

Probabilistic Concept Bottleneck Models



AI WITH NOTHING TO HIDE

Giuseppe Marra & Pietro Barbiero & David Debot

giuseppe.marra@kuleuven.be

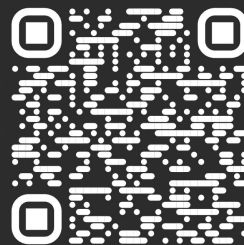
pietro.barbiero@ibm.com

david.debot@kuleuven.be

In collaboration with: Giovanni de Felice and Mateo Espinosa Zarlenga

Tutorial

uai2026



Who are we?



Giuseppe Marra

Assistant Professor

KU Leuven

giuseppe.marra@kuleuven.be

KU LEUVEN



Pietro Barbiero

Swiss Postdoctoral Fellow

IBM Research (Switzerland)

pietro.barbiero@ibm.com

IBM Research



David Debot

PhD Candidate

KU Leuven

david.debot@kuleuven.be

KU LEUVEN



Giovanni De Felice

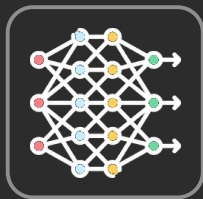


Mateo Espinosa Zarlenga



The problem

When is f interpretable for the **target user**?



Machine
learning model



Target
user

The solution

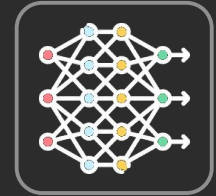
**What counts as
interpretable for
the user X?**

**The Standard
Interpretable Model**

**Interpretable &
accurate model**



Target
user



Machine
learning model

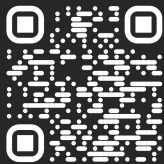
What is this course NOT about?

We will not have time to dive deep into:

1. “Traditional” explainable AI (XAI) methodologies
2. Deep philosophical aspects of explaining models
3. Post-hoc interpretability such as Mechanistic Interpretability



“Interpretable Machine Learning”
Christoph Molnar



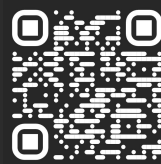
“The Mythos of Interpretability”
Lipton (2018) [1]



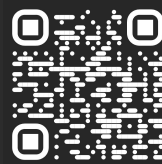
“To Explain or to Predict?”
Shmueli (2018) [2]



“Explanation Theory”
Bromberger (1992) [3]



Distill Circuits Thread



Anthropic Circuits Thread

[1] Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[2] Lipton. “The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.” *Queue* (2018).

[3] Shmueli. “To Explain or to Predict?.” *Statist. Sci.* 25 (3) 289 - 310, August 2010. <https://doi.org/10.1214/10-STS330>

[4] Bromberger. *On what we know we don't know: Explanation, theory, linguistics, and how questions shape them*. University of Chicago Press, 1992.

[5] Cammarata et al. “Thread: circuits.” *Distill* 5.3 (2020): e24.

[6] Anthropic “Transformer Circuits Thread” found at <https://transformer-circuits.pub/>

Tutorial Outline

I. Interpretability theory



~50 mins

II. Concept representations



~40 mins

III. Task predictions



~50 mins

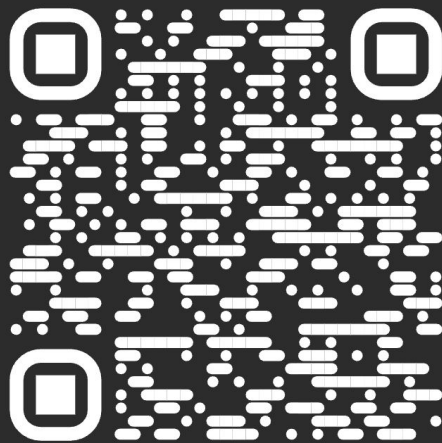
IV. PyTorch Concepts



~20 mins

Course Website and Materials

This course's slides, schedule, and materials are all available at our website:



interpretabledeeplearning.github.io

Introduction to interpretability

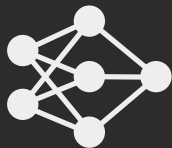


Why do we need interpretability?



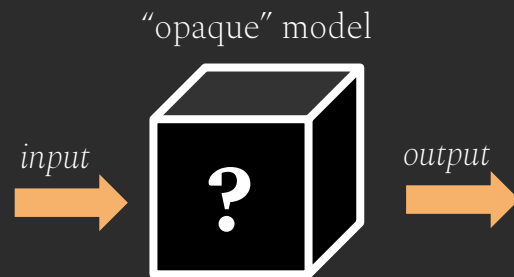
How is interpretability defined?

Why do we need interpretability in the first place?

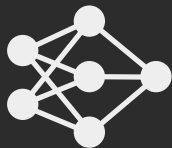


- Highly parametric
- Complicated inference steps
- Non-linearities
- Sensitive to initial states and update rules

$\approx$

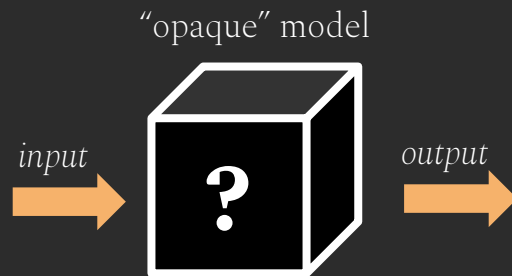


Why do we need interpretability in the first place?



- Highly parametric
- Complicated inference steps
- Non-linearities
- Sensitive to initial states and update rules

$\approx$



1. Blindly using opaque models can lead to all sorts of problems

Why Amazon's Automated Hiring Tool Discriminated Against Women

Predictive policing algorithms are racist. They need to be dismantled.

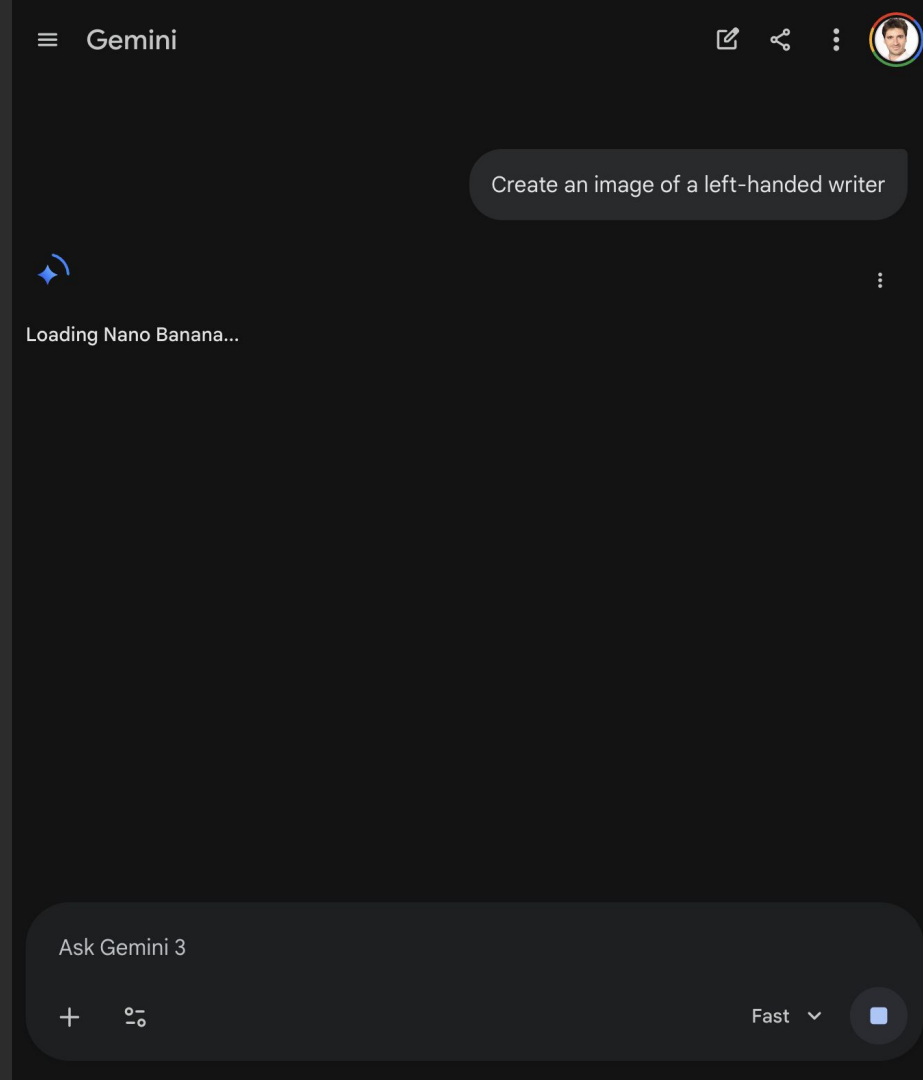
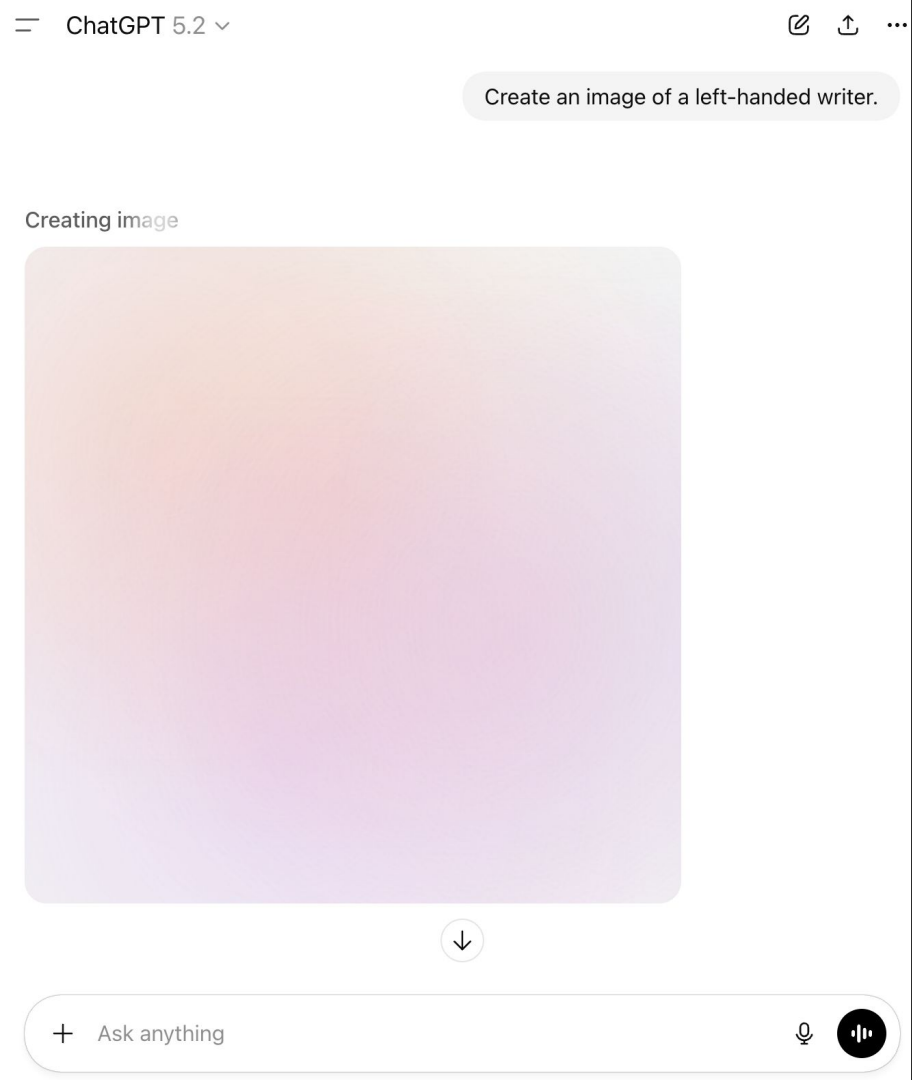
Wrongfully Accused by an Algorithm

In what may be the first known case of its kind, a faulty facial recognition match led to a Michigan man's arrest for a crime he did not commit.

[1] Kashmir Hill, "Wrongfully Accused by an Algorithm." The New York Times (2020).

[2] Rachel Goodman, "Why Amazon's Automated Hiring Tool Discriminated Against Women." ACLU (2018).

[3] Will Douglas Heaven, "Predictive policing algorithms are racist. They need to be dismantled." MIT Technology Review (2020).



Create an image of a left-handed writer.

Image created • Left-handed writing at a wooden desk



+ Ask anything



Gemini

Create an image of a left-handed writer



Ask Gemini 3



Fast

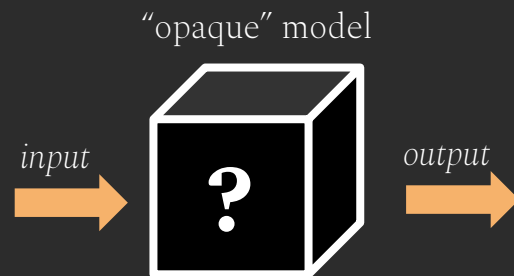


Why do we need interpretability in the first place?



- Highly parametric
- Complicated inference steps
- Non-linearities
- Sensitive to initial states and update rules

$\approx$



1. **Blindly using opaque models** can lead to all sorts of **problems**
2. **Regulatory/legal constraints:**

General Data Protection Regulations (GDPR, 2016)

“The data subject shall have the right not to be subject to a **decision** based solely on **automated processing**, including profiling,...” (Art. 22)

The data subject has the right to “**meaningful information** about the **logic** involved” in the decision. (Art. 13 and 15)

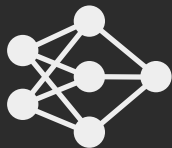
EU AI Act (2024):

“Any affected person subject to a **decision** which is taken by... a **high-risk AI system** ... shall have the right to obtain from the deployer **clear and meaningful explanations** (Art. 86)

[1] GDPR, EU. "Automated individual decision-making, including profiling." (2022).

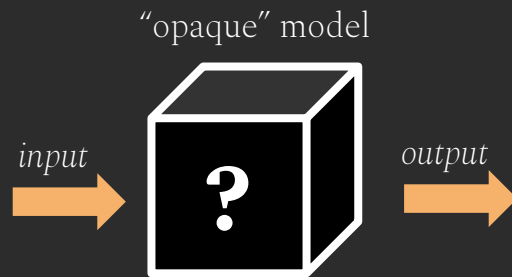
[2] Act, EU Artificial Intelligence. "The EU Artificial Intelligence Act." (2024).

Why do we need interpretability in the first place?



- Highly parametric
- Complicated inference steps
- Non-linearities
- Sensitive to initial states and update rules

$\approx$



1. **Blindly using opaque models** can lead to all sorts of **problems**
2. **Regulatory/legal** constraints
3. **Human-AI interactions** are enabled by understanding how a model reasons

But what does interpretability really mean?

Welcome to the Wild West of interpretability terminology



*“A method is interpretable **if a user can** correctly and efficiently **predict** the method’s **results**” [1]*

*“Systems are interpretable **if their operations can be understood** by a human” [2]*

*“Interpretability is the degree to which an observer can **understand the cause of a decision**” [3]*

*“There is **no universal, mathematical definition** of interpretability, and there never will be” [4]*



Informal



Not actionable

[1] Kim et al. “Examples are not enough, learn to criticize! criticism for interpretability” NeurIPS (2016).

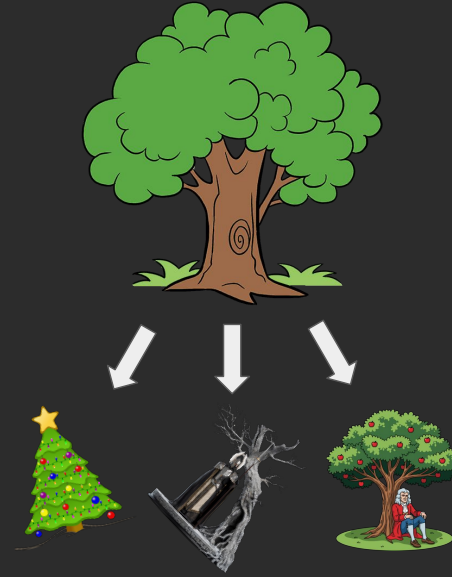
[2] Biran et al. “Explanation and Justification in Machine Learning: A Survey” IJCAI workshop (2017).

[3] Miller. “Explanation in artificial intelligence: Insights from the social sciences” Artificial Intelligence (2019).

[4] Murphy. “Probabilistic machine learning: Advanced topics” MIT press (2023).

How do modern theories analyse complex phenomena?

“That which is apprehended by intelligence and reason is always in the **same state**”



~400 BC

philosophy

How do modern theories analyse complex phenomena?

“That which is apprehended by intelligence and reason is always in the **same state**”

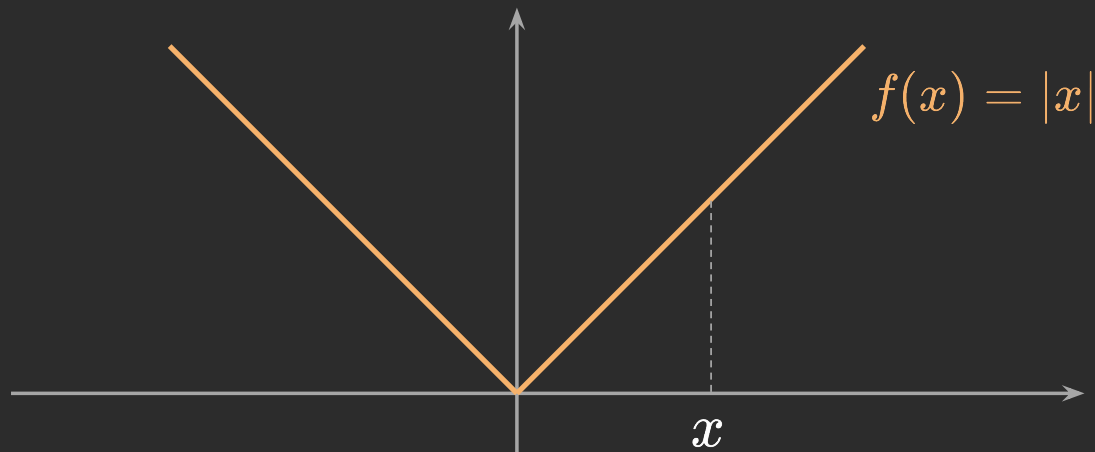


~400 BC
philosophy

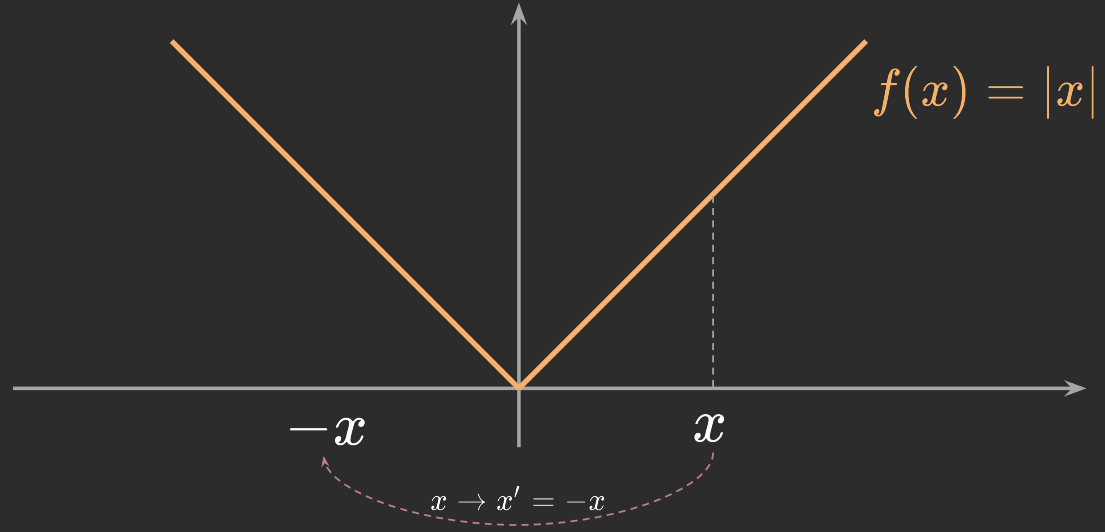
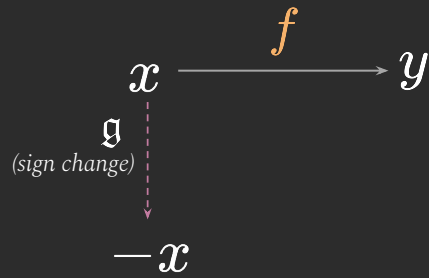
1872
geometry

How do modern theories analyse complex phenomena?

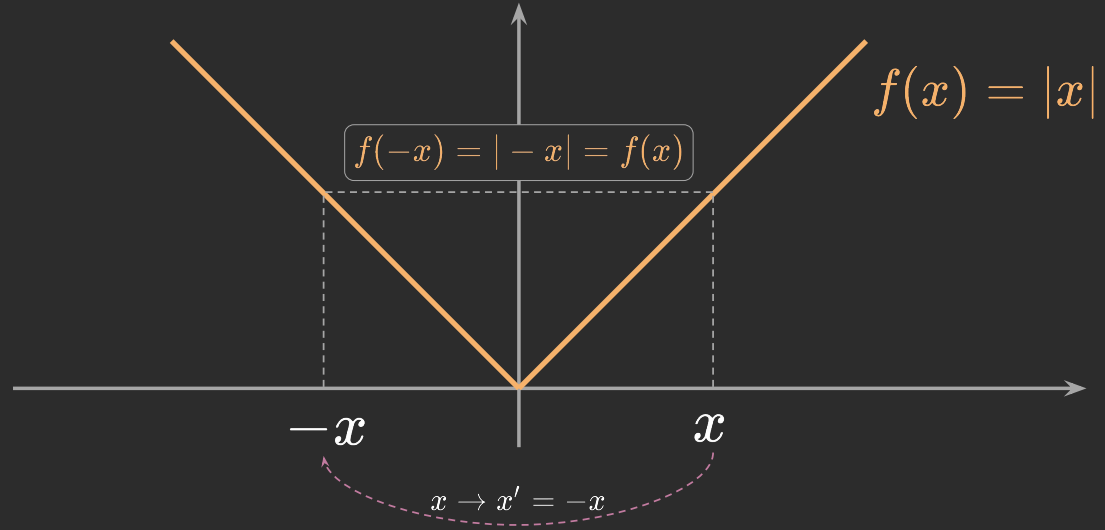
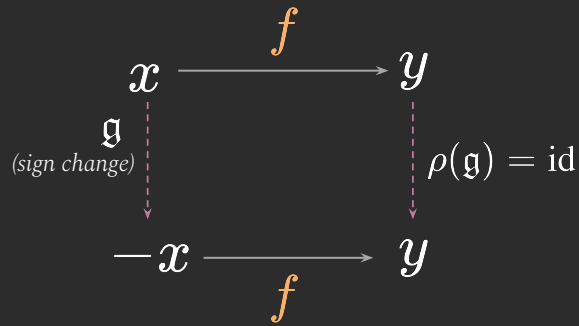
$$x \xrightarrow{f} y$$



How do modern theories analyse complex phenomena?

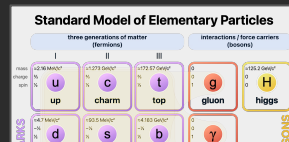


How do modern theories analyse complex phenomena?



How do modern theories analyse complex phenomena?

“That which is apprehended by intelligence and reason is always in the **same state**”



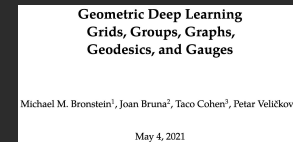
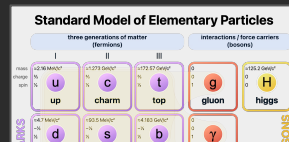
~400 BC
philosophy

1872
geometry

'900
physics

How do modern theories analyse complex phenomena?

