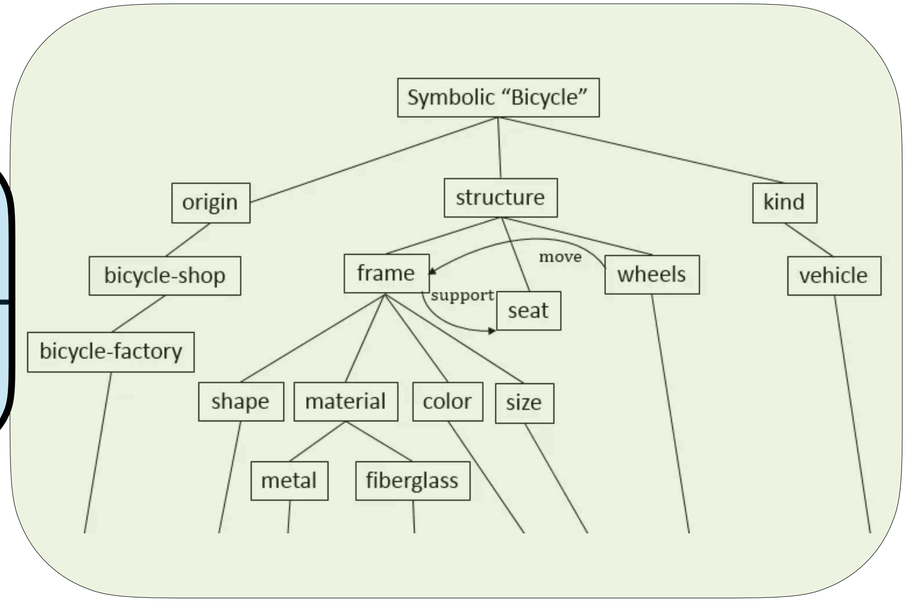


Table. Recall of VCTree (CVPR 2019) on the **28 least frequent** classes: **without logic**; **with logic**.

Benchmark. Visual Genome







Differentiable

Logics!

Neural networks:

Continuous,

Trained with backpropagation

Symbolic AI:

Discrete

Differentiable Logics

- ✓ How to **create a differentiable loss function** for Boolean formulas?
- ✓ Previous: **Probability of formula**
  - ✓ Theoretically well-motivated (later lectures)
  - ✓ But **computationally expensive**
- ✓ Next: **Fuzzy truth value of formula**
  - ✓ Theoretically a bit shaky!
  - ✓ But **computationally** very nice

Neural Network
Predict classes on image

red green

$\neg\mathrm{red} \vee \neg\mathrm{green}$

Differentiable Logic
Enforce Boolean formula

Boolean Formulas as fuzzy values

Example

$\phi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

X is **0.3 true**, meaning: A little bit true

Y is **0.4 true**, which is a bit more true

Z is 0.9 **true**, which is almost completely true

Probability vs fuzzy

- **Probability of X is π(X)=0.3**
  - **X** is true 30% of the time
  - Example: A die has 50% probability to land on an even number
- **Fuzzy truth value of X π(X)**
  - **X** is a bit true
  - Example: John is tall is somewhat true

Boolean Formulas & Fuzziness

Question: What if each Boolean variable **X** has a **fuzzy truth value π(X) in [0, 1]**?

What **semantics** should we give to satisfiability of the formulas?