“That which is apprehended by intelligence and reason is always in the **same state**”



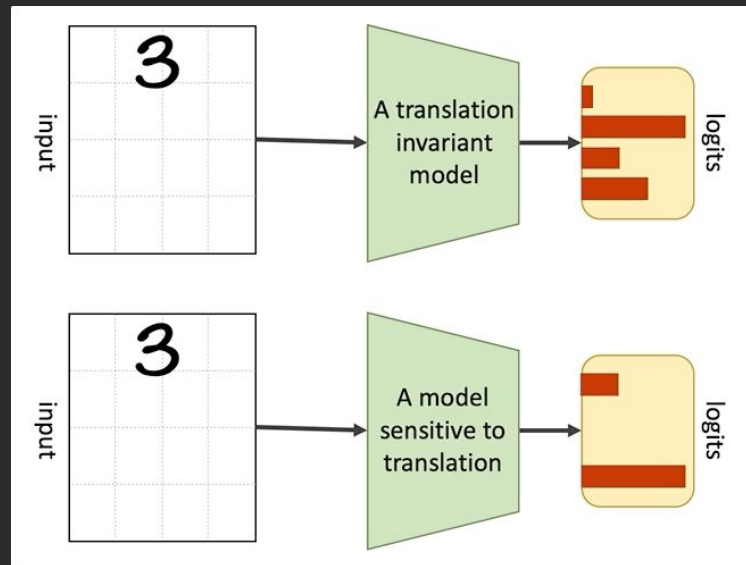
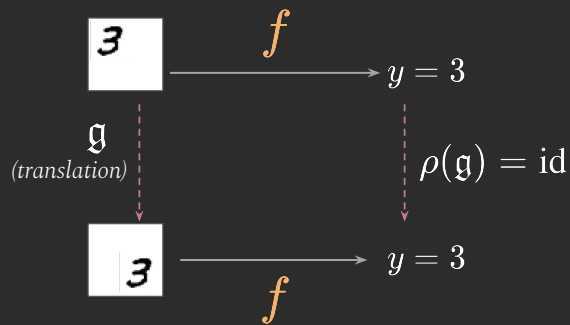
~400 BC
philosophy

1872
geometry

'900
physics

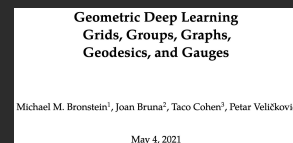
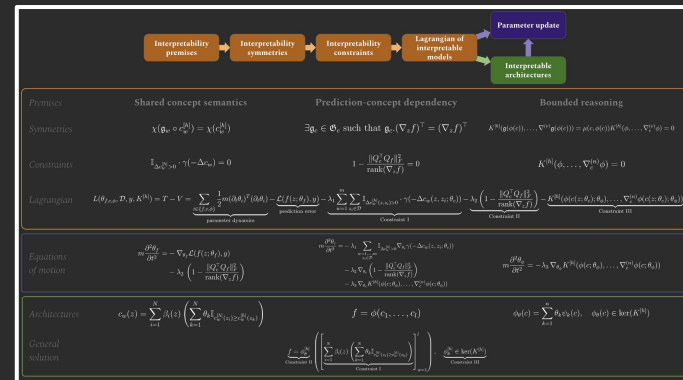
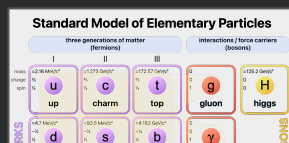
2021
deep learning

How do modern theories analyse complex phenomena?



How do modern theories analyse complex phenomena?

“That which is apprehended by intelligence and reason is always in the **same state**”



~400 BC
philosophy

1872
geometry

'900
physics

2021
deep learning

2026
interpretable ML

The Standard Interpretable Model

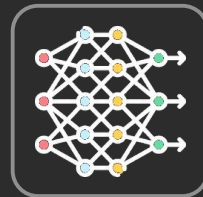
What counts as
interpretable for
the user X?

The Standard
Interpretable Model

Interpretable &
accurate model

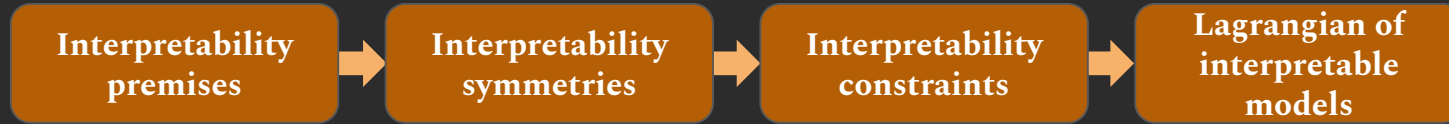


Target
user



Machine
learning model

The Standard Interpretable Model



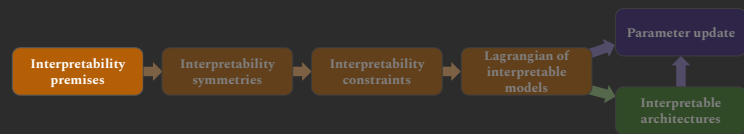
The Standard Interpretable Model



The Standard Interpretable Model



The Standard Interpretable Model



Premises

Symmetries

Constraints

Lagrangian

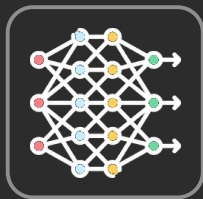
*Equations
of motion*

Architectures

*General
solution*

Interpretability premises

When is f interpretable for the **target user**?



Machine
learning model



Bounded
formal user

Assumption: bounded and formal users



Bounded
formal user



Alfred Tarski



Herbert Simon

[1] Tarski et al. "Undecidable theories." Elsevier (1953).

[2] Simon "Administrative Behavior: A Study of Decision-Making Process in Administrative Organization." (1947).

Assumption: bounded and formal users



Bounded
formal user

apple

red

gray

Fixed vocabulary of symbols
(syntax)

$$c_{\text{red}}^{[h]}(\text{apple}) = 0.9$$

$$c_{\text{red}}^{[h]}(\text{lemon}) = 0.2$$

Mechanism to assign meaning to symbols
(semantics)



Alfred Tarski



Herbert Simon

[1] Tarski et al. "Undecidable theories." Elsevier (1953).

[2] Simon "Administrative Behavior: A Study of Decision-Making Process in Administrative Organization." (1947).

The diagram illustrates the components of a cognitive system, divided into two main sections: **SYNTAX** and **SEMANTICS**.

SYNTAX (Fixed vocabulary of symbols): This section shows three symbols in boxes: "apple", "red", and "gray".

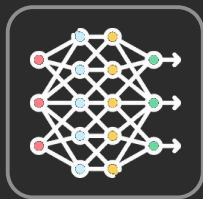
SEMANTICS (Mechanism to assign meaning to symbols): This section shows two examples of how symbols are assigned meaning:

- $c_{\text{red}}^{[h]}(\text{apple}) = 0.9$
- $c_{\text{red}}^{[h]}(\text{lemon}) = 0.2$

The diagram also includes a cartoon illustration of a person at a table with two jars labeled "apple" and "lemon". The person is looking at the jars and saying "MMM, BERRY YUMMY". The cartoon is titled "BOUNDED RATIONALITY" and "UNBOUNDED RATIONALITY". The "BOUNDED" side shows a person with a speech bubble saying "MMM, BERRY YUMMY". The "UNBOUNDED" side shows a person with a speech bubble saying "F_c = \frac{m^2}{\lambda}" and "w_1, w_2 = 2 \times 0" and "E = mc^2" and "S = AB".

Interpretability premises

When is f interpretable for the **target user**?

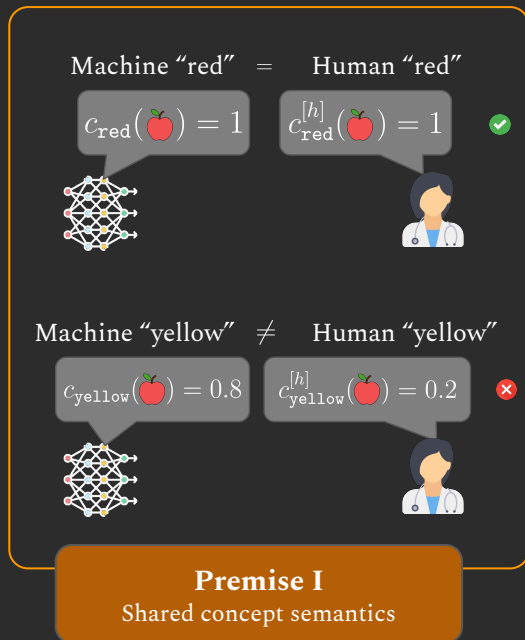


Machine
learning model

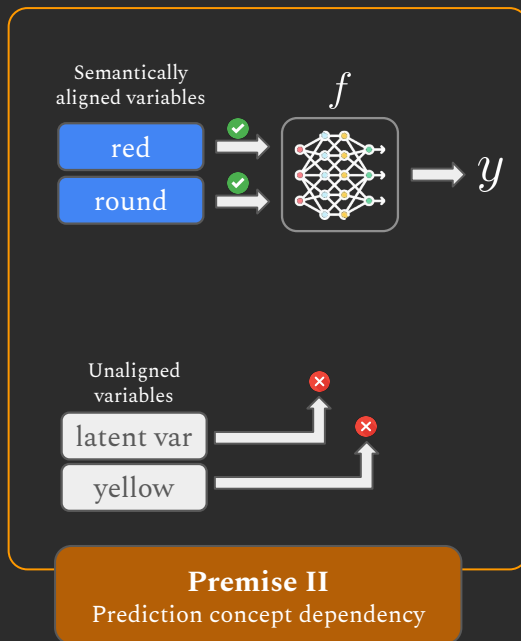
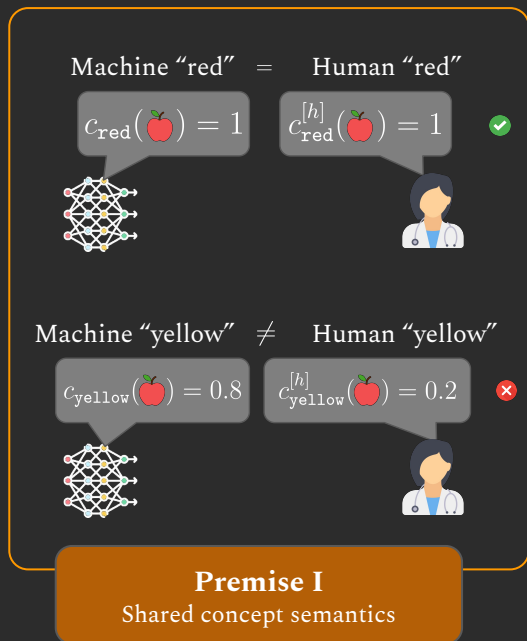


Bounded
formal user

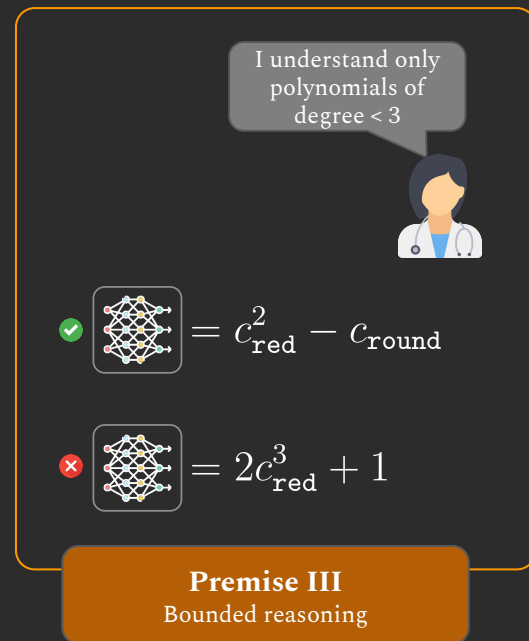
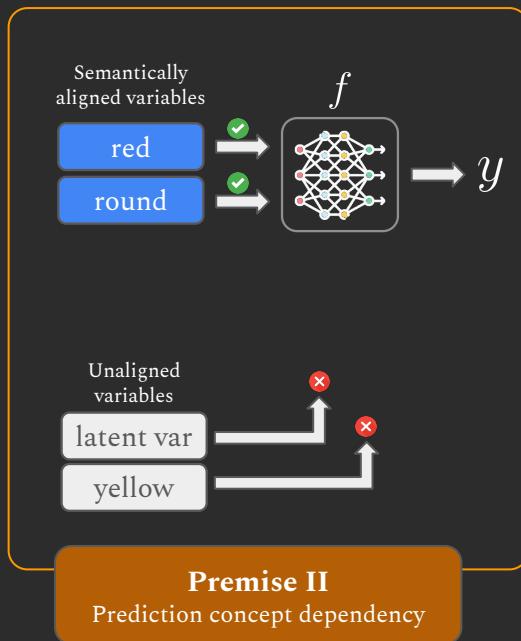
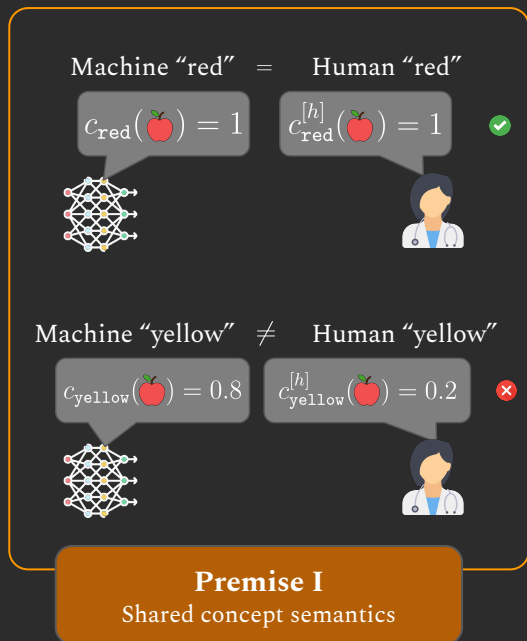
Interpretability premises



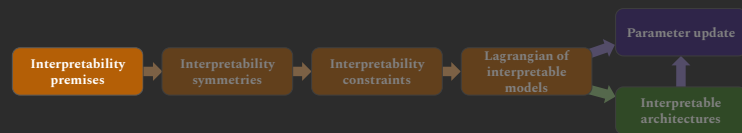
Interpretability premises



Interpretability premises



The Standard Interpretable Model



Premises

Shared concept semantics

Prediction-concept dependency

Bounded reasoning

Symmetries

Constraints

Lagrangian

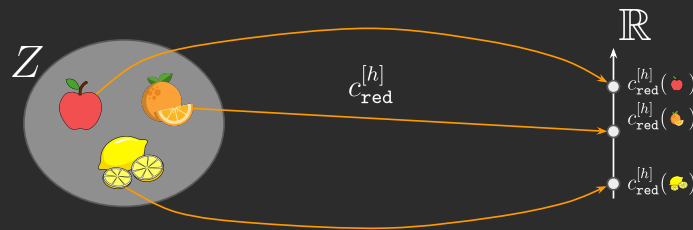
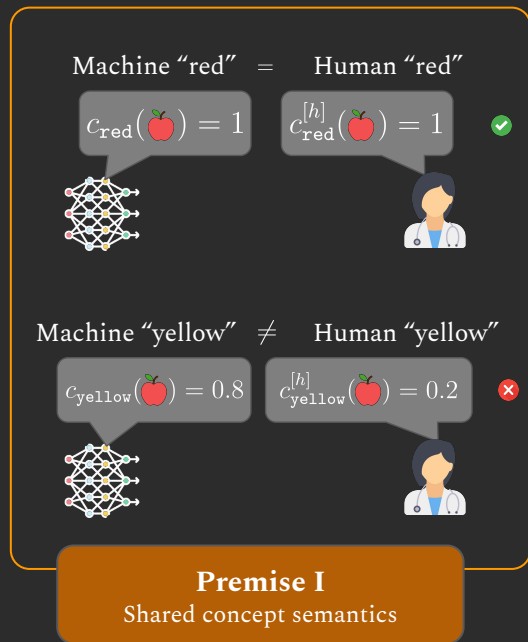
*Equations
of motion*

Architectures

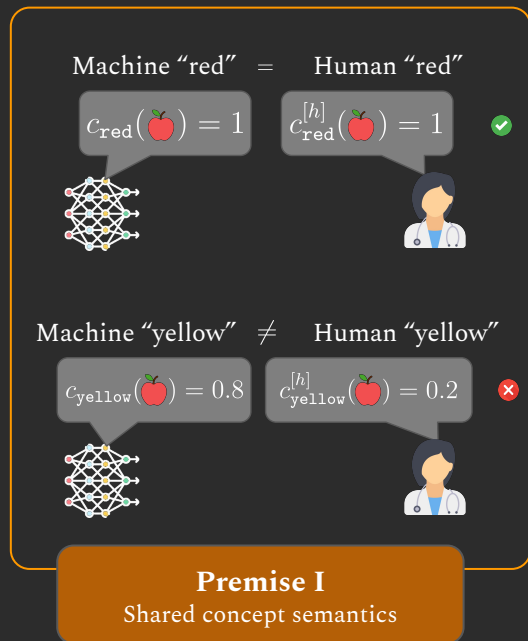
*General
solution*

Premise I $\rightarrow$ Symmetry I

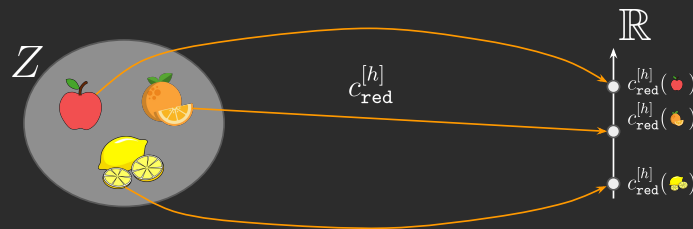
Concept (or interpretation) map:



Premise I $\rightarrow$ Symmetry I



Concept (or interpretation) map:

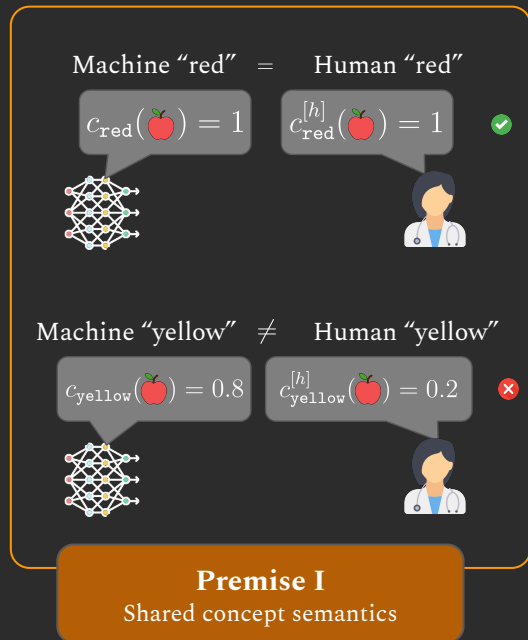


What is a general formal definition of the concept “red”?

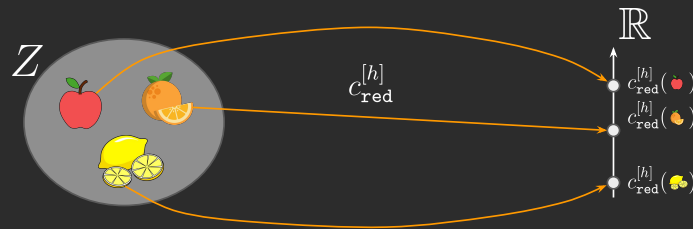
The set of all tuples (z_i, z_j) such that $c_{\text{red}}^{[h]}(z_i) < c_{\text{red}}^{[h]}(z_j)$

$$\chi(c_{\text{red}}^{[h]}) = \{(\text{lemon}, \text{orange}), (\text{orange}, \text{apple}), (\text{lemon}, \text{apple})\}$$

Premise I → Symmetry I



Concept (or interpretation) map:



What is a general formal definition of the concept “red”?

The set of all tuples (z_i, z_j) such that $c_{\text{red}}^{[h]}(z_i) < c_{\text{red}}^{[h]}(z_j)$

$$\chi(c_{\text{red}}^{[h]}) = \{(\text{lemon}, \text{orange}), (\text{orange}, \text{apple}), (\text{lemon}, \text{apple})\}$$

Which transformations preserve concepts?

Monotonic maps:

$$\mathfrak{G}_w = \{\mathfrak{g}_w : \mathbb{R} \rightarrow \mathbb{R} \mid \forall s_1, s_2 \in \mathbb{R}, \quad s_1 < s_2 \implies \mathfrak{g}_w(s_1) < \mathfrak{g}_w(s_2)\}$$

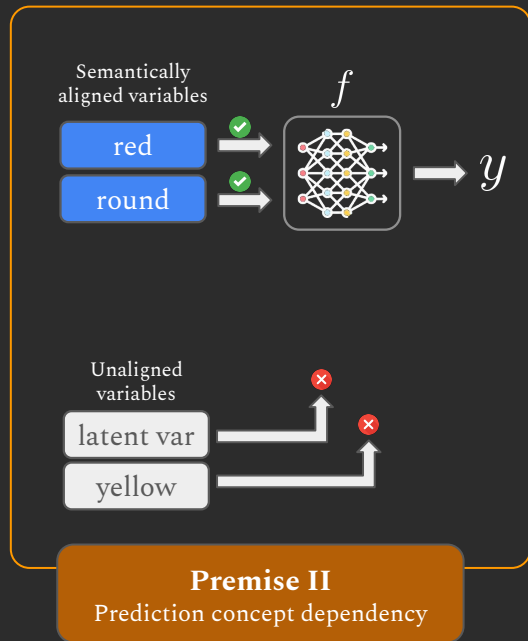
$$\chi(\mathfrak{g}_w \circ c_{\text{red}}^{[h]}) = \chi(c_{\text{red}}^{[h]}) = \{(\text{lemon}, \text{orange}), (\text{orange}, \text{apple}), (\text{lemon}, \text{apple})\}$$

Premise II $\rightarrow$ Symmetry II

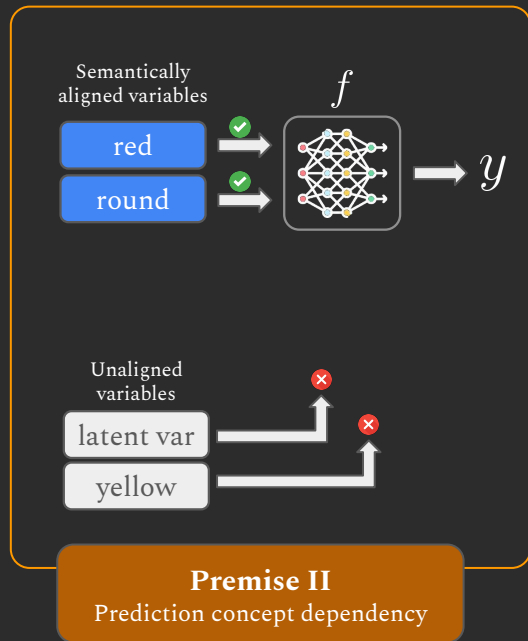
When does a function depend exclusively on aligned concepts?

When any change in the function output $\nabla_z f_1, \dots, \nabla_z f_v$ can be explained by a change in aligned concepts $\nabla_z c_1, \dots, \nabla_z c_l$

$$\text{span}\{\nabla_z f_1, \dots, \nabla_z f_v\} \subseteq \text{span}\{\nabla_z c_1, \dots, \nabla_z c_l\}$$



Premise II $\rightarrow$ Symmetry II



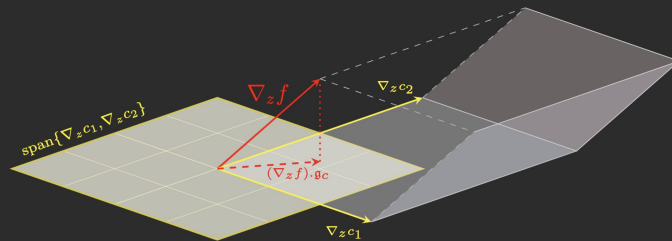
When does a function depend exclusively on aligned concepts?