Fuzzy logic operators

- ✓ Fuzzy logics give **new meaning** to logical connectives
- ✓ Example: The **Gödel logic**
  - ✓ AND $\mathbf{X} \wedge \mathbf{Y}$
    - ✓ $\mathbf{min(\pi(X), \pi(Y))}$
  - ✓ OR $\mathbf{X} \vee \mathbf{Y}$
    - ✓ $\mathbf{max(\pi(X), \pi(Y))}$
  - ✓ NOT $\neg\mathbf{X}$
    - ✓ $\mathbf{1-\pi(X)}$

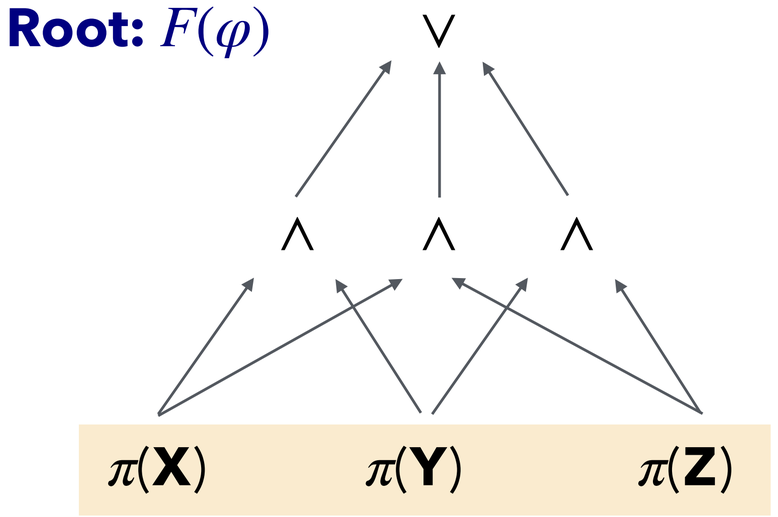
Gödel logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

Replace connectives with min / max

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →

Neural network →



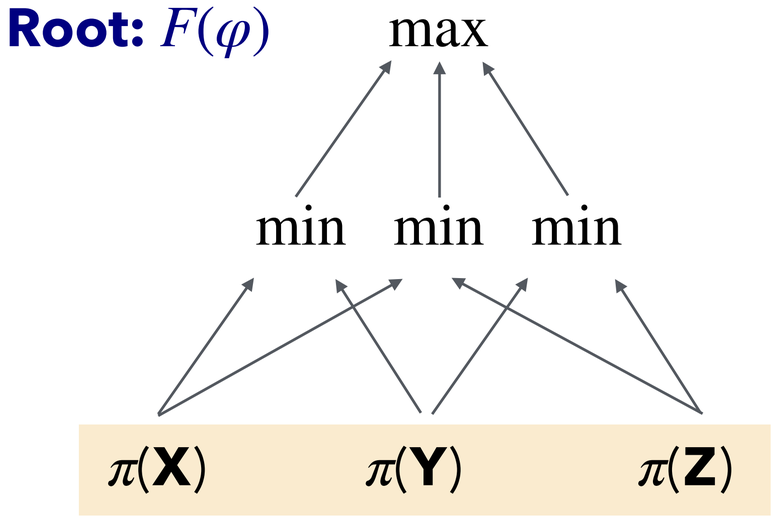
Leaves: fuzzy truth values from neural network

Gödel logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →

Neural network →



Leaves: fuzzy truth values from neural network

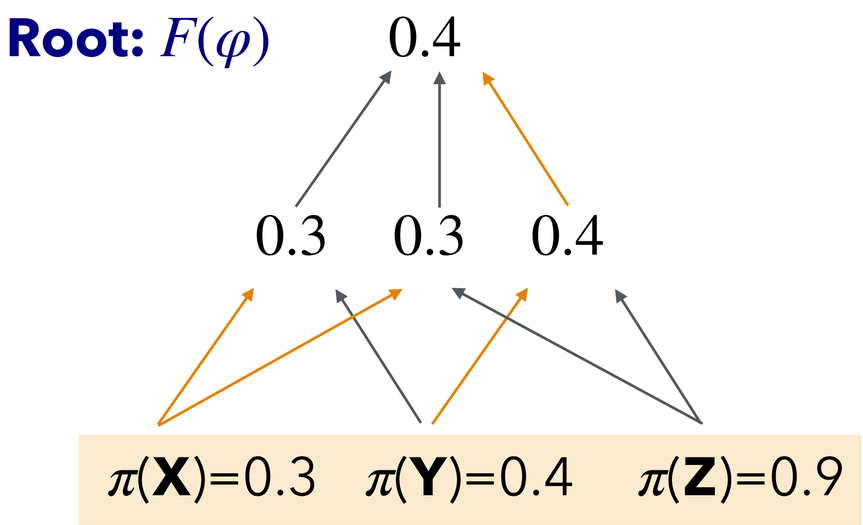
Gödel logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