When any change in the function output $\nabla_z f_1, \dots, \nabla_z f_v$ can be explained by a change in aligned concepts $\nabla_z c_1, \dots, \nabla_z c_l$

$$\text{span}\{\nabla_z f_1, \dots, \nabla_z f_v\} \subseteq \text{span}\{\nabla_z c_1, \dots, \nabla_z c_l\}$$

Which transformation guarantees prediction-concept dependency?

A projection into the concept subspace that leaves the output gradients invariant



$$\exists \mathbf{g}_c \in \mathfrak{G}_c \text{ such that } \mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

$$\mathfrak{G}_c = \{\mathbf{g}_c : Z \rightarrow Z \mid \mathbf{g}_c^2 = \mathbf{g}_c, \text{ im}(\mathbf{g}_c) \subseteq \text{span}(\nabla_z c)\}$$

Premise III → Symmetry III

How can a user specify the hypothesis space?

$$\{\phi \mid K^{[h]}(\phi) = \nabla_c^3 \phi = 0\}$$

$$\mathcal{H} \quad \begin{matrix} 2c_1^2 + 3 & c_1^2 + 3c_2 \\ c_1^2 + c_3^2 \end{matrix}$$

I understand only
polynomials of
degree < 3



✓  $= c_{\text{red}}^2 - c_{\text{round}}$

✗  $= 2c_{\text{red}}^3 + 1$

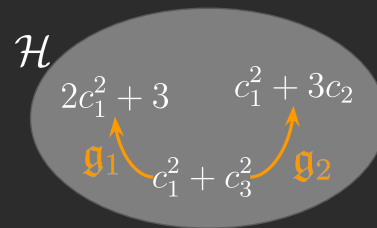
Premise III

Bounded reasoning

Premise III → Symmetry III

How can a user specify the hypothesis space?

$$\{\phi \mid K^{[h]}(\phi) = \nabla_c^3 \phi = 0\}$$



Which transformations of concept formulae preserve the hypothesis space?

All transformations that map solutions of $K^{[h]}$ to other solutions

$$K^{[h]}(\mathbf{g}(\phi(c)), \dots, \nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\phi, \dots, \nabla_c^{(\nu)} \phi) = 0$$

I understand only
polynomials of
degree < 3



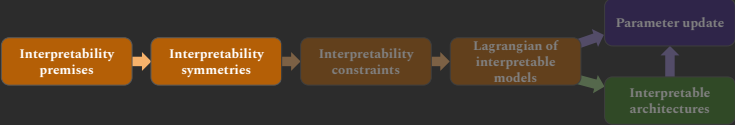
✓  $= c_{\text{red}}^2 - c_{\text{round}}$

✗  $= 2c_{\text{red}}^3 + 1$

Premise III

Bounded reasoning

The Standard Interpretable Model



<i>Premises</i>	Shared concept semantics	Prediction-concept dependency	Bounded reasoning
<i>Symmetries</i>	$\chi(\mathbf{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$	$\exists \mathbf{g}_c \in \mathfrak{G}_c$ such that $\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$	$K^{[h]}(\mathbf{g}(\phi(c)), \dots, \nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\phi, \dots, \nabla_c^{(\nu)} \phi) = 0$
<i>Constraints</i>			
<i>Lagrangian</i>			

<i>Equations of motion</i>

<i>Architectures</i>
<i>General solution</i>

Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I

Machine “red” = Human “red”

$$c_{\text{red}}(\text{apple}) = 1$$



$$c_{\text{red}}^{[h]}(\text{apple}) = 1$$



Machine “yellow” $\neq$ Human “yellow”

$$c_{\text{yellow}}(\text{apple}) = 0.8$$



$$c_{\text{yellow}}^{[h]}(\text{apple}) = 0.2$$



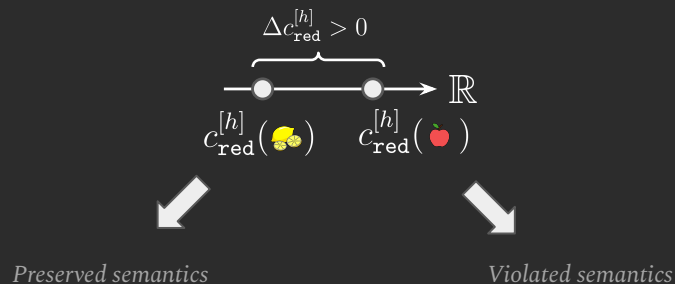
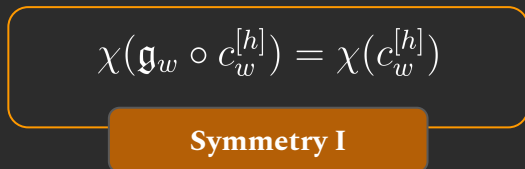
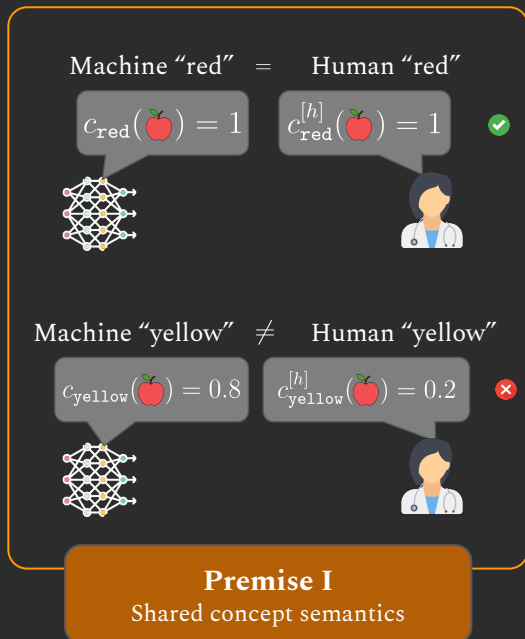
Premise I

Shared concept semantics

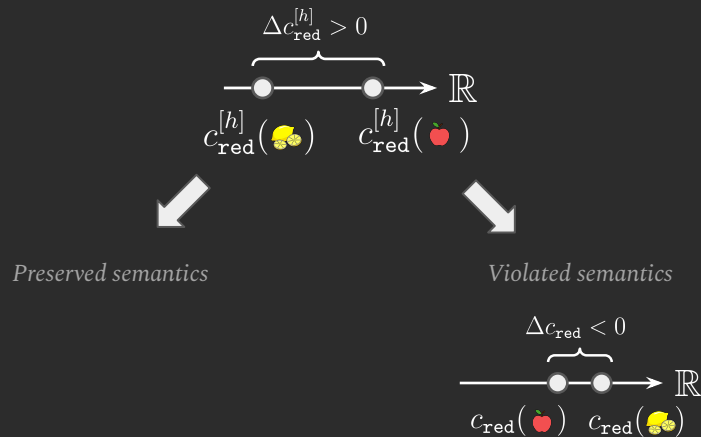
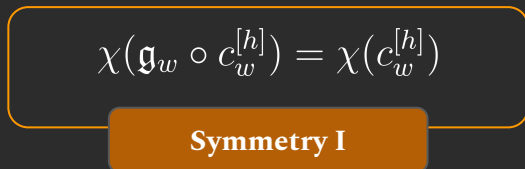
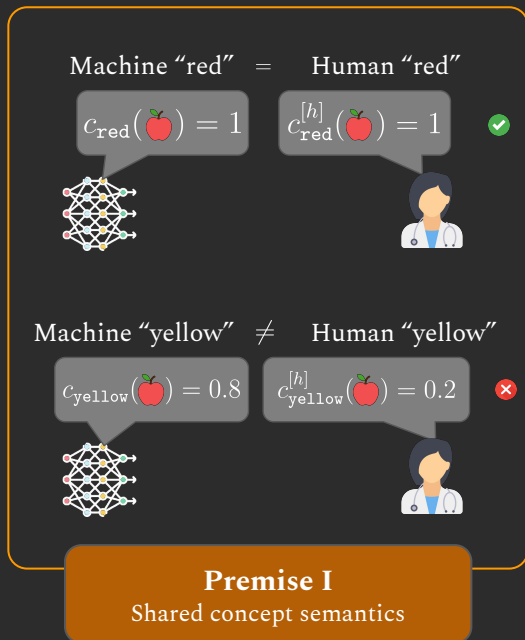
$$\chi(\mathfrak{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

Symmetry I

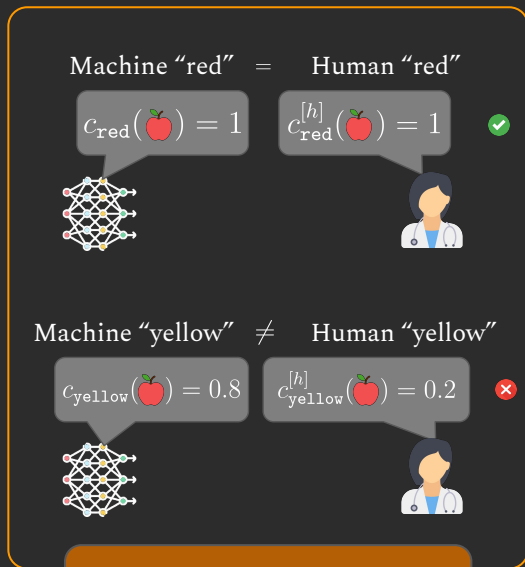
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I



Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I



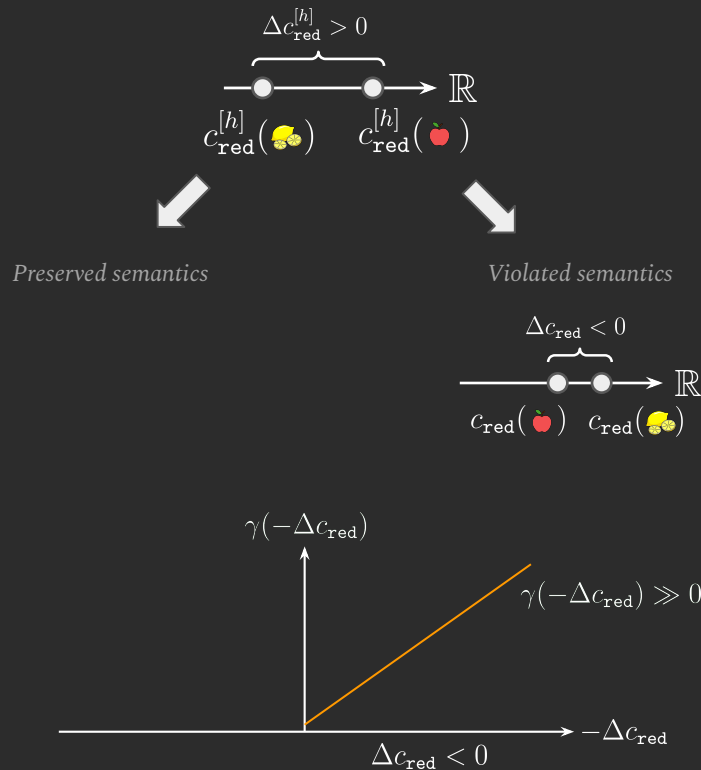
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I



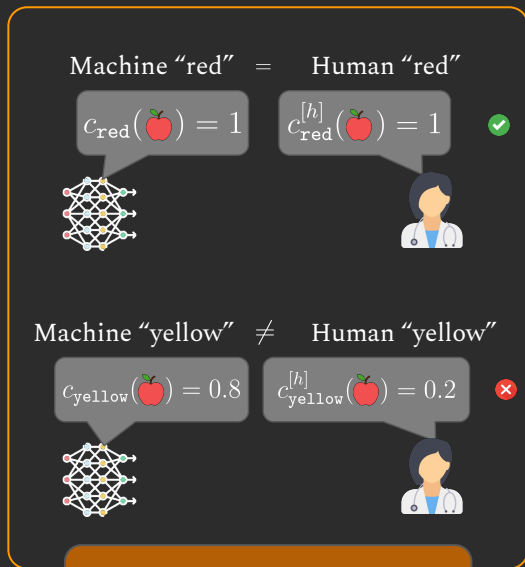
Premise I
Shared concept semantics

$$\chi(\mathfrak{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

Symmetry I



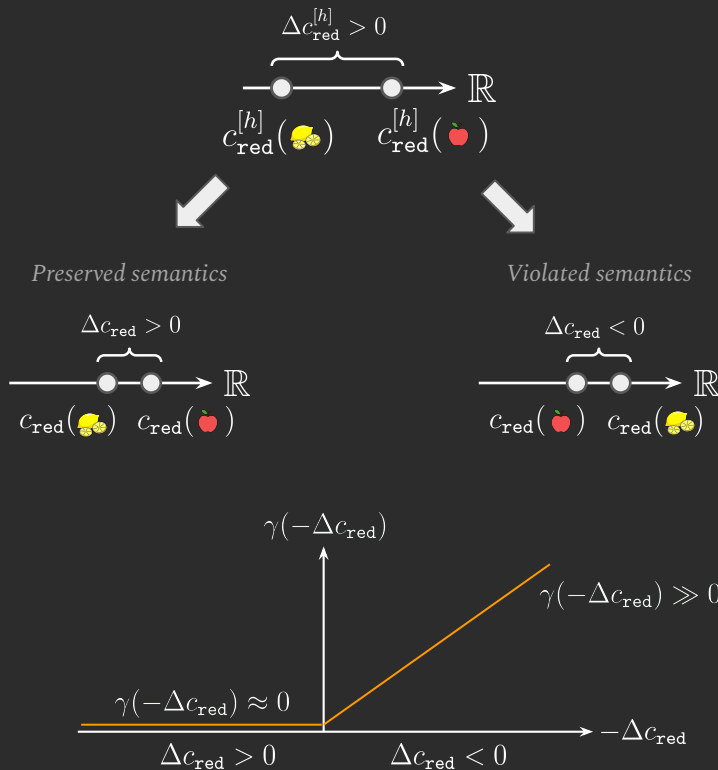
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I



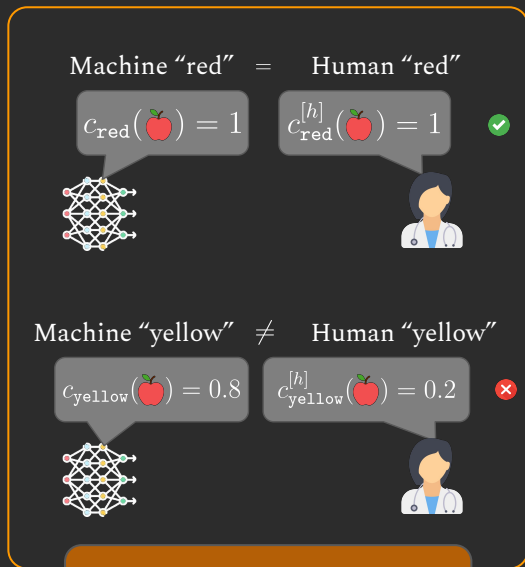
Premise I
Shared concept semantics

$$\chi(\mathfrak{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

Symmetry I



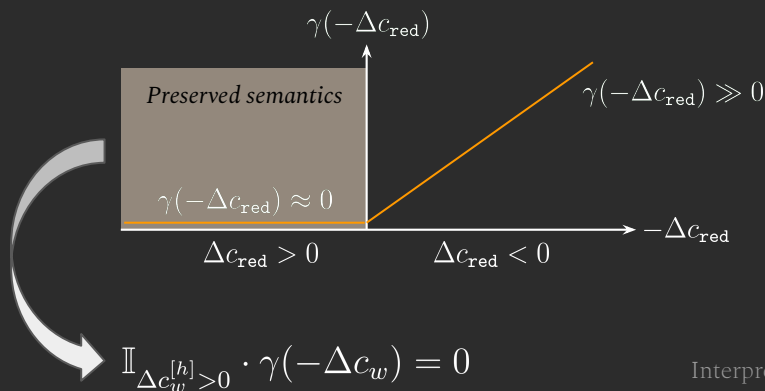
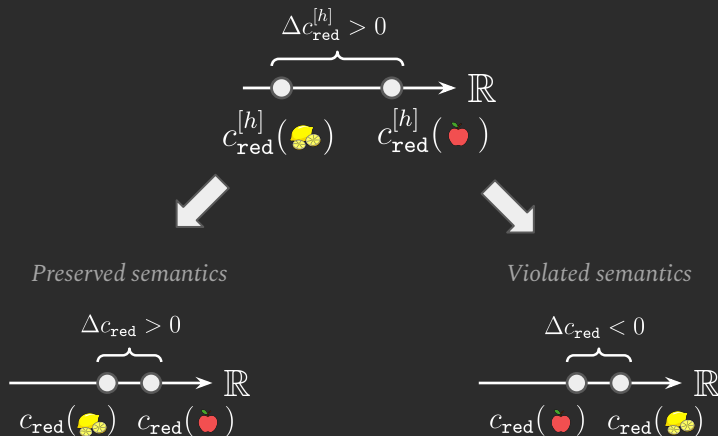
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I



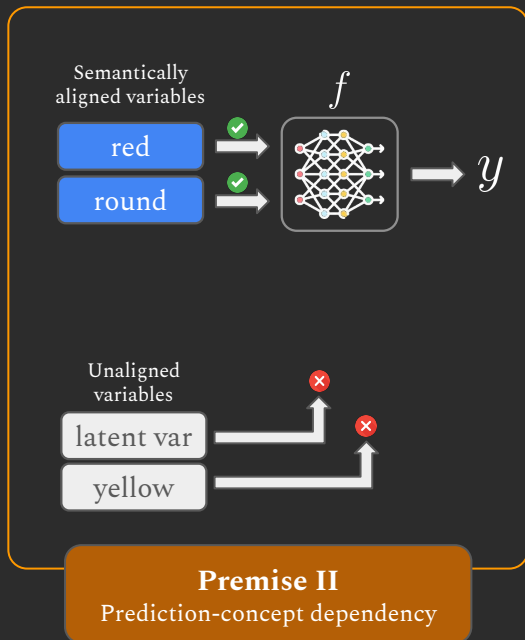
Premise I
Shared concept semantics

$$\chi(\mathfrak{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

Symmetry I



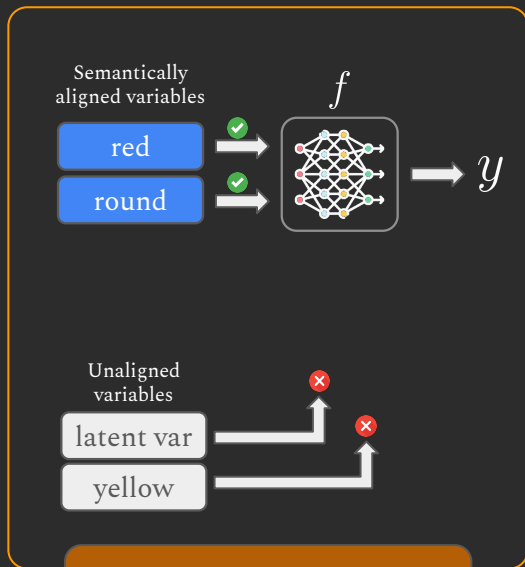
Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II



$$\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

Symmetry II

Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II

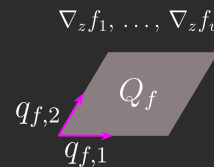
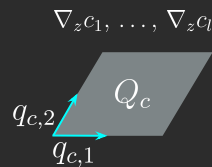


Premise II

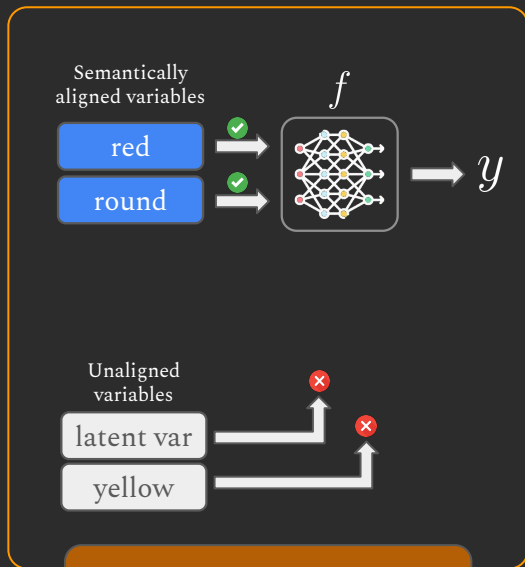
Prediction-concept dependency

$$\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

Symmetry II



Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II

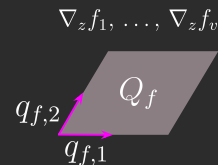
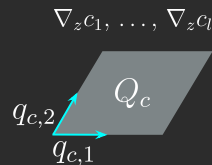


Premise II

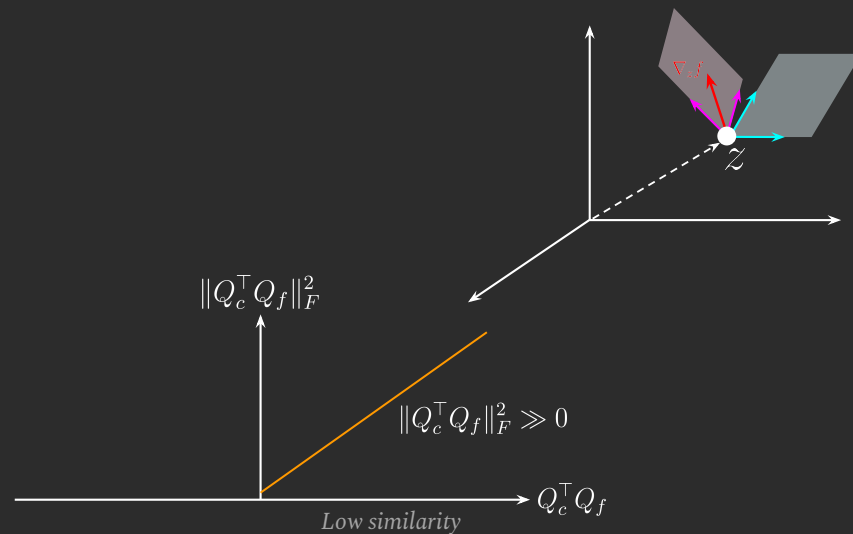
Prediction-concept dependency

$$\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

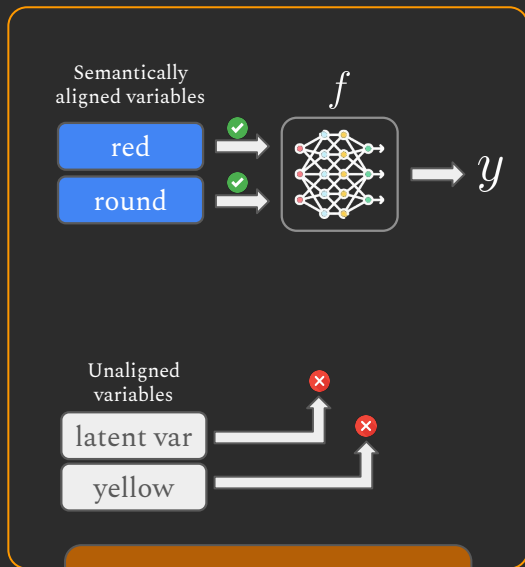
Symmetry II



Predictions do not depend on concepts



Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II

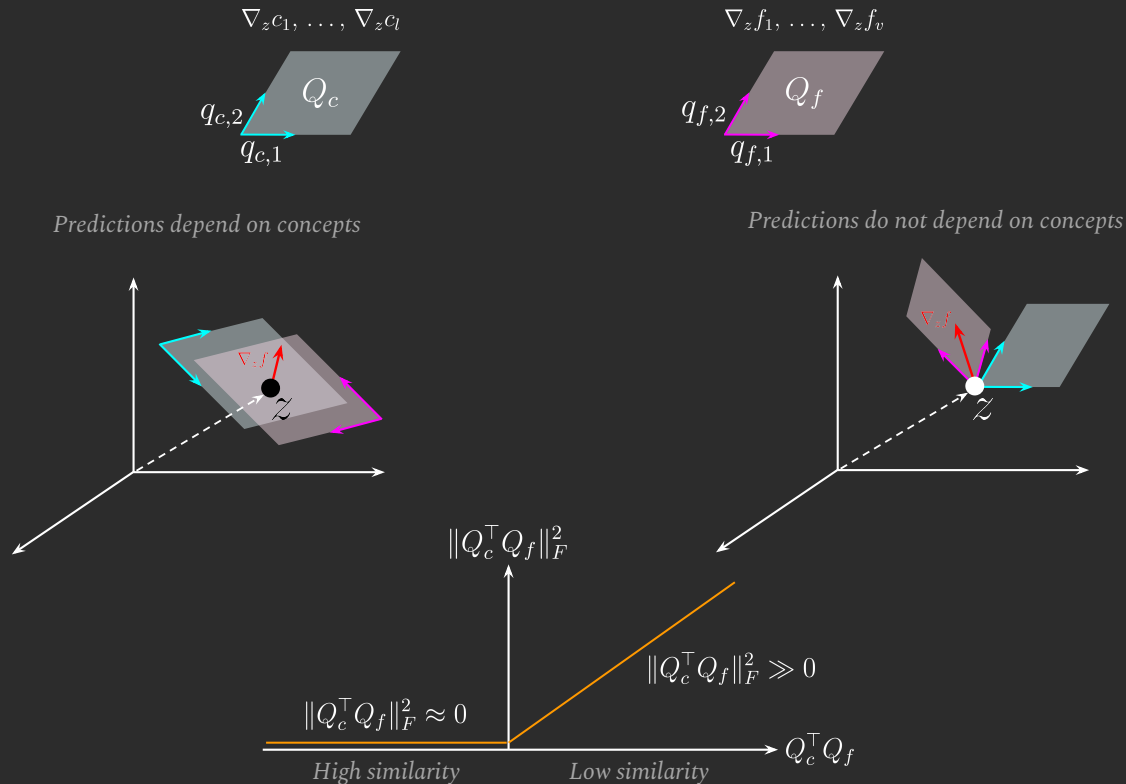


Premise II

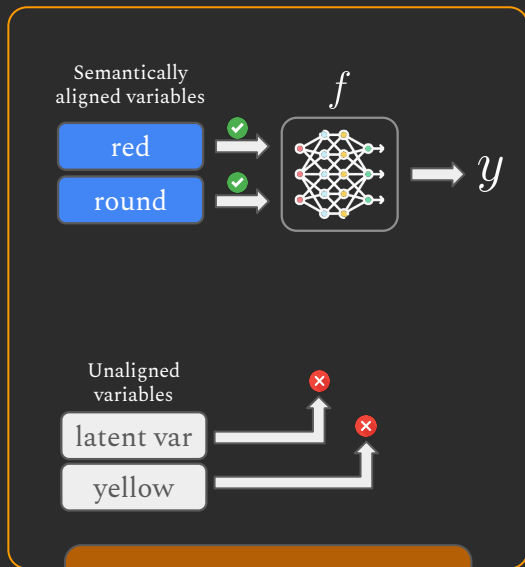
Prediction-concept dependency

$$\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

Symmetry II



Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II

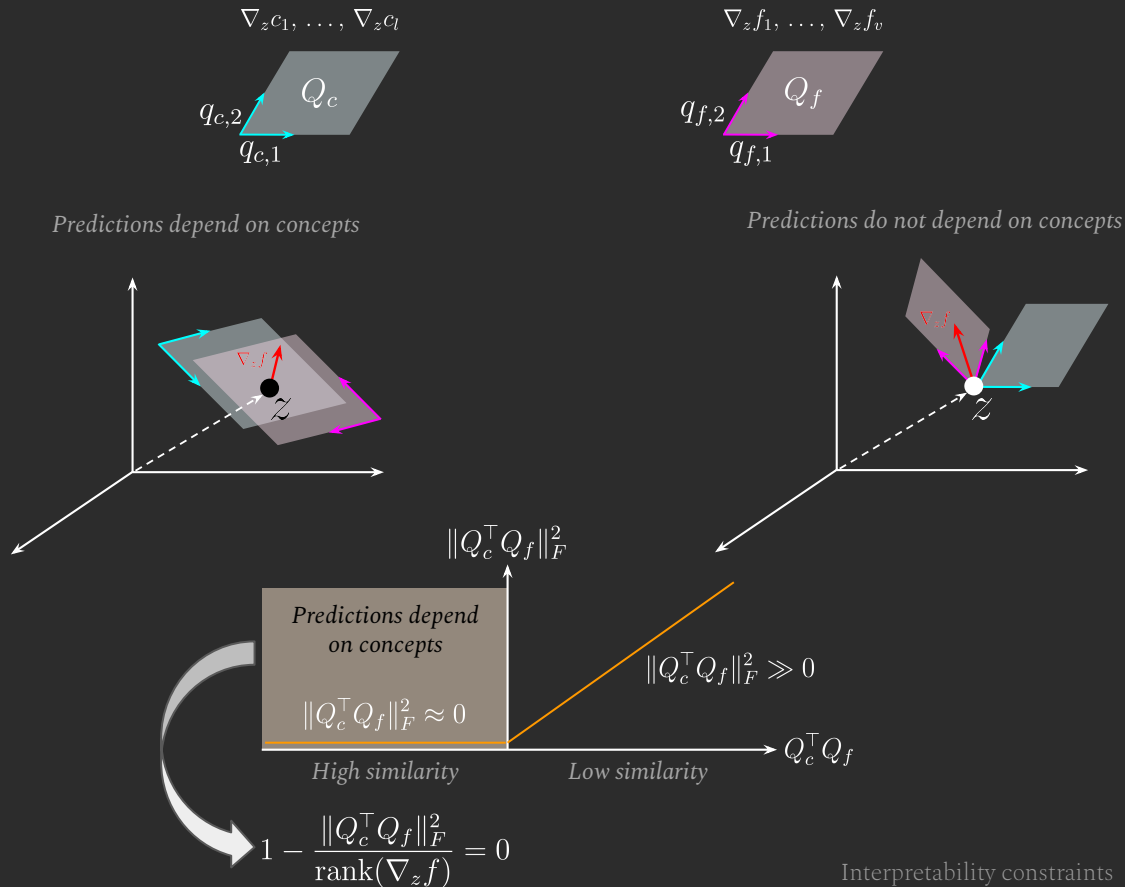


Premise II

Prediction-concept dependency

$$\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

Symmetry II



Premise III $\rightarrow$ Symmetry III $\rightarrow$ Constraint III

I understand only
polynomials of
degree < 3



✓  $= c_{\text{red}}^2 - c_{\text{round}}$

✗  $= 2c_{\text{red}}^3 + 1$

Premise III
Bounded reasoning

$$K^{[h]}(\nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\nabla_c^{(\nu)} \phi) = 0$$



$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

$$K^{[h]}(\nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\nabla_c^{(\nu)} \phi) = 0$$

Symmetry III

The Standard Interpretable Model



<i>Premises</i>	Shared concept semantics	Prediction-concept dependency	Bounded reasoning
<i>Symmetries</i>	$\chi(\mathbf{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$	$\exists \mathbf{g}_c \in \mathfrak{G}_c$ such that $\mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$	$K^{[h]}(\mathbf{g}(\phi(c)), \dots, \nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\phi, \dots, \nabla_c^{(\nu)} \phi) = 0$
<i>Constraints</i>	$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$	$1 - \frac{\ Q_c^\top Q_f\ _F^2}{\text{rank}(\nabla_z f)} = 0$	$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$
<i>Lagrangian</i>			

<i>Equations of motion</i>

<i>Architectures</i>
<i>General solution</i>

Lagrangian of interpretable machine learning models

$$\mathbb{I}_{\Delta c_w > 0} \cdot \gamma(-\Delta c_w) = 0$$

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Constraints

Lagrangian of interpretable machine learning models

$$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$$

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Constraints

$$\min_{\theta_f, \theta_c, \theta_\phi} \mathcal{L}(f(z; \theta_f), y)$$

$$\text{s.t.} \quad \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c)) = 0 \quad (\text{Constraint 1})$$

$$\left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right) = 0 \quad (\text{Constraint 2})$$

$$K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi)) = 0 \quad (\text{Constraint 3})$$

Constrained optimisation problem

Lagrangian of interpretable machine learning models

$$\begin{aligned}\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) &= 0 \\ 1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} &= 0 \\ K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) &= 0\end{aligned}$$

Constraints

$$\begin{aligned}\min_{\theta_f, \theta_c, \theta_\phi} \quad & \mathcal{L}(f(z; \theta_f), y) \\ \text{s.t.} \quad & \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c)) = 0 \quad (\text{Constraint 1}) \\ & \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right) = 0 \quad (\text{Constraint 2}) \\ & K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi)) = 0 \quad (\text{Constraint 3})\end{aligned}$$

Constrained optimisation problem

$$\begin{aligned}\max_{\lambda_i > 0} \min_{\theta_f, \theta_\phi, \theta_c} \quad & L(\theta_f, \theta_c, \phi, \lambda_i, \mathcal{D}, y, K^{[h]}) = \mathcal{L}(f(z; \theta_f), y) \\ & + \lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c)) \\ & + \lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right) \\ & + \lambda_3 K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi))\end{aligned}$$

Unconstrained optimisation problem

Lagrangian of interpretable machine learning models

$$\begin{aligned}\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) &= 0 \\ 1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} &= 0 \\ K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) &= 0\end{aligned}$$

Constraints

$$\begin{aligned}\min_{\theta_f, \theta_c, \theta_\phi} \quad & \mathcal{L}(f(z; \theta_f), y) \\ \text{s.t.} \quad & \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c)) = 0 \quad (\text{Constraint 1}) \\ & \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right) = 0 \quad (\text{Constraint 2}) \\ & K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi)) = 0 \quad (\text{Constraint 3})\end{aligned}$$

Constrained optimisation problem

$$\begin{aligned}\max_{\lambda_i > 0} \min_{\theta_f, \theta_c, \theta_\phi} \quad & L(\theta_f, \theta_c, \phi, \lambda_i, \mathcal{D}, y, K^{[h]}) = \mathcal{L}(f(z; \theta_f), y) \\ & + \lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c)) \\ & + \lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right) \\ & + \lambda_3 K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi))\end{aligned}$$

Unconstrained optimisation problem

$$L(\theta_{f,c,\phi}, \mathcal{D}, y, K^{[h]}) = T - V = \underbrace{\sum_{i \in \{f,c,\phi\}} \frac{1}{2} m (\partial_i \theta_i)^T (\partial_i \theta_i)}_{\text{parameter dynamics}} - \underbrace{\mathcal{L}(f(z; \theta_f), y)}_{\text{prediction error}} - \underbrace{\lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c))}_{\text{Constraint I}} - \underbrace{\lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right)}_{\text{Constraint II}} - \underbrace{\lambda_3 K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi))}_{\text{Constraint III}}$$

Lagrangian

The Standard Interpretable Model



Premises

Shared concept semantics

Prediction-concept dependency

Bounded reasoning

Symmetries

$$\chi(\mathbf{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

$$\exists \mathbf{g}_c \in \mathfrak{G}_c \text{ such that } \mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

$$K^{[h]}(\mathbf{g}(\phi(c)), \dots, \nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\phi, \dots, \nabla_c^{(\nu)} \phi) = 0$$

Constraints

$$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$$

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Lagrangian

$$L(\theta_{f,c,\phi}, \mathcal{D}, y, K^{[h]}) = T - V = \underbrace{\sum_{i \in \{f,c,\phi\}} \frac{1}{2} m(\partial_i \theta_i)^T (\partial_i \theta_i)}_{\text{parameter dynamics}} - \underbrace{\mathcal{L}(f(z; \theta_f), y)}_{\text{prediction error}} - \underbrace{\lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c))}_{\text{Constraint I}} - \underbrace{\lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right)}_{\text{Constraint II}} - \underbrace{K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi))}_{\text{Constraint III}}$$

Equations

of motion

Architectures

General

solution

Trajectories towards interpretability

$$L(\mathcal{D}, \theta) = \underbrace{T(\partial_t \theta)}_{\text{parameter dynamics}} - \underbrace{V(\mathcal{D}, \theta)}_{\text{objective function}}$$

Lagrangian

$$S = \iint L(\mathcal{D}, \theta) dx dt$$

Action

Principle of least action

$$\delta S = 0 \iff \frac{d}{dt} \left(\frac{\partial L}{\partial (\partial_t \theta)} \right) - \frac{\partial L}{\partial \theta} = 0$$

Euler-Lagrange equations

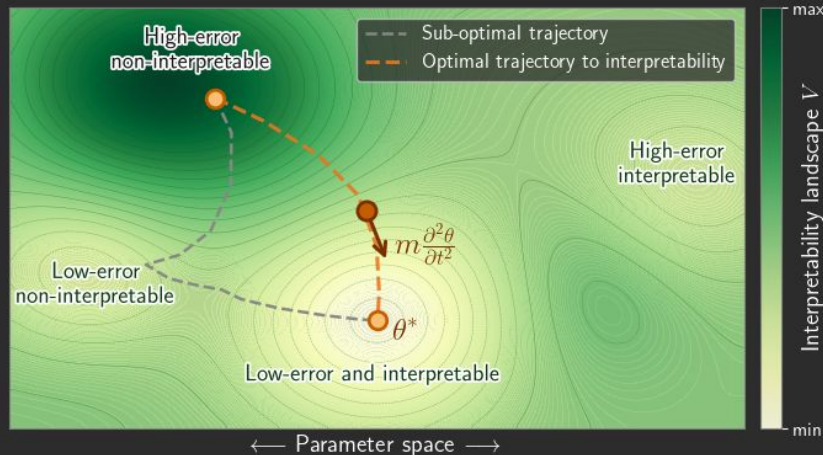
$$m \frac{\partial^2 \theta}{\partial t^2} = \nabla_{\theta} V(\mathcal{D}, \theta)$$

Equations of motion

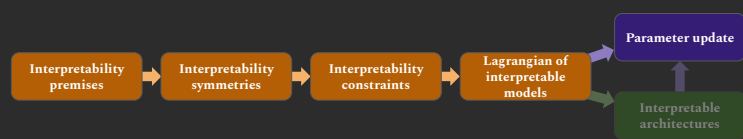
Discretization

$$\theta_{t+1} = \theta_t + (\theta_t - \theta_{t-1}) - \frac{(\Delta t)^2}{m} \nabla_{\theta} V(\mathcal{D}, \theta)$$

Parameter update



The Standard Interpretable Model



Premises

Shared concept semantics

Prediction-concept dependency

Bounded reasoning

Symmetries

$$\chi(\mathbf{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

$$\exists \mathbf{g}_c \in \mathfrak{G}_c \text{ such that } \mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

$$K^{[h]}(\mathbf{g}(\phi(c)), \dots, \nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\phi, \dots, \nabla_c^{(\nu)} \phi) = 0$$

Constraints

$$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$$

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Lagrangian

$$L(\theta_{f,c,\phi}, \mathcal{D}, y, K^{[h]}) = T - V = \underbrace{\sum_{i \in \{f,c,\phi\}} \frac{1}{2} m (\partial_i \theta_i)^T (\partial_i \theta_i)}_{\text{parameter dynamics}} - \underbrace{\underbrace{\mathcal{L}(f(z; \theta_f), y)}_{\text{prediction error}} - \lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c))}_{\text{Constraint I}} - \underbrace{\lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right)}_{\text{Constraint II}} - \underbrace{K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi))}_{\text{Constraint III}}$$

Equations

of motion

$$m \frac{\partial^2 \theta_f}{\partial t^2} = -\nabla_{\theta_f} \mathcal{L}(f(z; \theta_f), y) - \lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right)$$

$$m \frac{\partial^2 \theta_c}{\partial t^2} = -\lambda_1 \sum_{\substack{w=1, \dots, m \\ z_i \in \mathcal{D}}} \mathbb{I}_{\Delta c_w^{[h]} > 0} \nabla_{\theta_c} \gamma(-\Delta c_w(z, z_i; \theta_c)) - \lambda_2 \nabla_{\theta_c} \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right) - \lambda_3 \nabla_{\theta_c} K^{[h]}(\phi(c; \theta_\phi), \dots, \nabla_c^{(n)} \phi(c; \theta_\phi))$$

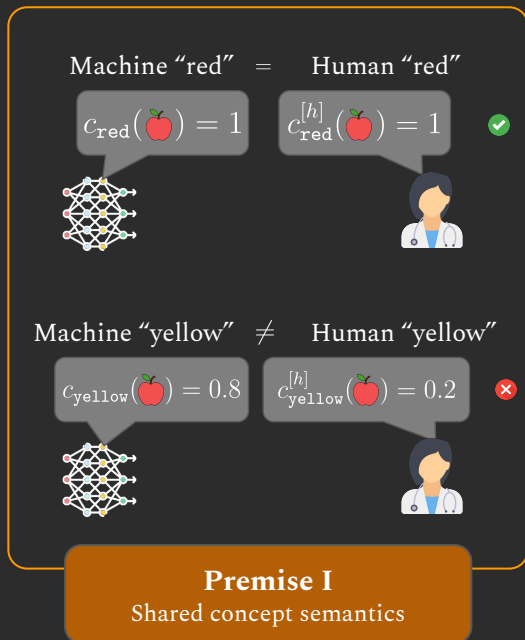
$$m \frac{\partial^2 \theta_\phi}{\partial t^2} = -\lambda_3 \nabla_{\theta_\phi} K^{[h]}(\phi(c; \theta_\phi), \dots, \nabla_c^{(n)} \phi(c; \theta_\phi))$$

Architectures

General

solution

Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



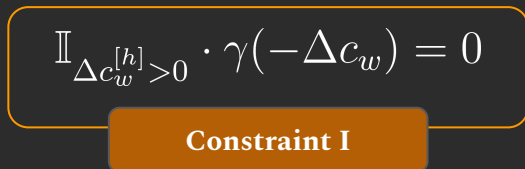
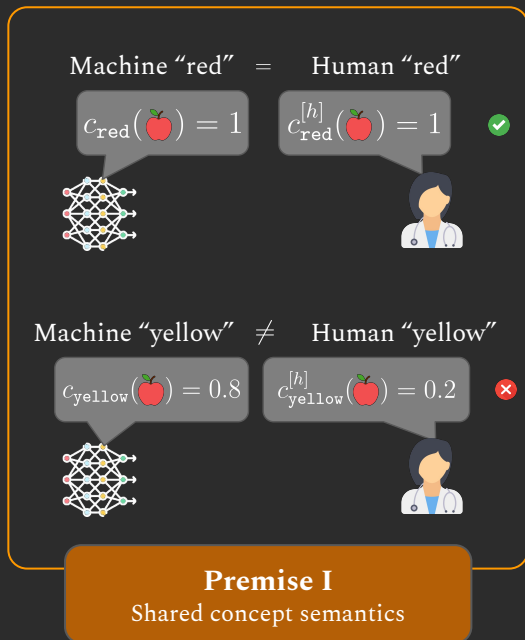
$$\text{---} \bullet \text{---} \bullet \text{---} \bullet \rightarrow \mathbb{R}$$

$$c_{\text{red}}^{[h]}(\text{lemon}) \quad c_{\text{red}}^{[h]}(\text{orange}) \quad c_{\text{red}}^{[h]}(\text{apple})$$

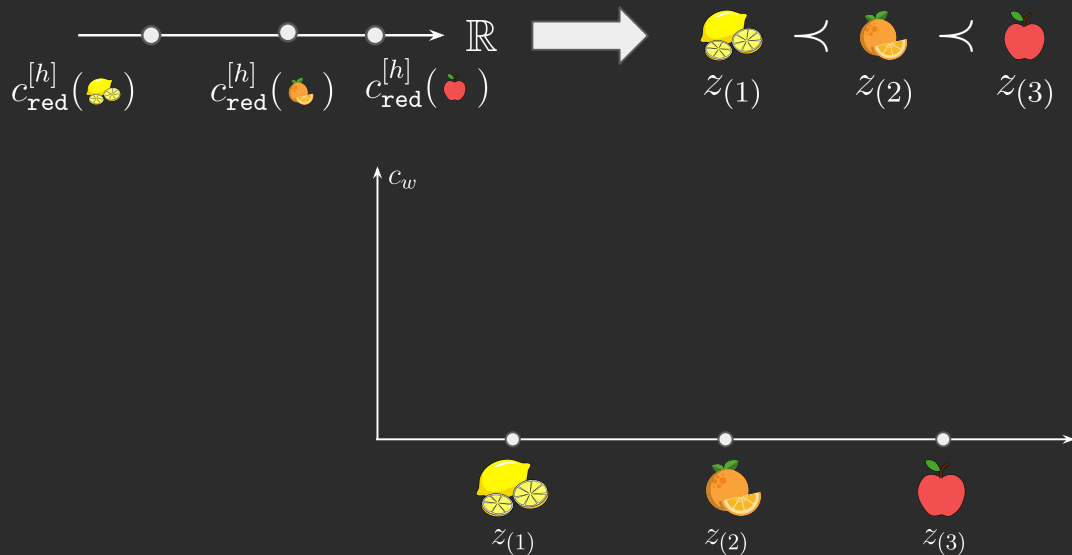
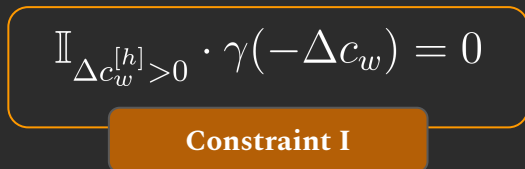
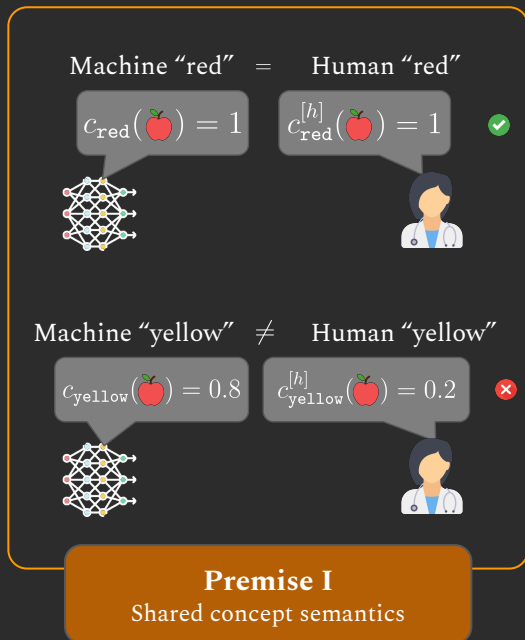
$$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$$

Constraint I

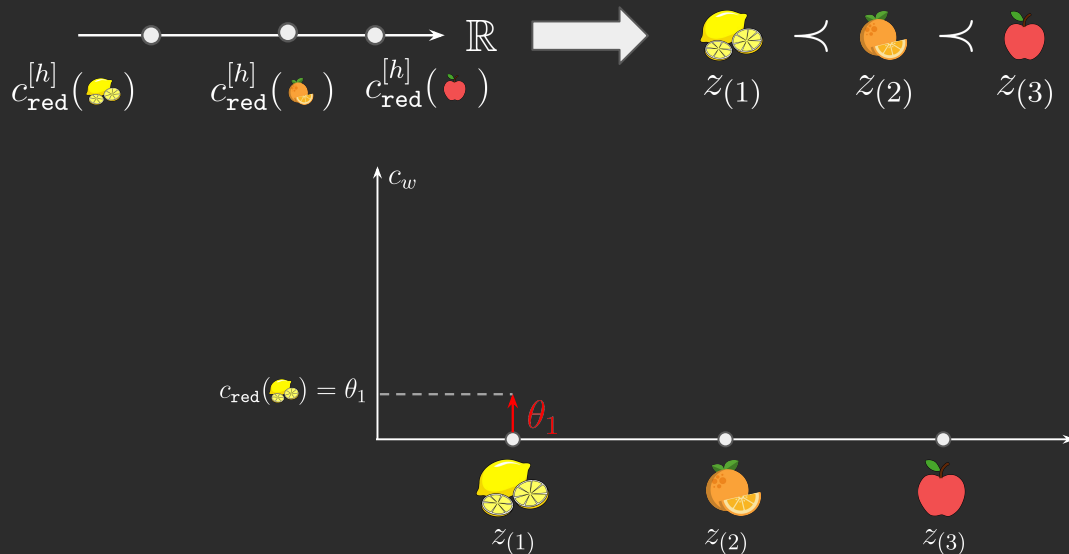
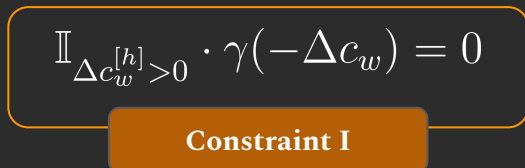
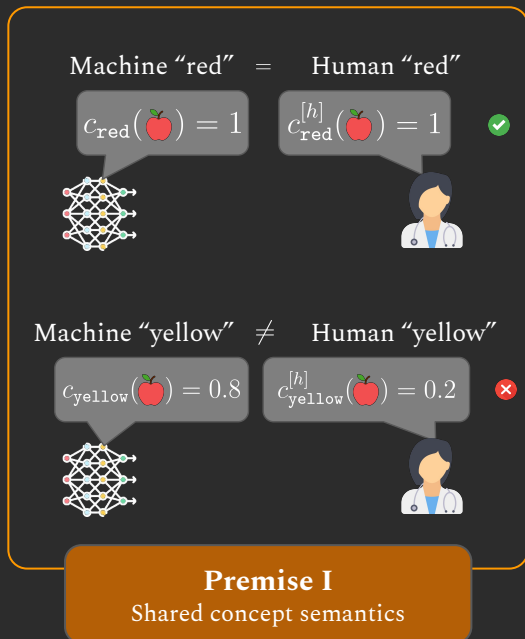
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