Compute truth values by traversing tree

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →

Neural network →



Leaves: fuzzy truth values from neural network

Gödel logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

Backpropagate over activated tree to increase $\pi(\mathbf{Y})$

Root: $F(\varphi)$ 0.4

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →

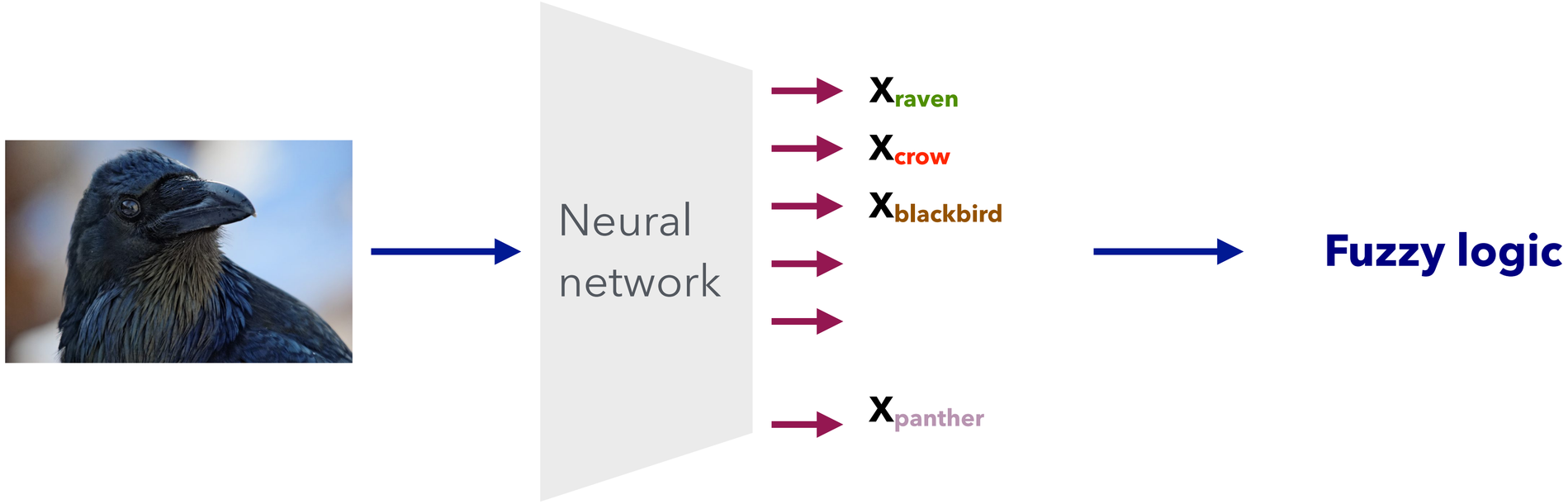
0.3 0.3 0.4

Neural network ←

$\pi(\mathbf{X})=0.3$ $\pi(\mathbf{Y})=0.4$ $\pi(\mathbf{Z})=0.9$

Leaves: fuzzy truth values from neural network

Training DNNs Under Boolean Constraints



Goal: Train the DNN knowing that the input shows a **raven or crow or blackbird**

Reduces to: Train the DNN subject to the Boolean constraint $x_{raven} \vee x_{crow} \vee x_{blackbird}$