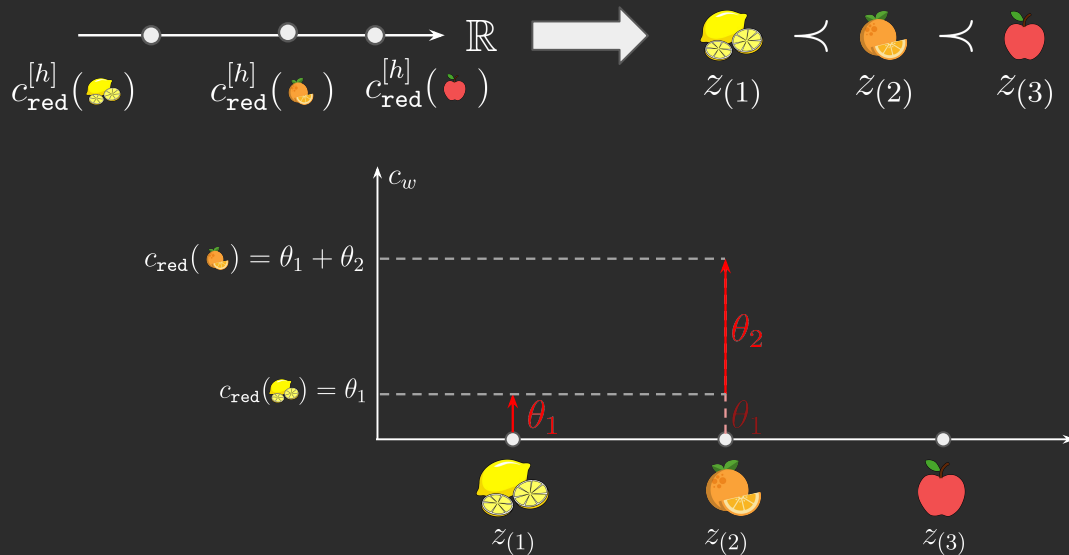
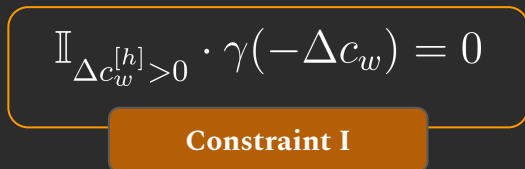
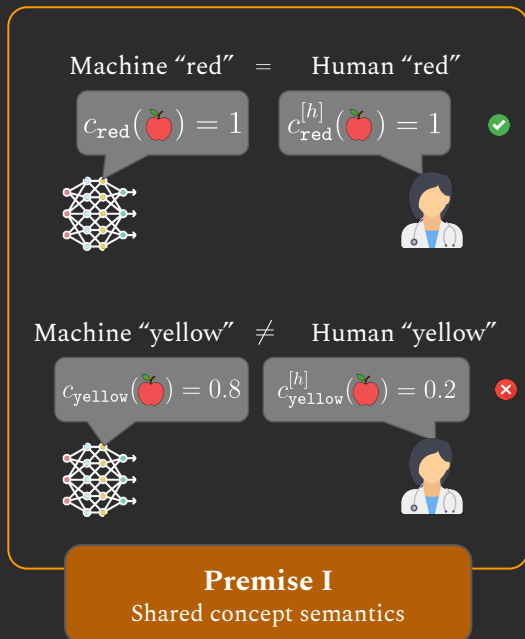
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



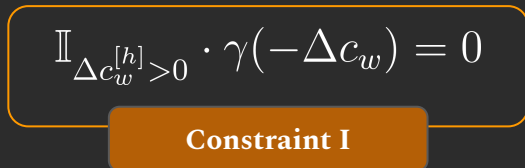
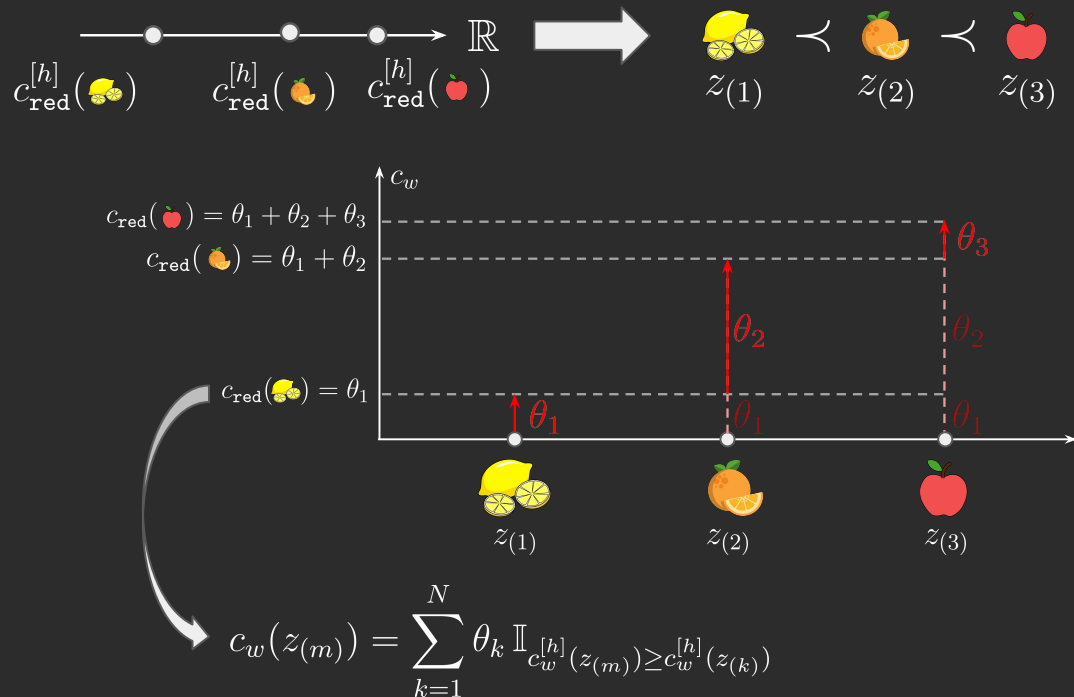
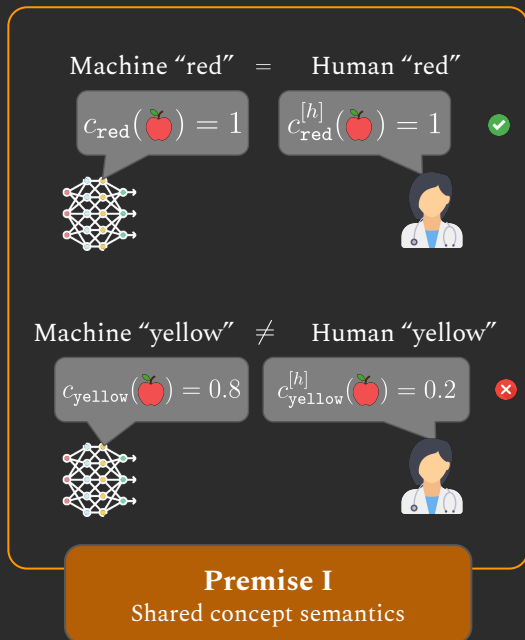
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



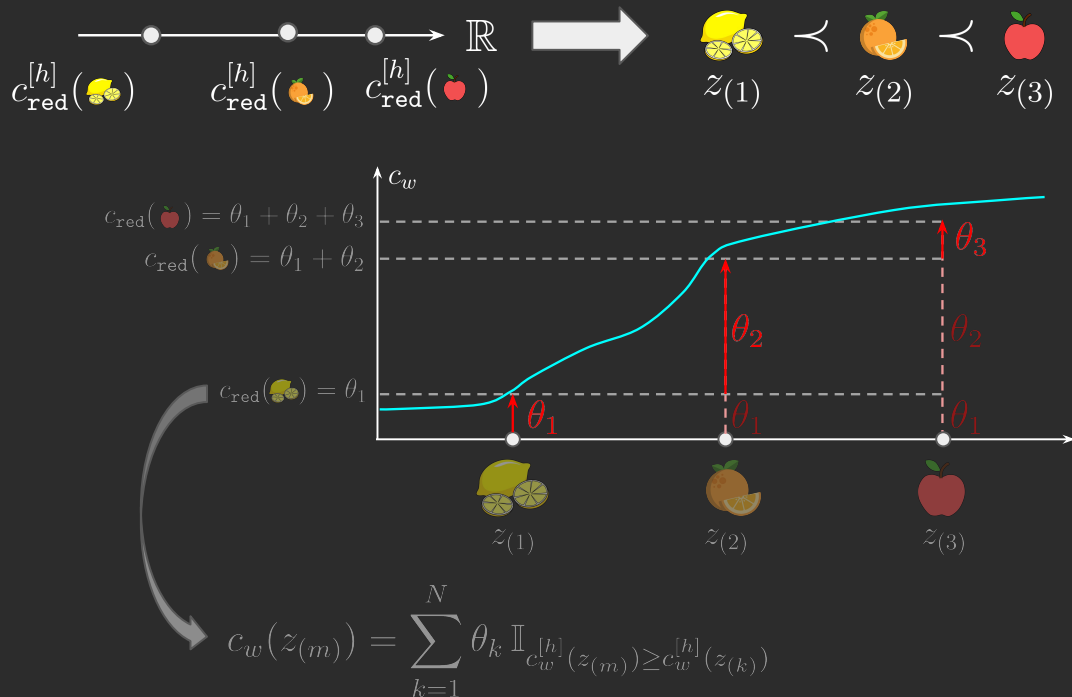
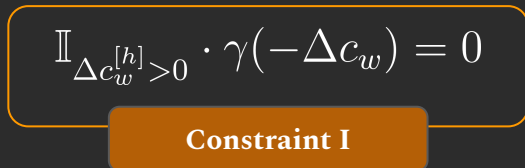
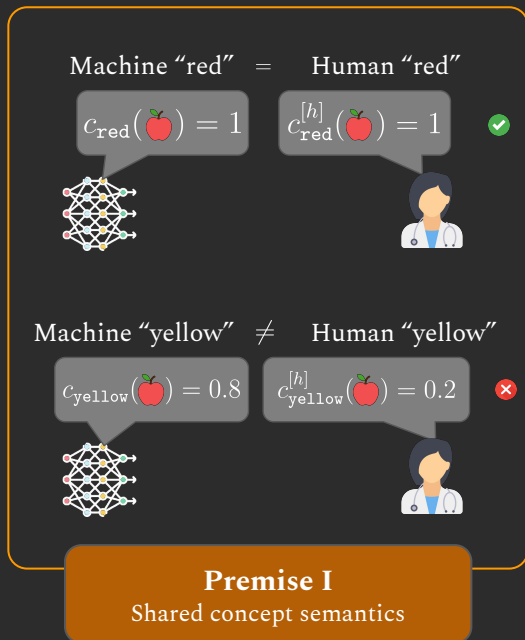
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



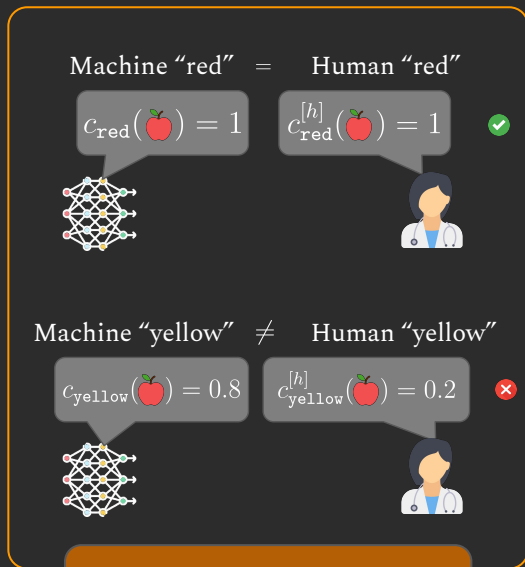
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



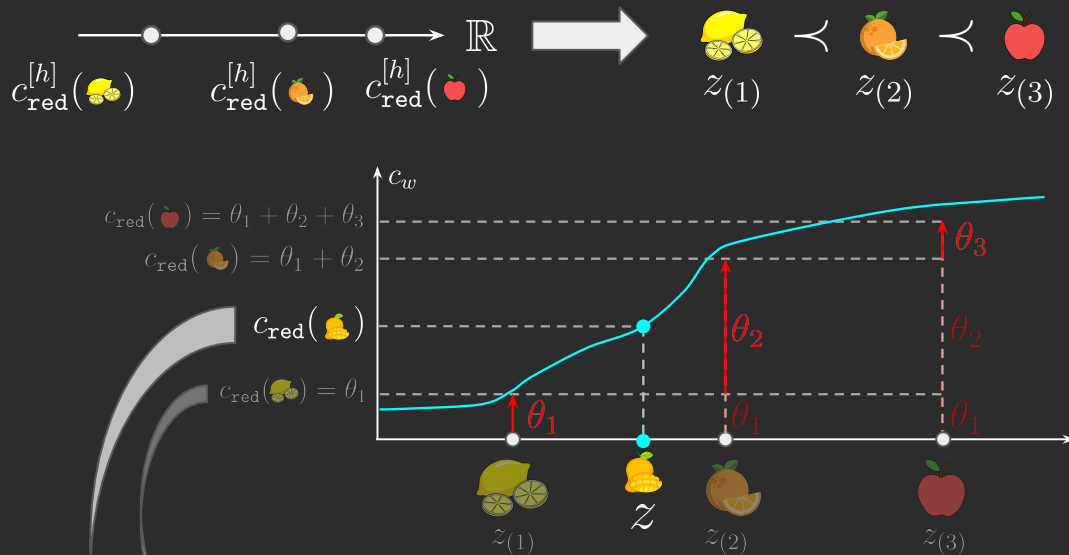
Premise I $\rightarrow$ Symmetry I $\rightarrow$ Constraint I $\rightarrow$ Architecture I



Premise I
Shared concept semantics

$$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$$

Constraint I

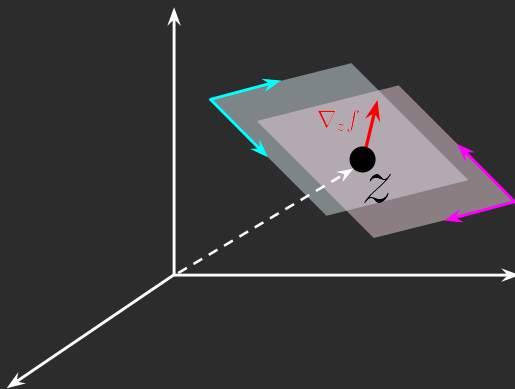
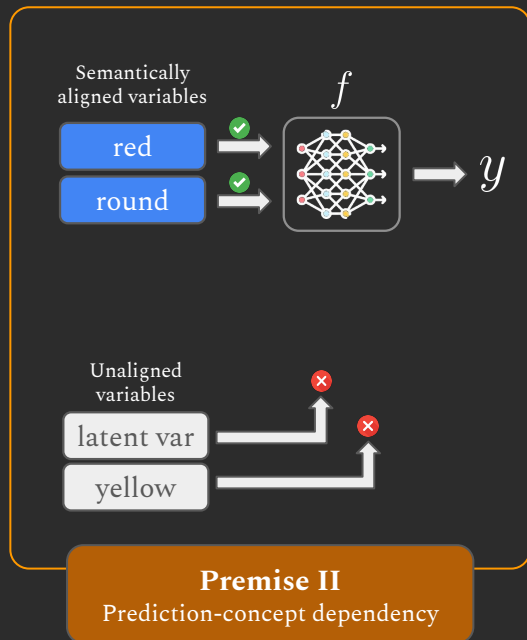


$$c_w(z(m)) = \sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z(m)) \geq c_w^{[h]}(z(k))}$$

$$c_w(z) = \sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right)$$

"Instance-based model"

Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II $\rightarrow$ Architecture II



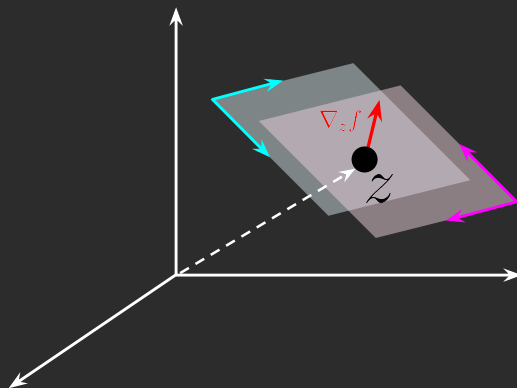
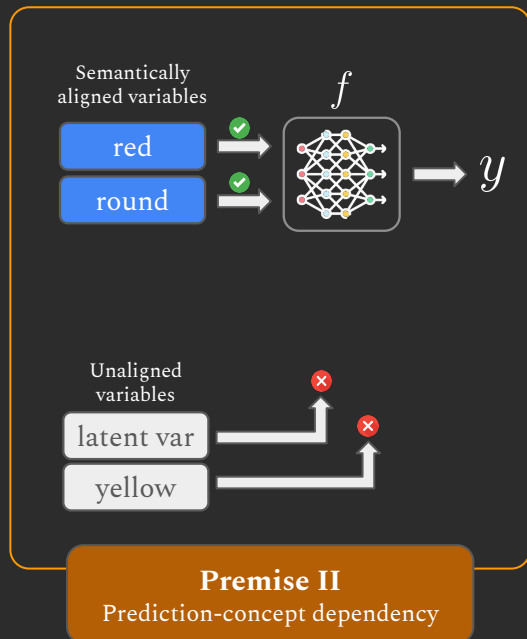
$$f = \phi(c_1, \dots, c_l)$$

“Concept bottleneck”

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

Constraint II

Premise II $\rightarrow$ Symmetry II $\rightarrow$ Constraint II $\rightarrow$ Architecture II



$$f = \phi(c_1, \dots, c_l)$$

“Concept bottleneck”

Proof using the chain rule

$$\nabla_z(\phi \circ c) = \frac{\partial \phi}{\partial c} \nabla_z c$$

$$f = \phi(c) \iff \forall z \in Z, \nabla_z f = \frac{\partial \phi}{\partial c} \nabla_z c$$

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

Constraint II

Premise III $\rightarrow$ Symmetry III $\rightarrow$ Constraint III $\rightarrow$ Architecture III

I understand only
polynomials of
degree < 3



✓  $= c_{\text{red}}^2 - c_{\text{round}}$

✗  $= 2c_{\text{red}}^3 + 1$

Premise III
Bounded reasoning

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Constraint III

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$



Solve the differential equation

$$\phi_{\theta}(c) = \sum_{k=1}^n \theta_k \psi_k(c), \quad \phi_{\theta}(c) \in \ker(K^{[h]})$$

"General solution"

Premise III → Symmetry III → Constraint III → Architecture III

I understand only
polynomials of
degree < 3



✓  = $c_{\text{red}}^2 - c_{\text{round}}$

✗  = $2c_{\text{red}}^3 + 1$

Premise III
Bounded reasoning

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Constraint III

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$



Solve the differential equation

$$\phi_\theta(c) = \sum_{k=1}^n \theta_k \psi_k(c), \quad \phi_\theta(c) \in \ker(K^{[h]})$$

"General solution"

Case	PDE $K^{[h]}$	1st Order PDE System	General Solution
Linear	$\nabla_c^2 \phi = 0$	$\begin{cases} \nabla_c \phi = u_2 \\ \nabla_c u_2 = 0 \end{cases}$	$\phi(c) = \theta_1^\top c + \theta_0$
Exponential	$\nabla_c \phi - \theta^\top \phi = 0$	$\nabla_c (\ln \phi) = \theta^\top$	$\phi(c) = e^{\theta^\top c + \theta_0}$
Piece-wise Constant	$\nabla_c \phi = \sum_i \theta_i \delta(c - c_i)$	$\nabla_c \phi = \sum_i \theta_i \delta(c - c_i)$	$\phi(c) = \sum_i \theta_i \sum_{j=1}^l H(c_j - c_{i,j}) + \theta_0$
Periodic	$\nabla_c^2 \phi + \omega^2 \phi = 0$	$\begin{cases} \nabla_c \phi = u_2 \\ \nabla_c u_2 = -\omega^2 \phi I_{m \times m} \end{cases}$	$\phi(c) = \theta_1 \cos(k^\top c) + \theta_0 \sin(k^\top c)$

Interpretable architectures

$$c_w(z) = \sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right) \quad f = \phi(c_1, \dots, c_l) \quad \phi_\theta(c) = \sum_{k=1}^n \theta_k \psi_k(c), \quad \phi_\theta(c) \in \ker(K^{[h]})$$

Architectures

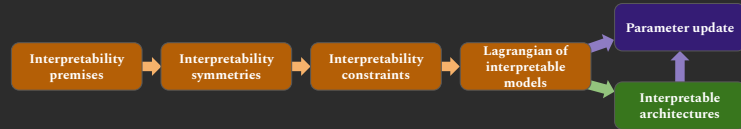
$$\underbrace{f = \phi_\theta^{[h]}}_{\text{Constraint II}} \left(\underbrace{\left[\sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right) \right]_{w=1}^l}_{\text{Constraint I}} \right), \quad \underbrace{\phi_\theta^{[h]} \in \ker(K^{[h]})}_{\text{Constraint III}}$$

General solution

$$L(\theta_{c,\phi}, \mathcal{D}, y, K^{[h]}) = T - \mathcal{L} \left(\phi_\theta^{[h]} \left(\left[\sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right) \right]_{w=1}^l \right), y \right), \quad \phi_\theta^{[h]} \in \ker(K^{[h]})$$

Lagrangian

The Standard Interpretable Model



Premises

Shared concept semantics

Prediction-concept dependency

Bounded reasoning

Symmetries

$$\chi(\mathbf{g}_w \circ c_w^{[h]}) = \chi(c_w^{[h]})$$

$$\exists \mathbf{g}_c \in \mathfrak{G}_c \text{ such that } \mathbf{g}_c \cdot (\nabla_z f)^\top = (\nabla_z f)^\top$$

$$K^{[h]}(\mathbf{g}(\phi(c)), \dots, \nabla^{(\nu)} \mathbf{g}(\phi(c))) = \mu(c, \phi(c)) K^{[h]}(\phi, \dots, \nabla_c^{(\nu)} \phi) = 0$$

Constraints

$$\mathbb{I}_{\Delta c_w^{[h]} > 0} \cdot \gamma(-\Delta c_w) = 0$$

$$1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} = 0$$

$$K^{[h]}(\phi, \dots, \nabla_c^{(n)} \phi) = 0$$

Lagrangian

$$L(\theta_{f,c,\phi}, \mathcal{D}, y, K^{[h]}) = T - V = \underbrace{\sum_{i \in \{f,c,\phi\}} \frac{1}{2} m (\partial_i \theta_i)^T (\partial_i \theta_i)}_{\text{parameter dynamics}} - \underbrace{\mathcal{L}(f(z; \theta_f), y)}_{\text{prediction error}} - \underbrace{\lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z_i, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c))}_{\text{Constraint I}} - \underbrace{\lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right)}_{\text{Constraint II}} - \underbrace{K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{(n)} \phi(c(z; \theta_c); \theta_\phi))}_{\text{Constraint III}}$$

Equations

of motion

$$m \frac{\partial^2 \theta_f}{\partial t^2} = -\nabla_{\theta_f} \mathcal{L}(f(z; \theta_f), y) - \lambda_2 \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right)$$

$$\begin{aligned} m \frac{\partial^2 \theta_c}{\partial t^2} = & -\lambda_1 \sum_{w=1, \dots, m} \mathbb{I}_{\Delta c_w^{[h]} > 0} \nabla_{\theta_c} \gamma(-\Delta c_w(z, z_i; \theta_c)) \\ & - \lambda_2 \nabla_{\theta_c} \left(1 - \frac{\|Q_c^\top Q_f\|_F^2}{\text{rank}(\nabla_z f)} \right) \\ & - \lambda_3 \nabla_{\theta_c} K^{[h]}(\phi(c; \theta_\phi), \dots, \nabla_c^{(n)} \phi(c; \theta_\phi)) \end{aligned}$$

$$m \frac{\partial^2 \theta_\phi}{\partial t^2} = -\lambda_3 \nabla_{\theta_\phi} K^{[h]}(\phi(c; \theta_\phi), \dots, \nabla_c^{(n)} \phi(c; \theta_\phi))$$

Architectures

$$c_w(z) = \sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right)$$

$$f = \phi(c_1, \dots, c_l)$$

$$\phi_\theta(c) = \sum_{k=1}^n \theta_k \psi_k(c), \quad \phi_\theta(c) \in \ker(K^{[h]})$$

General

solution

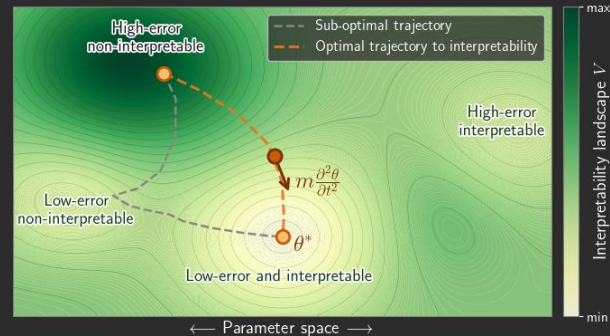
$$\underbrace{f = \phi_\theta^{[h]}}_{\text{Constraint II}} \left(\underbrace{\left[\sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right) \right]_{w=1}^l}_{\text{Constraint I}} \right), \quad \underbrace{\phi_\theta^{[h]} \in \ker(K^{[h]})}_{\text{Constraint III}}$$

Lessons learnt

Interpretability “in the loss”

$$L(\theta_{f,c,\phi}, \mathcal{D}, y, K^{[h]}) = T - V = \underbrace{\sum_{i \in \{f,c,\phi\}} \frac{1}{2} m(\partial_i \theta_i)^T (\partial_i \theta_i)}_{\text{parameter dynamics}} - \underbrace{\mathcal{L}(f(z; \theta_f), y)}_{\text{prediction error}} - \underbrace{\lambda_1 \sum_{w=1}^m \sum_{z_i \in \mathcal{D}} \mathbb{I}_{\Delta c_w^{[h]}(z, z_i) > 0} \cdot \gamma(-\Delta c_w(z, z_i; \theta_c))}_{\text{Constraint I}} - \underbrace{\lambda_2 \left(1 - \frac{\|Q_c^T Q_f\|_F^2}{\text{rank}(\nabla_z f)}\right)}_{\text{Constraint II}} - \underbrace{K^{[h]}(\phi(c(z; \theta_c); \theta_\phi), \dots, \nabla_c^{[m]} \phi(c(z; \theta_c); \theta_\phi))}_{\text{Constraint III}}$$

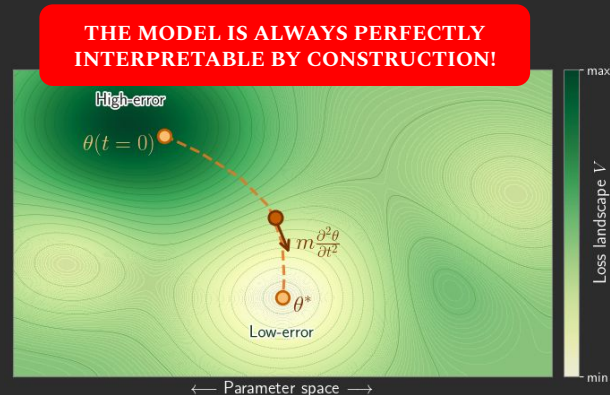
Lagrangian #1



Interpretability “in the architecture”

$$L(\theta_{c,\phi}, \mathcal{D}, y, K^{[h]}) = T - \mathcal{L} \left(\phi_\theta^{[h]} \left(\left[\sum_{i=1}^N \beta_i(z) \left(\sum_{k=1}^N \theta_k \mathbb{I}_{c_w^{[h]}(z_i) \geq c_w^{[h]}(z_k)} \right) \right]_{w=1}^l \right), y \right), \phi_\theta^{[h]} \in \ker(K^{[h]})$$

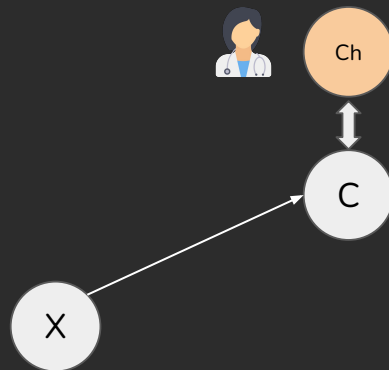
Lagrangian #2



Lessons learnt

Premise I

Shared concept semantics



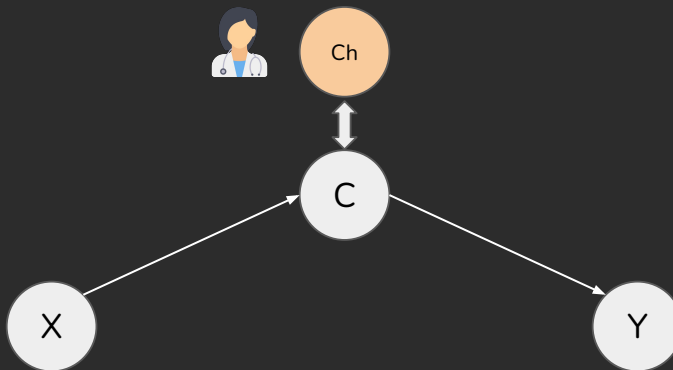
Lessons learnt

Premise I

Shared concept semantics

Premise II

Prediction concept dependency



Lessons learnt

Premise I

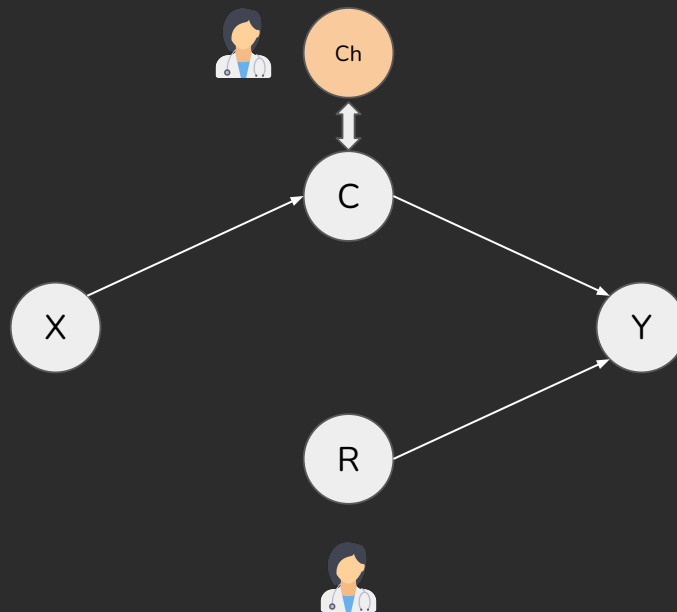
Shared concept semantics

Premise II

Prediction concept dependency

Premise III

Bounded reasoning



Lessons learnt

Premise I

Shared concept semantics

Premise II

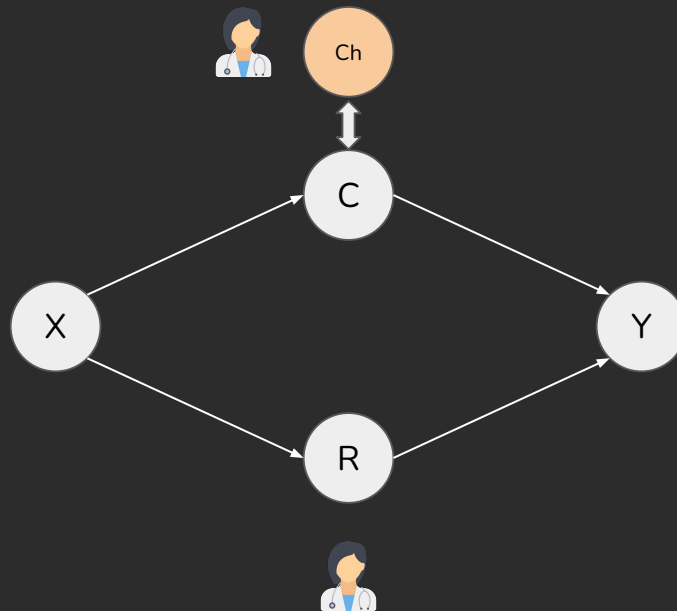
Prediction concept dependency

Premise III

Bounded reasoning

Universal approximation

Residual path



Tutorial Outline

I. Interpretability theory



~50 mins

II. Concept representations



~40 mins

III. Task predictions



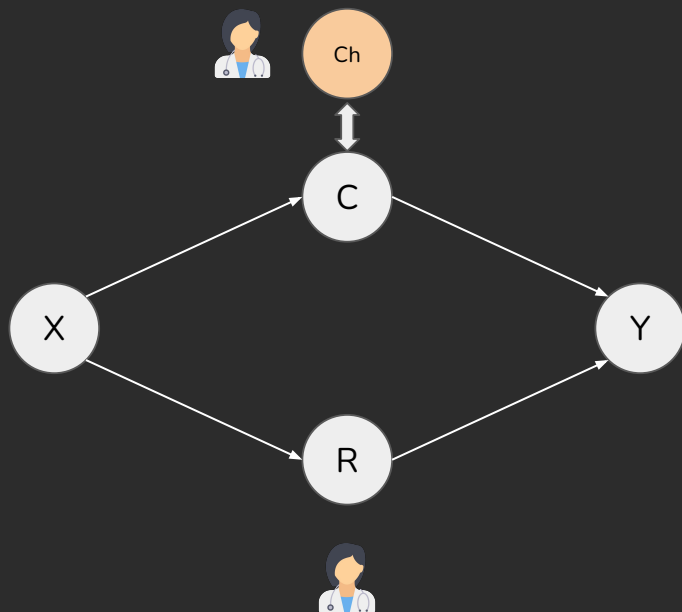
~50 mins

IV. PyTorch Concepts



~20 mins

The Probabilistic Concept-based Model Blueprint



$$p(c, r, y | x) = p(y | c, r) p(c | x) p(r | x)$$

$p(c | x)$ - Concept Encoder

$p(r | x)$ - Prediction rule selection

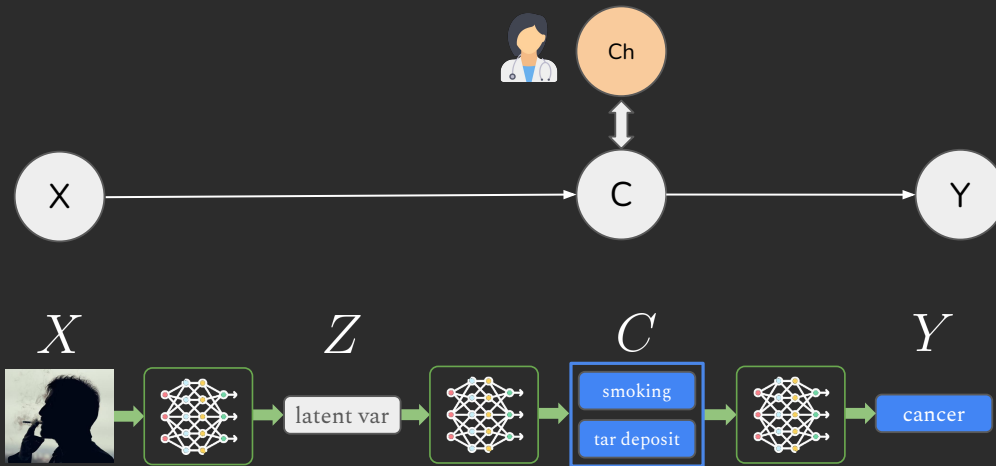
$p(y | c, r)$ - Task Predictor



Concept Bottleneck Models (CBMs)

A **Concept Bottleneck Model (CBM)** [1]:

$$p(y, c \mid x) = p(y \mid c)p(c \mid x)$$





Concept Bottleneck Models (CBMs)

A **Concept Bottleneck Model (CBM)** [1]:

1. Access to k binary concept labels $\mathbf{C}$ for each training sample.

$$\left(\begin{array}{c} \mathbf{x} \\ \mathbf{c} \\ y \end{array} \right) \sim \mathcal{D}$$

The diagram illustrates the input data for a Concept Bottleneck Model. It shows a vector $\mathbf{x}$ (represented by an X-ray image of a knee joint), a vector $\mathbf{c}$ (represented by a list of concept labels: $c_{\text{sclerosis}}$, c_{spurs} , $\vdots$, $c_{\text{osteophytes}}$, $c_{\text{narrow_joint}}$), and a scalar y (represented by the number 2). The entire input is drawn from a distribution $\mathcal{D}$.

Concept Bottleneck Models (CBMs)



A **Concept Bottleneck Model (CBM)** [1] exploits this idea by assuming:

1. Access to k binary concept labels C for each training sample.
2. Each concept in C can be independently predicted from X

$$p(c|x) = \prod_i p(c_i|x)$$

Concept Bottleneck Models (CBMs)



A **Concept Bottleneck Model (CBM)** [1] exploits this idea by assuming:

1. Access to k binary concept labels C for each training sample.
2. Each concept in C can be independently predicted from X
3. The downstream task can be perfectly predicted from C alone.

$$p(y|c, r) = p(y|c)$$

How to train your CBM

Given a training set $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(i)})\}_{j=1}^N$, we can train a CBM in one of three ways:

1. Max-loglikelihood (or “independent” train)

$$\max_{\theta, \phi} \sum_{\hat{x}, \hat{y}, \hat{c} \sim \mathcal{D}} \log p_{\theta, \phi}(\hat{y}, \hat{c} | \hat{x})$$

$$\max_{\theta, \phi} \sum_{\hat{x}, \hat{y}, \hat{c} \sim \mathcal{D}} \log p_{\theta}(\hat{y} | \hat{c}) + \log p_{\phi}(\hat{c} | \hat{x})$$

How to train your CBM

Given a training set $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(i)})\}_{j=1}^N$, we can train a CBM in one of three ways:

2. Sequential:

$$\phi^* = \operatorname{argmax}_{\phi} \sum_{\hat{x}, \hat{c} \sim \mathcal{D}} \log p_{\phi}(\hat{c} | \hat{x})$$

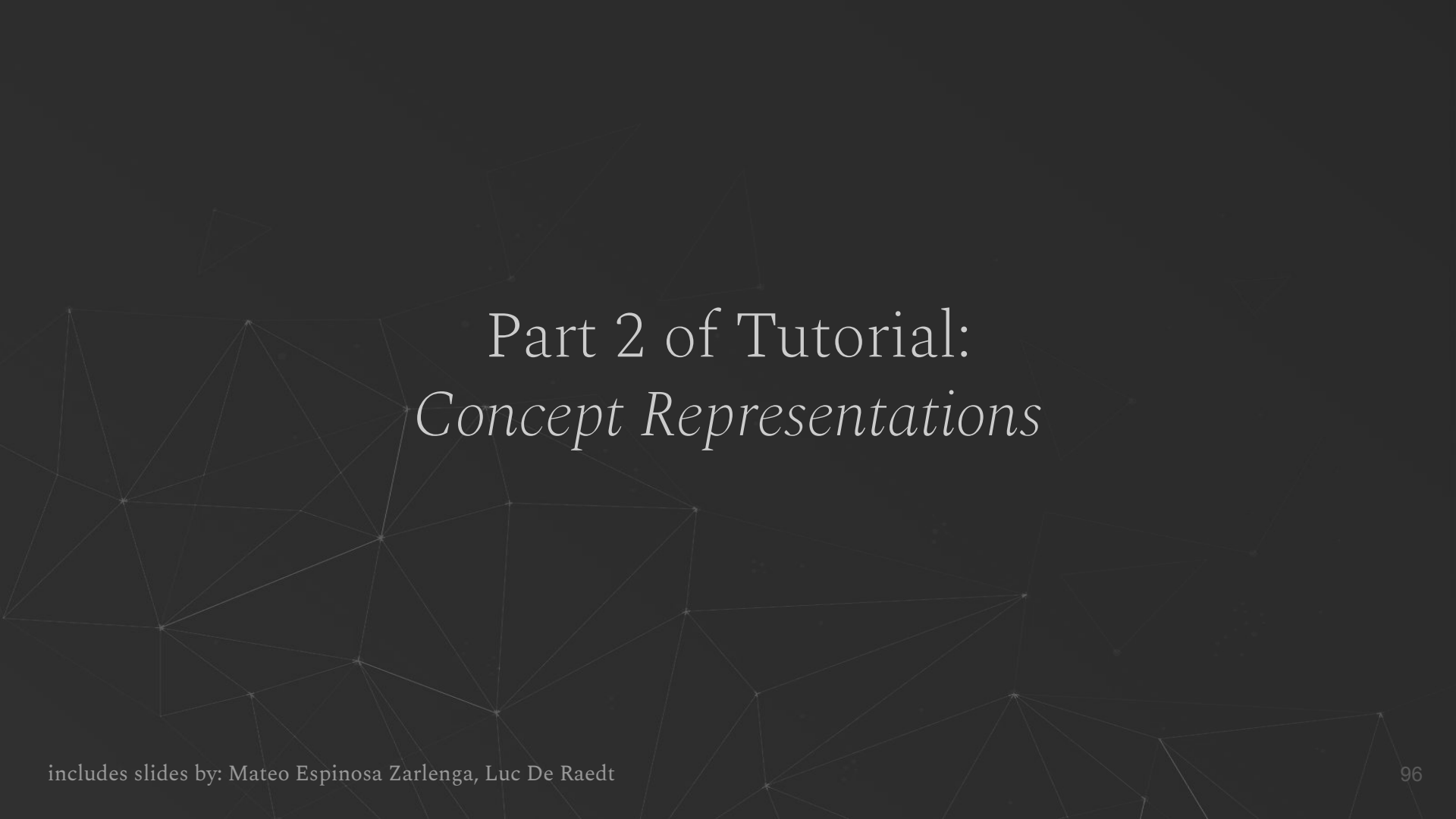
$$\theta^* = \operatorname{argmax}_{\theta} \sum_{\hat{x}, \hat{y} \sim \mathcal{D}, c \sim p_{\phi^*}(c | \hat{x})} \log p_{\theta}(\hat{y} | c)$$

How to train your CBM

Given a training set $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(i)})\}_{j=1}^N$, we can train a CBM in one of three ways:

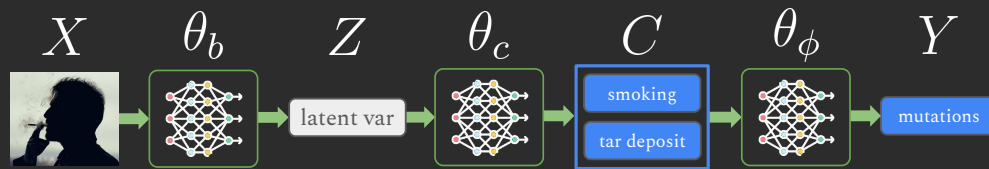
3. Joint:

$$\max_{\theta, \phi} \sum_{\hat{x}, \hat{y}, \hat{c} \sim \mathcal{D}, c \sim p_{\phi}(c | \hat{x})} \log p_{\theta}(\hat{y} | c) + \lambda \log p_{\phi}(\hat{c} | \hat{x})$$

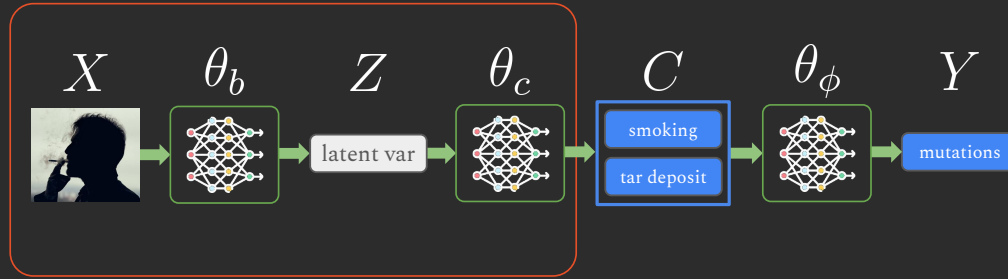


Part 2 of Tutorial: *Concept Representations*

includes slides by: Mateo Espinosa Zarlenga, Luc De Raedt

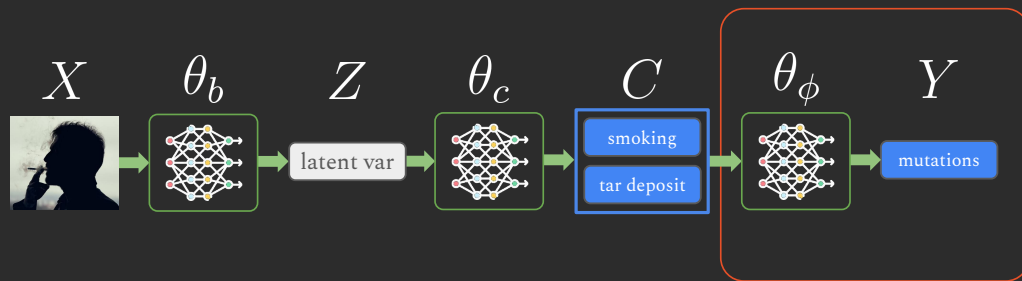


But concepts are actually more than just a bottleneck
They are a **semantic API** consumed by (at least) three actors:



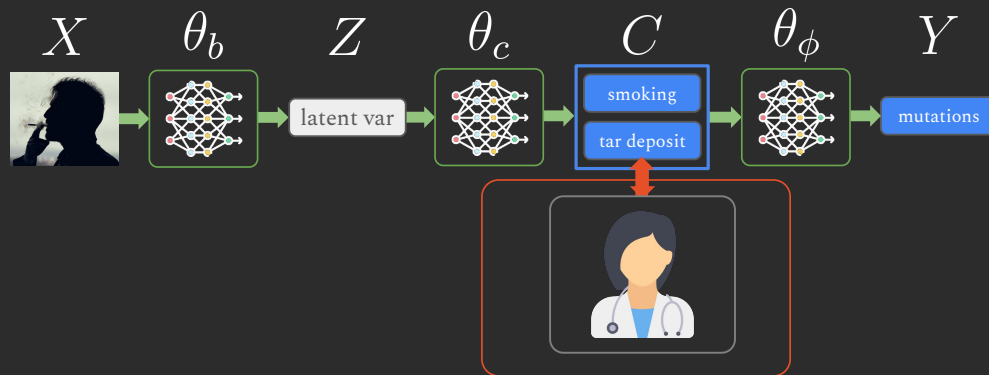
But concepts are actually more than just a bottleneck
They are a **semantic API** consumed by (at least) three actors:

- a machine learning concept encoder model



But concepts are actually more than just a bottleneck
They are a **semantic API** consumed by (at least) three actors:

- a machine learning encoder model
- a task predictor



But concepts are actually more than just a bottleneck
They are a **semantic API** consumed by (at least) three actors:

- a machine learning encoder model
- a task predictor
- a target user

This semantic interface needs a contract

- How do define the interface such that these three actors communicate efficiently?
- Natural language is appealing, but it weakens our control over the interaction
 - a key reason for using interpretable models.
 - but see [1]
- **Formal languages**, e.g. logic, form a middle ground:
 - precise enough for machines to manipulate
 - still close enough to human meaning

The formal language contract

We need to specify:

1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

e.g. predicates = {sunny, tall, funny, fatherOf}

entities = {bart, homer, lisa, ...}

arities: sunny/0, tall/1, funny/1, fatherOf/2

The formal language contract

We need to specify:

- 1) Syntax
- 2) Semantics:
 - a) what values do we allow?
 - b) which values do we assign at each symbol?

e.g. True/False, $[0,1]$, $(-\infty, +\infty)$, $\mathbb{R}^n$

sunny = 0.8

fatherOf(homer,bart) = True

The formal language contract

We need to specify:

- 1) Syntax
- 2) Semantics
- 3) Connectives / Operators / Algebra
 - a) how do we compose the symbols to create more complex sentences?