Fuzzy logic operators

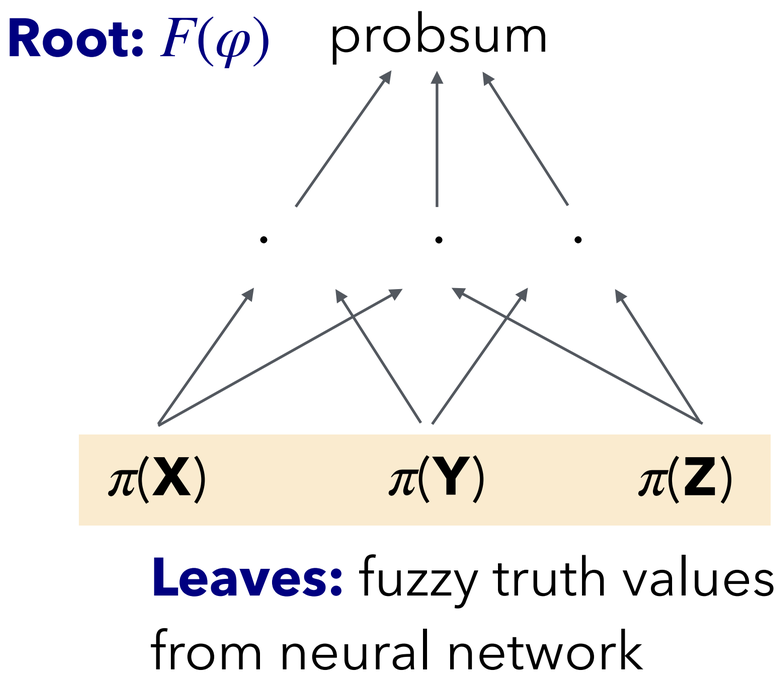
- ✓ Example 2: The **Product logic**
  - ✓ AND $\mathbf{X} \wedge \mathbf{Y}$
    - ✓ $\pi(\mathbf{X}) \cdot \pi(\mathbf{Y})$ (probability two independent events are both true)
  - ✓ OR $\mathbf{X} \vee \mathbf{Y}$
    - ✓ $\pi(\mathbf{X}) + \pi(\mathbf{Y}) - \pi(\mathbf{X}) \cdot \pi(\mathbf{Y})$ (probability at least one of two independent events is true)
  - ✓ NOT $\neg\mathbf{X}$
    - ✓ $1-\pi(\mathbf{X})$

Product logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →

Neural network →



Leaves: fuzzy truth values from neural network

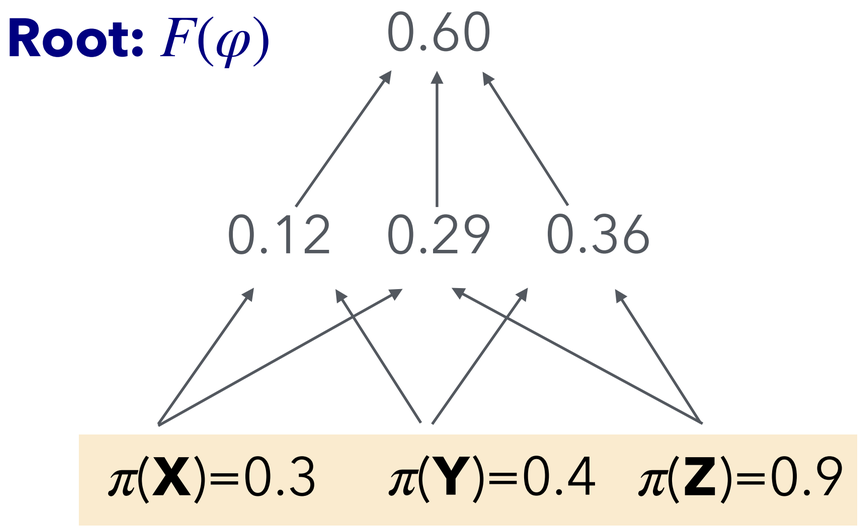
Product logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