The Concept Bottleneck Model contract

1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?

The Concept Bottleneck Model contract

1) Syntax:

- a) Vocabulary of symbols (predicates and entities)

Given and Fixed

- b) The arities of the predicates

Only properties (0-arity; propositional)

2) Semantics:

- a) what values do we allow?

Unclear, at the base Boolean

- b) which values do we assign at each symbol?

Fully Supervised

The Concept Bottleneck Model contract

1) Syntax:

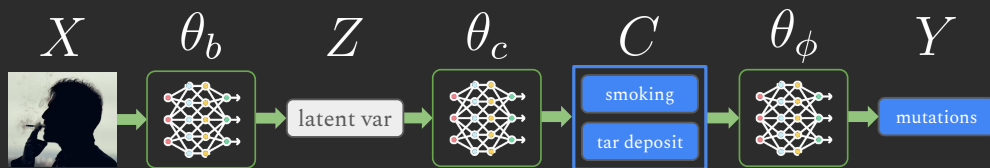
- a) Vocabulary of symbols (predicates and entities)

Given and Fixed

- b) The arities of the predicates

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?



The Concept Bottleneck Model contract

1) Syntax:

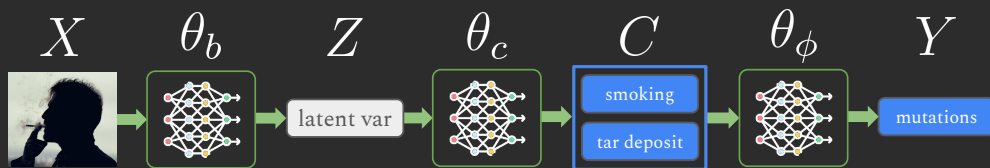
- a) Vocabulary of symbols (predicates and entities)

Given and Fixed

- b) The arities of the predicates

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?



Failure: **Insufficient vocabulary**

Failure: Incomplete vocabulary

Repair: automated concept discovery



This can be turned into a **fully automated pipeline** if we exploit language models as knowledge bases for extracting concept descriptions:



“List the most important features for recognizing something as a {class}:”

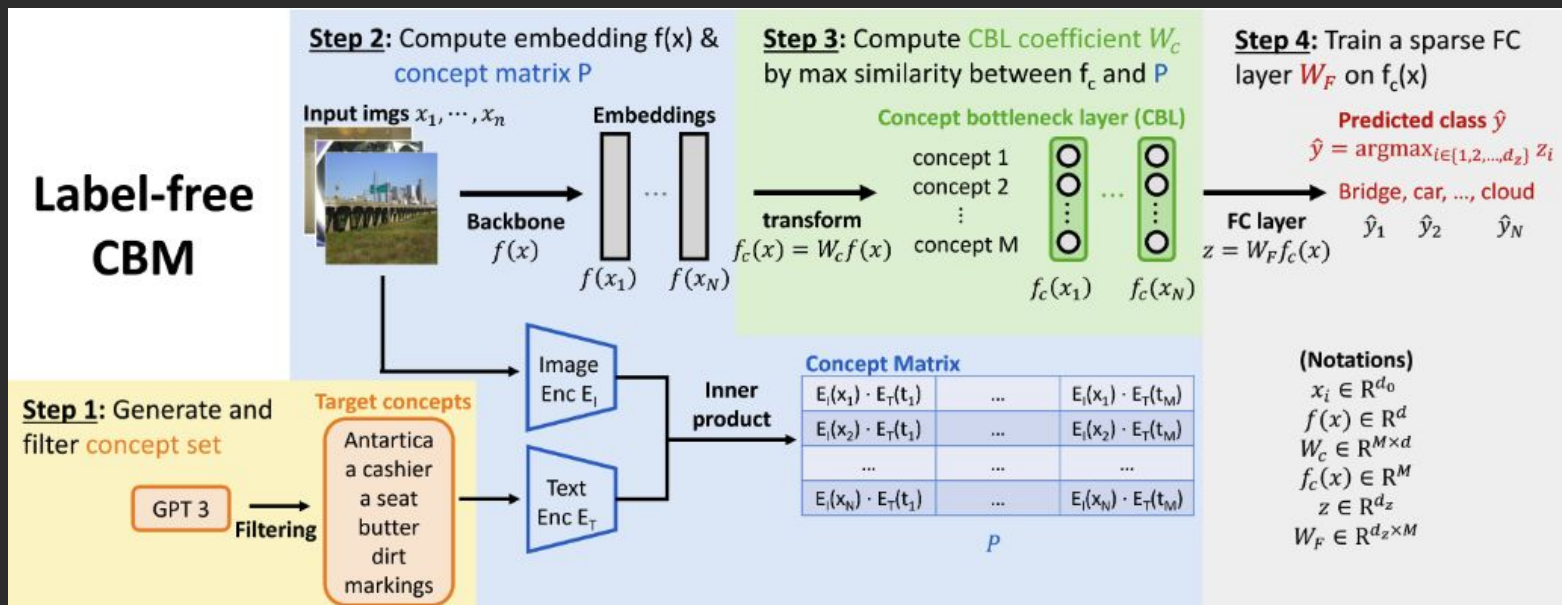
[1] Oikarinen et al. "Label-Free Concept Bottleneck Models." ICLR (2023).

[2] Yang, Yue, et al. "Language in a bottle: Language model guided concept bottlenecks for interpretable image classification." IEEE/CVF 2023.



Automating concept discovery

Label-free CBMs achieve this by introducing *filtering rules* and a *projection matrix* that produces sparse and discriminative scores aligned to CLIP's similarity scores





Automating concept discovery

Alternatively, one may filter the concept bank via submodular optimization:

$$\mathcal{F}(C_y) = \underbrace{\alpha \cdot \sum_{c \in C_y} D(c)}_{\text{discriminability}} + \underbrace{\beta \cdot \sum_{c_1 \in S_y} \max_{c_2 \in C_y} \phi(c_1, c_2)}_{\text{coverage}}$$

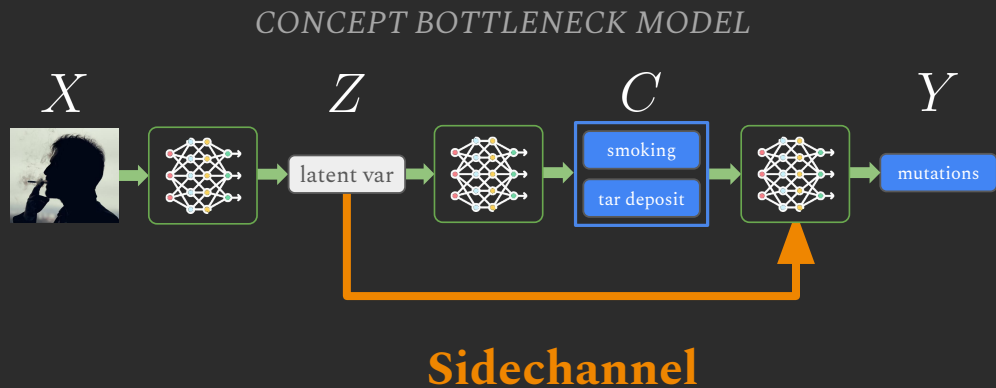
Concepts to select for class y

How good is the concept at discriminating only a handful of task labels?

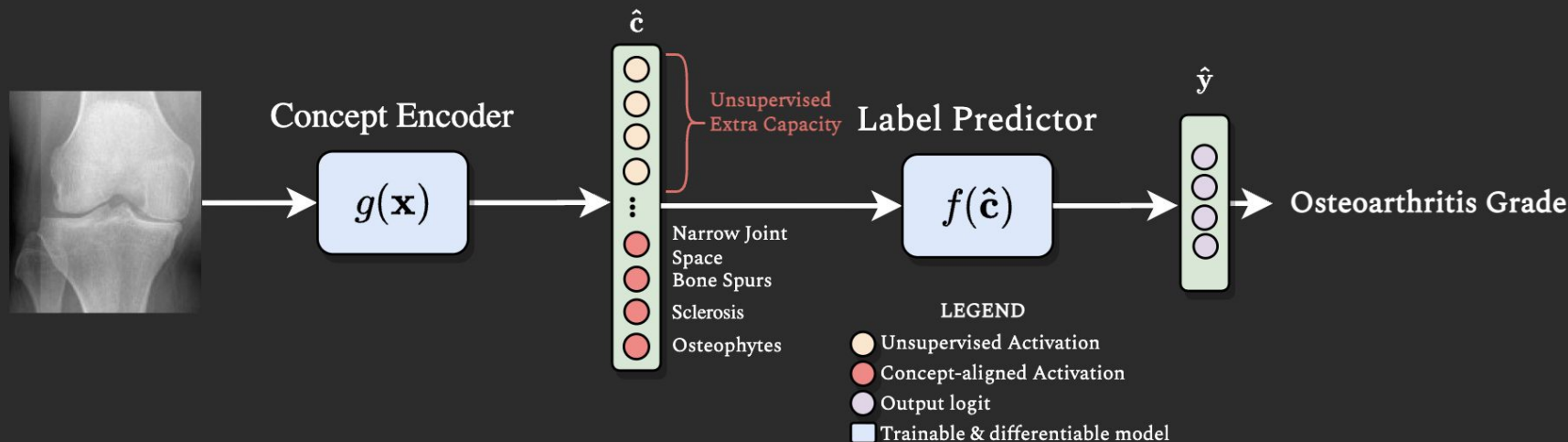
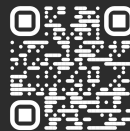
How good does the filtered concept set represent the original set of candidates?

Failure: Incomplete vocabulary

Repair: Adding extra (unsupervised) capacity



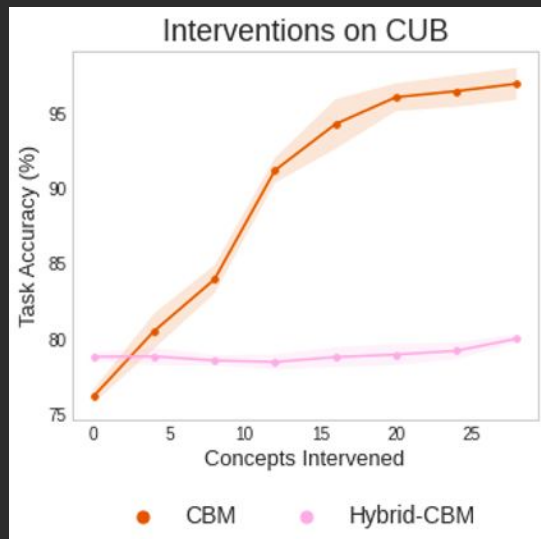
Failure: Incomplete vocabulary
Repair: Adding extra (unsupervised) capacity



A **"Hybrid" CBM** adds extra unsupervised capacity to the CBM's bottleneck

Solution attempt: unsupervised capacity

Problem: intervenability suffers when unsupervised capacity is introduced

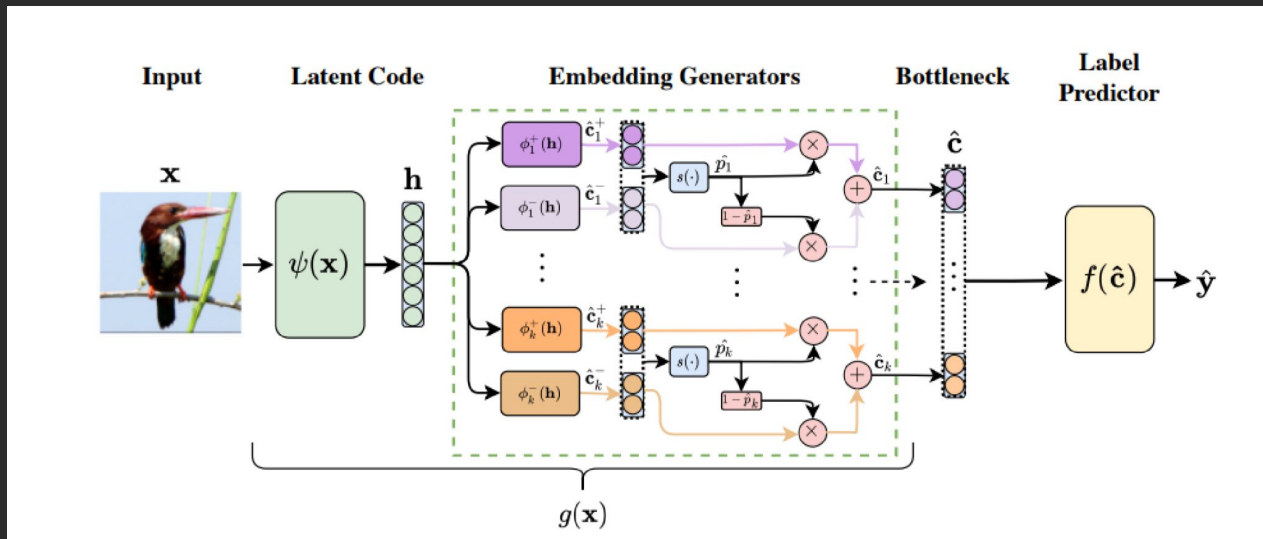


Solution idea: structured high-dimensional vectors

Idea: can we use structured extra capacity instead?

Overcoming incompleteness: concept embeddings

For each training concept, let's represent extra structured embedding representations of the concept



Learning dynamic concept embeddings

For example, we can decompose the concept representation $\hat{\mathbf{c}}_i$ as the mixture of two exogenous representations $\{\hat{\mathbf{c}}_i^+(\mathbf{x}), \hat{\mathbf{c}}_i^-(\mathbf{x})\}$:

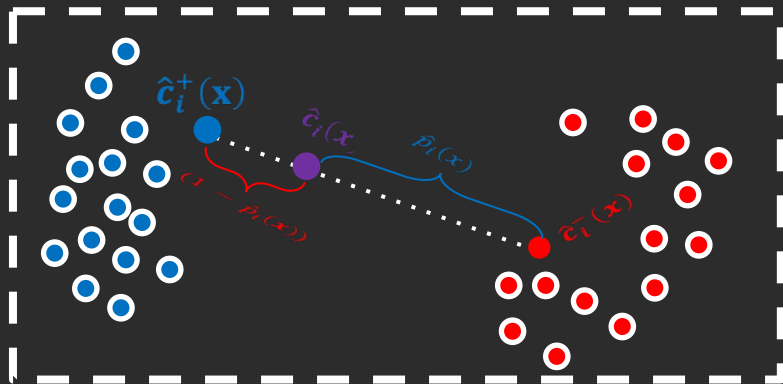
$$\hat{\mathbf{c}}_i(\mathbf{x}) = \hat{p}_i(\mathbf{x}) \hat{\mathbf{c}}_i^+(\mathbf{x}) + (1 - \hat{p}_i(\mathbf{x})) \hat{\mathbf{c}}_i^-(\mathbf{x})$$



"Positive" concept embeddings



"Negative" concept embeddings



The Concept Bottleneck Model contract

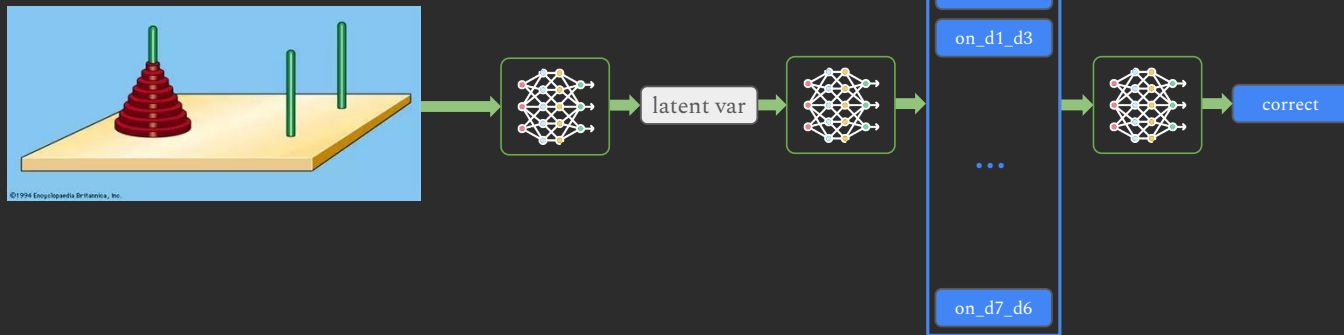
1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

Only flat concepts (propositional): no objects / entities

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?



The Concept Bottleneck Model contract

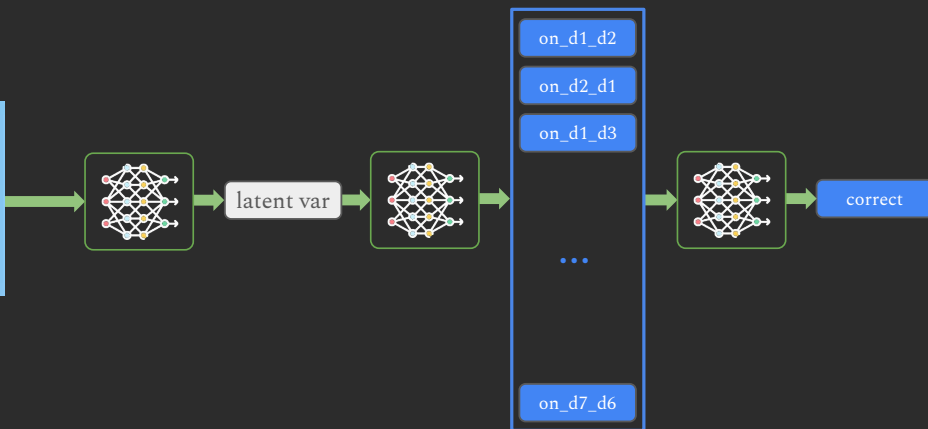
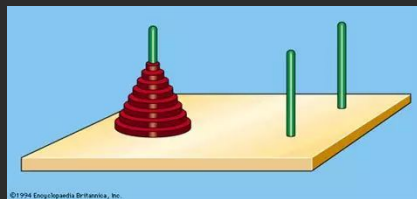
1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

Only flat concepts (propositional): no objects / entities

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?

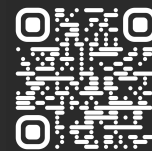


Failure:

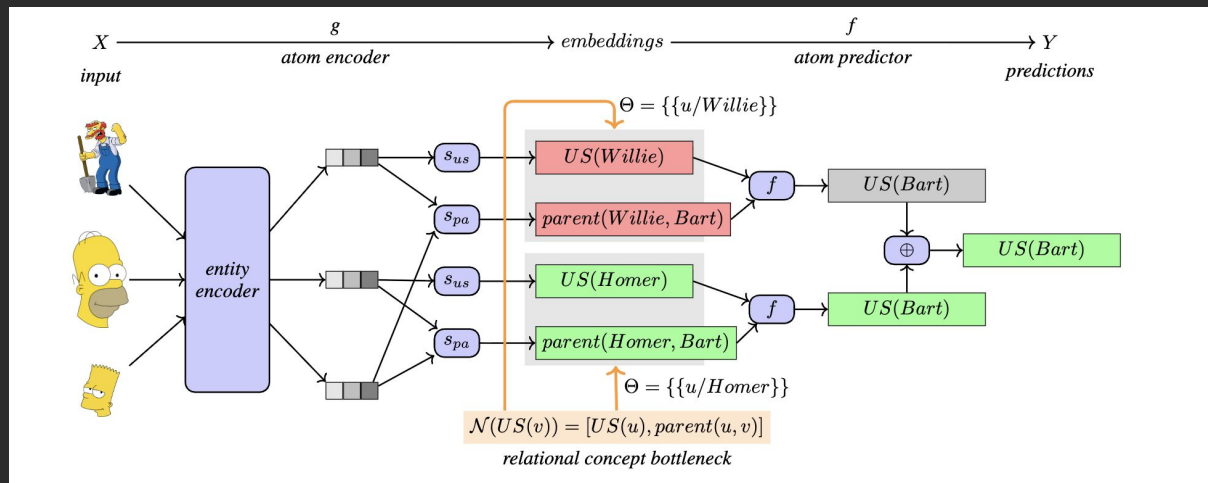
Limited
compositional
generalization

Failure: limited compositional generalization

Repair: Relational vocabulary

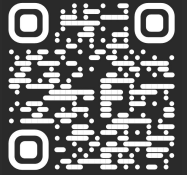


- Concept-encoders become relational

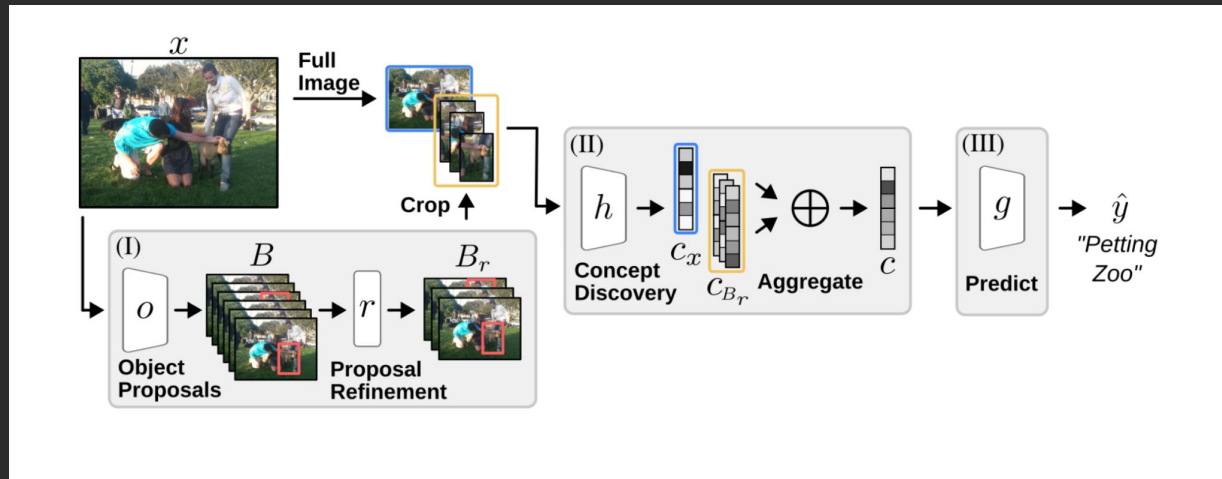


Failure: limited compositional generalization

Repair: object centric bottlenecks

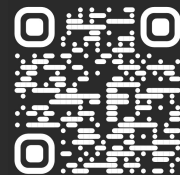


- Expand relational bottlenecks with an entity discovery model:
 - Mask-RCNN, SAM

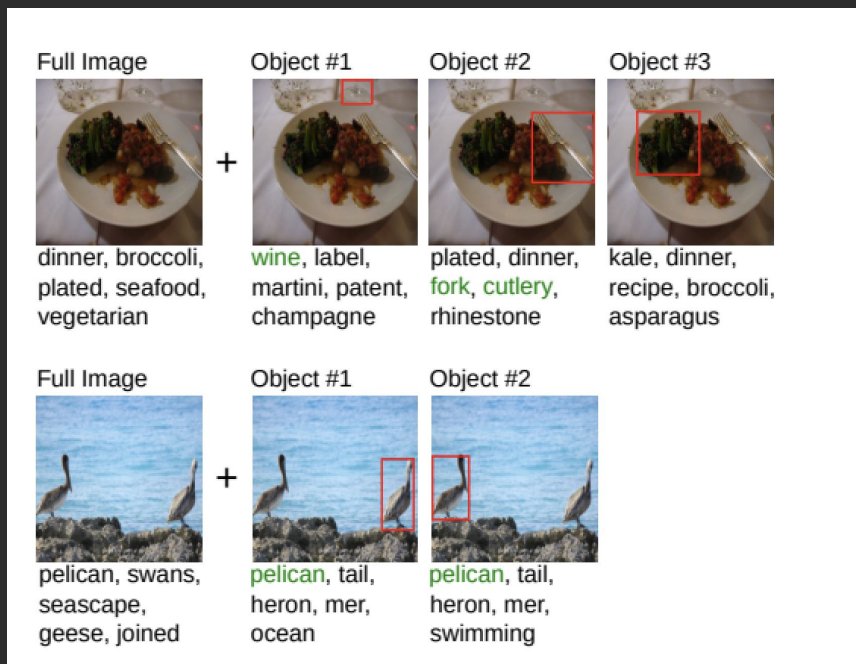


Failure: limited compositional generalization

Repair: object centric bottlenecks



- Expand relational bottlenecks with an entity discovery model:
 - Mask-RCNN, SAM



The Concept Bottleneck Model contract

1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

2) Semantics:

- a) what values do we allow?