$$1 - (1 - 0.12) \cdot (1 - 0.29) \cdot (1 - 0.36)$$

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →

Neural network →



Leaves: fuzzy truth values from neural network

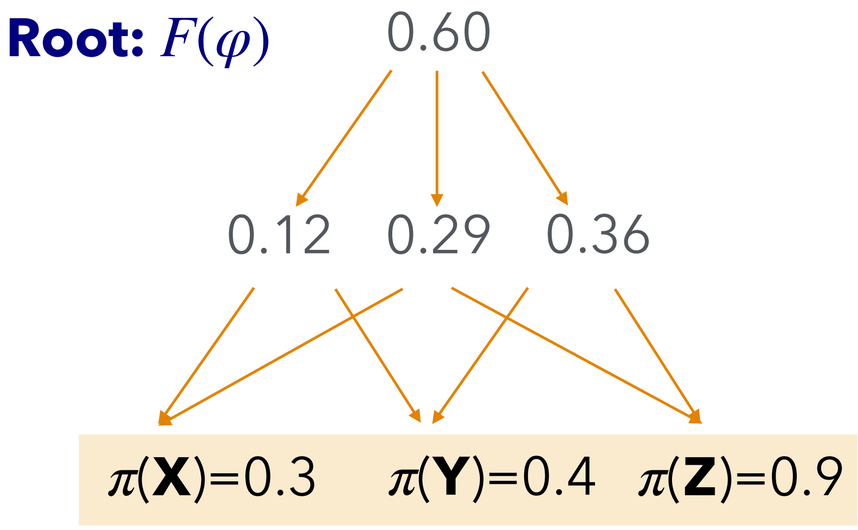
Product logic example

Example: Consider the formula $\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$

Backpropagates into all leaves

$$1 - (1 - 0.12) \cdot (1 - 0.29) \cdot (1 - 0.36)$$

$\varphi = \mathbf{X} \wedge \mathbf{Y} \vee \mathbf{Y} \wedge \mathbf{Z} \vee \mathbf{X} \wedge \mathbf{Z}$ →



← Neural network

Leaves: fuzzy truth values from neural network

Name	Conjunction
Gödel / minimum	$\min(a, b)$
Product	$a \cdot b$
Łukasiewicz	$\max(a + b - 1, 0)$

The basic fuzzy conjunctions *(t-norms)*…

How to choose?

van Krieken, Emile, Erman Acar, and Frank van Harmelen. "Analyzing differentiable fuzzy logic operators." Artificial Intelligence 302 (2022): 103602

Product t-norm

$$X \Rightarrow \pi(X)$$

$$X \wedge X \Rightarrow \pi(X) \cdot \pi(X)$$

Classical equivalences don't hold...

Equivalent formulas can create different loss

Lukasiewicz disjunction (*t-conorm*)

$$\bigvee_{i=1}^{1000} X_i \Rightarrow \min\left(\sum_{i=1}^{1000} \pi(X_i), 1\right)$$

Equals 1 if average probability $\geqslant \frac{1}{1000}$

Unexpected semantics…

Usually no gradients!

Use of fuzzy methods

- ✓ In Neurosymbolic AI, fuzzy losses are **very frequently used**
  - ✓ Most common framework: Logic Tensor Networks (LTN) (Serafini 2016, Badreddine 2022)
  - ✓ Has PyTorch and Tensorflow implementations
- ✓ Common choice in applications: Simple to implement
- ✓ **Computationally efficient:**
  - ✓ Just traverses formula, so linear in formula size!

https://github.com/tommasocarraro/LTNtorch

Recap of Today's Lecture

- ✓ **Basic notions**
- ✓ **Motivation**
- ✓ **Types of Integration**
- ✓ **Enforcing Boolean Constraints During Training**
  - ✓ **Losses Based on Probabilistic Logic**
  - ✓ **Application in Computer Vision**
  - ✓ **Losses Based on Fuzzy Logic**