Unclear

- b) which values do we assign at each symbol?

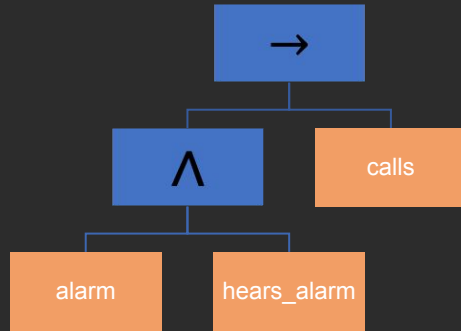


Semantics

- Given a concept (like **happy(homer)**):
- semantics ask questions connected to:
 - is it **True/False**?
 - **how (much)** happy is this?
 - **how likely** is that he is happy?
 - are there **factors** of happiness? e.g. [*pleasure=2.7, satisfaction=11.7, purpose=0.1, ...*]
 - an embedding?
- We can make a parallel with logic

Boolean Logic

$$\text{alarm} \wedge \text{hears_alarm} \rightarrow \text{calls}$$



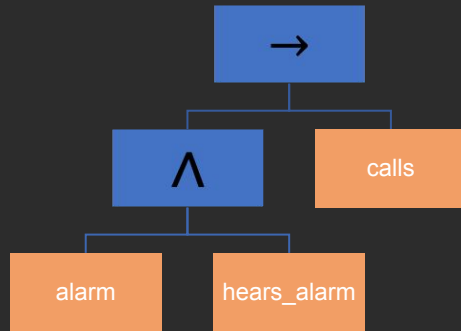
values are in $\{T,F\}$ (or $\{0,1\}$)

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

Fuzzy Logic

$$\text{alarm} \wedge \text{hears_alarm} \rightarrow \text{calls}$$



values are in $[0,1]$

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

Probabilistic Logic Semantics

- Two layer semantics
- Defines a **probability distribution** over **boolean interpretations**

$$p(x_1 = \textit{True}, x_2 = \textit{False}, \dots, x_n = \textit{True}) = 0.6$$

The Concept Bottleneck Model contract

2) Semantics:

a) what values do we allow?

Unclear

Learning: often with Binary Cross Entropy

- Implicit assumption: Independent Choice (Bernoulli concepts)

Task Prediction:

- Scores are passed to the task predictor: ~ Fuzzy Semantics

Intervention-time:

- True/False interventions: Boolean Semantics

Concept-Embedding Models:

- ?!?!?!?

The Concept Bottleneck Model contract

2) Semantics:

a) what values do we allow?

Unclear

Learning: often with Binary Cross Entropy

- Implicit assumption: Independent Choice (Bernoulli concepts)

Task Prediction:

- Scores are passed to the task predictor: ~ Fuzzy Semantics

Intervention-time:

- True/False interventions: Boolean Semantics

Concept-Embedding Models:

- ?!?!?!?

$$p(c|x) = \prod_i p(c_i|x)$$

The Concept Bottleneck Model contract

2) Semantics:

a) what values do we allow?

Unclear

Learning: often with Binary Cross Entropy

- Implicit assumption: Independent Choice (Bernoulli concepts)

Task Prediction:

- Scores are passed to the task predictor: ~ Fuzzy Semantics

Intervention-time:

- True/False interventions: Boolean Semantics

Concept-Embedding Models:

- !!!!!?

$$\theta^* = \operatorname{argmax}_{\theta} \sum_{\hat{x}, \hat{y} \sim \mathcal{D}, c \sim p_{\phi^*}(c | \hat{x})} \log p_{\theta}(\hat{y} | c)$$

↓

$$\theta^* = \operatorname{argmax}_{\theta} \sum_{\hat{x}, \hat{y} \sim \mathcal{D}} \log p_{\theta}(\hat{y} | p_{\phi^*}(c | \hat{x}))$$

Neural task predictors break concept semantics

- Misalignment between concept encoder and task
- For example, consider a probabilistic concept, $p(\text{tail}) = 0.7$
- score 0.7 $\rightarrow$ input to a neural network
- the model combines it with other concept scores in ways that no longer respect probability theory.
- Result = the original semantic meaning of the concept is lost.

The Concept Bottleneck Model contract

1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?

Fully Supervised



The Concept Bottleneck Model contract

1) Syntax:

- a) Vocabulary of symbols (predicates and entities)
- b) The arities of the predicates

2) Semantics:

- a) what values do we allow?
- b) which values do we assign at each symbol?

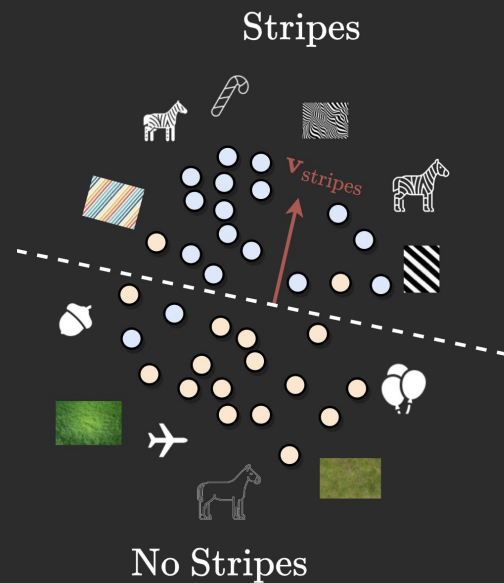
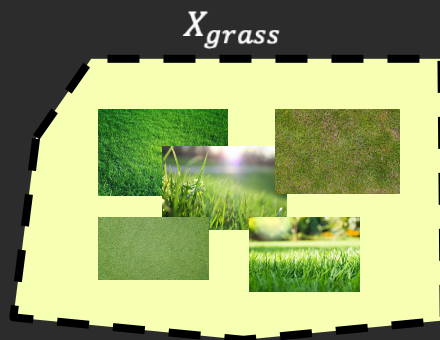
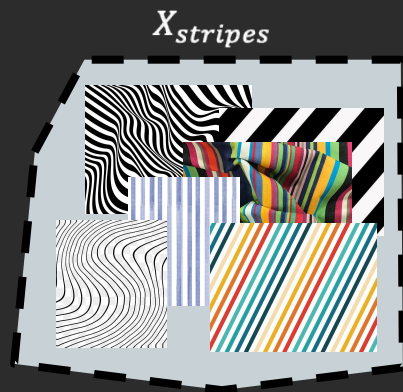
Fully Supervised



Failure: huge
labelling and
training effort

Failure: Huge training effort

Repair: Concept-activation vectors

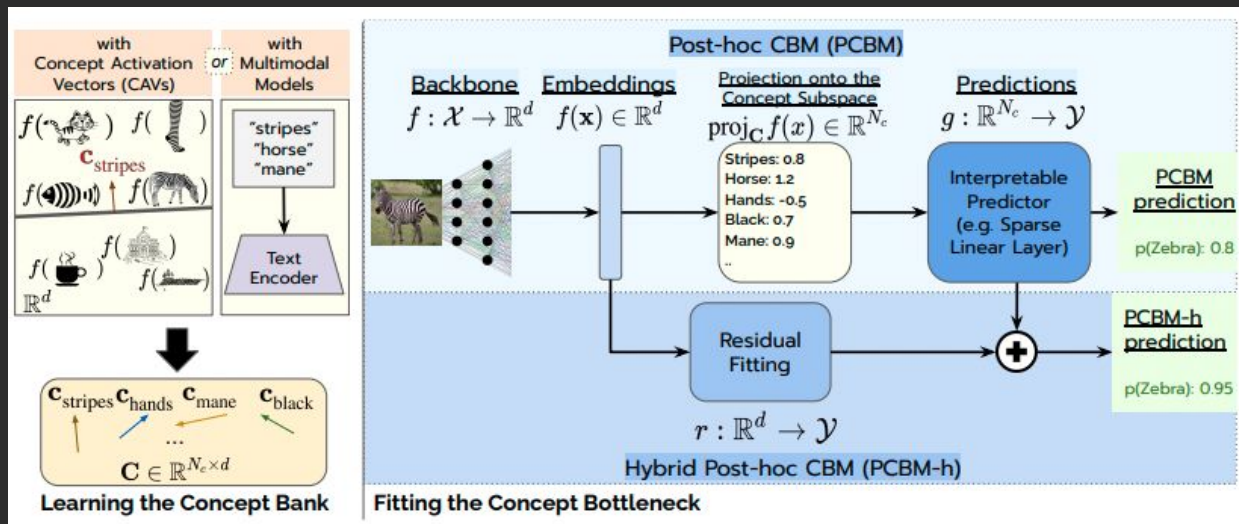




Failure: Huge training effort

Repair: Concept-activation vectors

A **Post-hoc CBM (PCBM)** uses a pretrained DNN to construct concept representations based on similarity scores from projections onto CAVs:



Failure: No available labels

Failure: No available labels
Repair: multimodal concept encoders

Describe (in text) the concepts that are useful for a task

Use a multimodal concept encoder (CLIP) to annotate large dataset

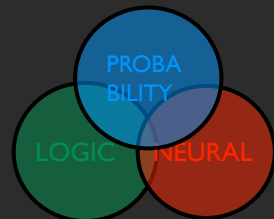
Failure: Few available labels
Repair: Constrain the concepts logically

Train concept encoders using a signal coming from a **semantic based regularization**:

- unsupervised loss functions encoding logic

$\text{Ski} \rightarrow \text{Snow} \vee \text{Mountain} \vee \text{Helmet}$

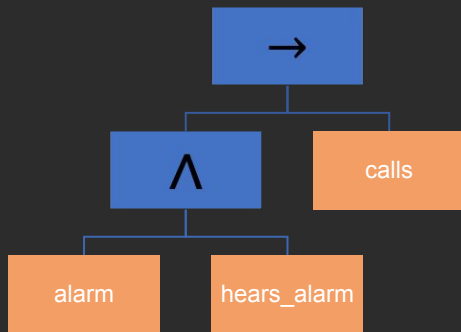
$\text{Dog} \wedge \text{Person} \rightarrow \text{Leash} \vee \text{Sofa} \vee \text{Ball}$



[1] Diligenti, Michelangelo, Marco Gori, and Claudio Sacca. "Semantic-based regularization for learning and inference." Artificial Intelligence 244 (2017): 143-165.

[2] Xu, Jingyi, et al. "A semantic loss function for deep learning with symbolic knowledge." International conference on machine learning. PMLR, 2018.

Constrain the concepts logically

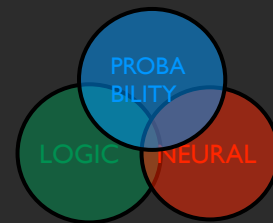
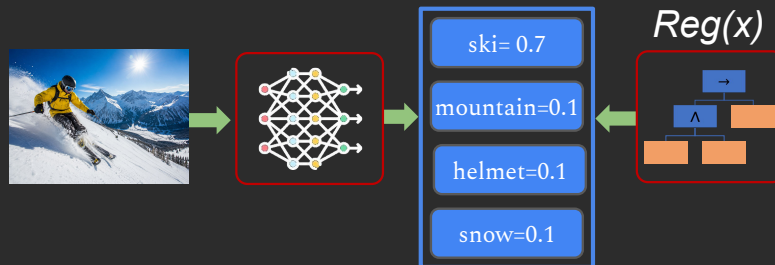


	Product	Łukasiewicz	Gödel
$x \wedge y$	$x \cdot y$	$\max(0, x + y - 1)$	$\min(x, y)$
$x \vee y$	$x + y - x \cdot y$	$\min(1, x + y)$	$\max(x, y)$
$\neg x$	$1 - x$	$1 - x$	$1 - x$
$x \Rightarrow y$ ($x > y$)	y/x	$\min(1, 1 - x + y)$	y

values are in $[0,1]$

$\text{Ski} \rightarrow \text{Snow} \vee \text{Mountain} \vee \text{Helmet}$

$\text{Reg}(x) = \text{fuzzy_impl}(f_ski(x), \text{fuzzy_or}(f_snow(x), f_mountain(x), f_helmet(x)))$

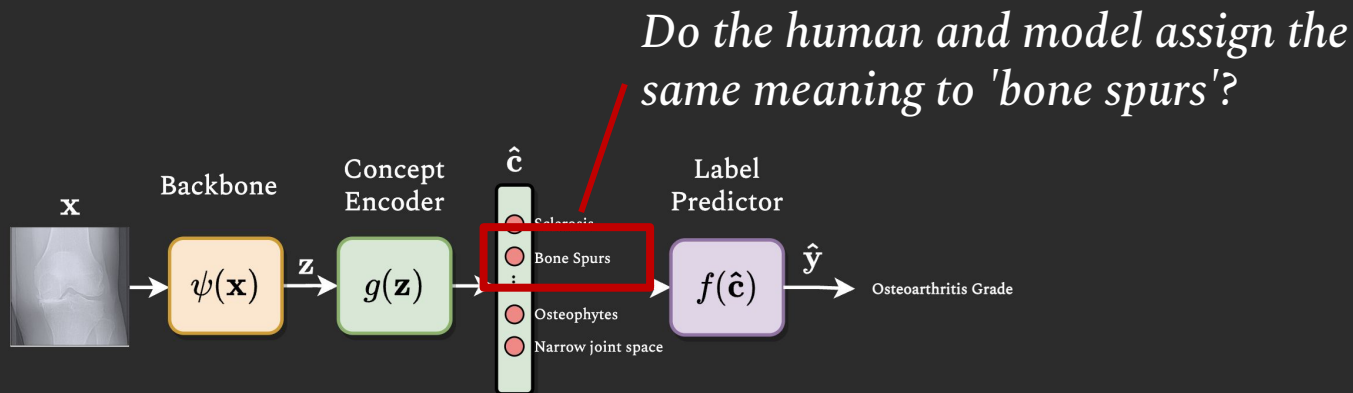


[1] Diligenti, Michelangelo, Marco Gori, and Claudio Sacca. "Semantic-based regularization for learning and inference." *Artificial Intelligence* 244 (2017): 143-165.

[2] Xu, Jingyi, et al. "A semantic loss function for deep learning with symbolic knowledge." *International conference on machine learning*. PMLR, 2018.

Alignment

The interpretability of CBMs rests on a **crucial assumption**:
That the learned concepts are *aligned* to the human



In CBMs: alignment through **concept supervision**... *but is it enough?*

Semantic unclarity leads to leakage

Leakage: the concept representation carries information beyond the intended human concept.

- Humans give a certain meaning
- The machine gives (possibly) that meaning + extra info
 - not a clean bottleneck
 - hidden information

Types of leakage

tail = 0.7

At least 4 types of leakage:

- Task leakage
- Input (distribution) leakage
- Concept leakage (entanglement)
- OOD leakage

Types of leakage

tail = 0.7

At least 4 types of leakage:

- Task leakage
- Input (distribution) leakage
- Concept leakage (entanglement)
- OOD leakage

tail = 0.7 vs tail = 0.8

tail = 0.7 means:

- tail is True
 - whiskers is True
- > cat is True

tail = 0.8 means:

- tail is True
 - whiskers is False
- > dog is True

Types of leakage

tail = 0.7

tail = 0.7 vs tail = 0.8

At least 4 types of leakage:

- Task leakage
- Input (distribution) leakage
- Concept leakage (entanglement)
- OOD leakage

Causes:

- incomplete concept set
 - thus continuous degree as extra capacity
- inexpressive task predictor
 - task needs a “non-linear” feature of concepts
 - use of continuous degree as the non linear feature

Types of leakage

tail = 0.7

At least 4 types of leakage:

- Task leakage
- Input (distribution) leakage
- Concept leakage (entanglement)
- OOD leakage

tail = 0.7

- Even when trained separately (no pressure from the task)
- concepts can encode other unrelated concepts that may be useful for an unknown task
 - (aka sequential training)

Types of leakage

tail = 0.7

At least 4 types of leakage:

- Task leakage
- Input (distribution) leakage
- Concept leakage (entanglement)
- OOD leakage

tail = 0.7

- Concepts can contain information of other (known) concepts
 - intervening on one is not enough

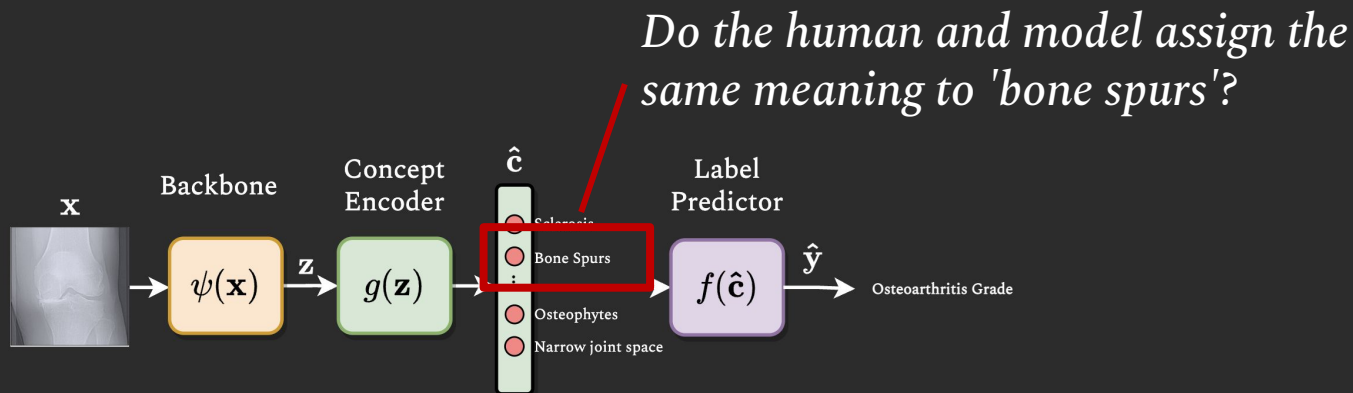
Types of leakage

tail = 0.7

At least 4 types of leakage:

- Task leakage
 - Input (distribution) leakage
 - Concept leakage (entanglement)
 - OOD leakage
- Even if no leakage in-distribution
 - OOD inputs can predicts misaligned concept scores

Can we do something about alignment?



Prototype-based concept prediction



Concept
Alignment
Table

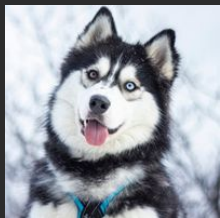
	$p(X_i S_i)$	$p(C_{i,almond\ eyes} S_i)$	$p(C_{i,triangular\ ear} S_i)$
$S_i = 1$		0.9	0.2
$S_i = 2$		0.0	0.8
$S_i = 3$		0.0	0.0

Prototype-based concept prediction



Concept
Alignment
Table

	$p(X_i S_i)$	$p(C_{i,almond\ eyes} S_i)$	$p(C_{i,triangular\ ear} S_i)$
$S_i = 1$		0.9	0.2
$S_i = 2$		0.0	0.8
$S_i = 3$		0.0	0.0





Prototype-based concept prediction

Concept
Alignment
Table

		$p(X_i S_i)$	$p(C_{i,almond\ eyes} S_i)$	$p(C_{i,triangular\ ear} S_i)$
$S_i = 1$		0.9	0.2	
$S_i = 2$		0.0	0.8	
$S_i = 3$		0.0	0.0	



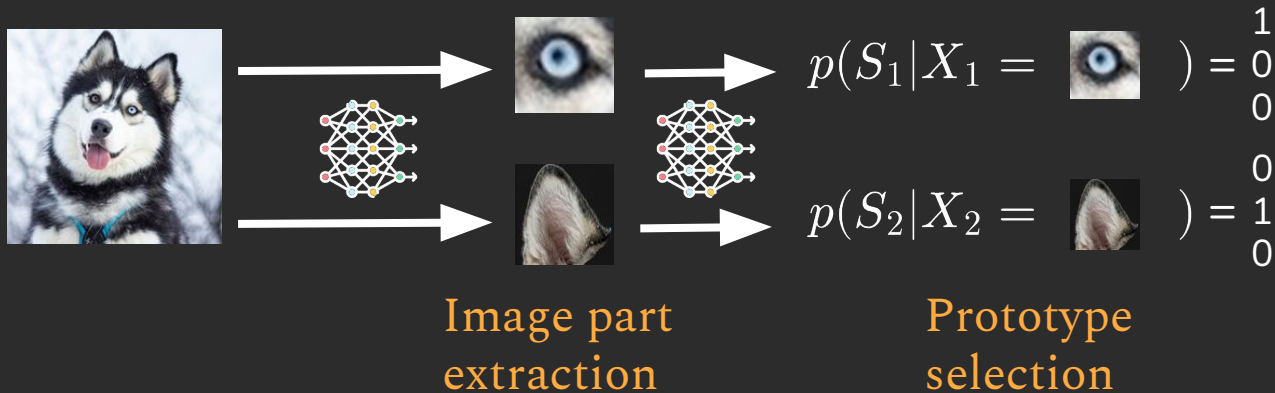
Image part
extraction

Prototype-based concept prediction



Concept
Alignment
Table

	$p(X_i S_i)$	$p(C_{i,almond\ eyes} S_i)$	$p(C_{i,triangular\ ear} S_i)$
$S_i = 1$ 		0.9	0.2
$S_i = 2$ 		0.0	0.8
$S_i = 3$ 		0.0	0.0

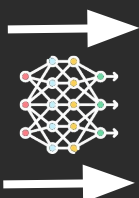
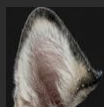
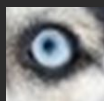
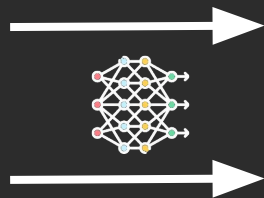
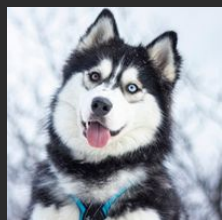




Prototype-based concept prediction

Concept
Alignment
Table

	$p(X_i S_i)$	$p(C_{i,almond\ eyes} S_i)$	$p(C_{i,triangular\ ear} S_i)$
$S_i = 1$ 		0.9	0.2
$S_i = 2$ 		0.0	0.8
$S_i = 3$ 		0.0	0.0



$$\begin{aligned} p(S_1|X_1 = \text{eye}) &= \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \\ p(S_2|X_2 = \text{ear}) &= \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \end{aligned}$$

Image part
extraction

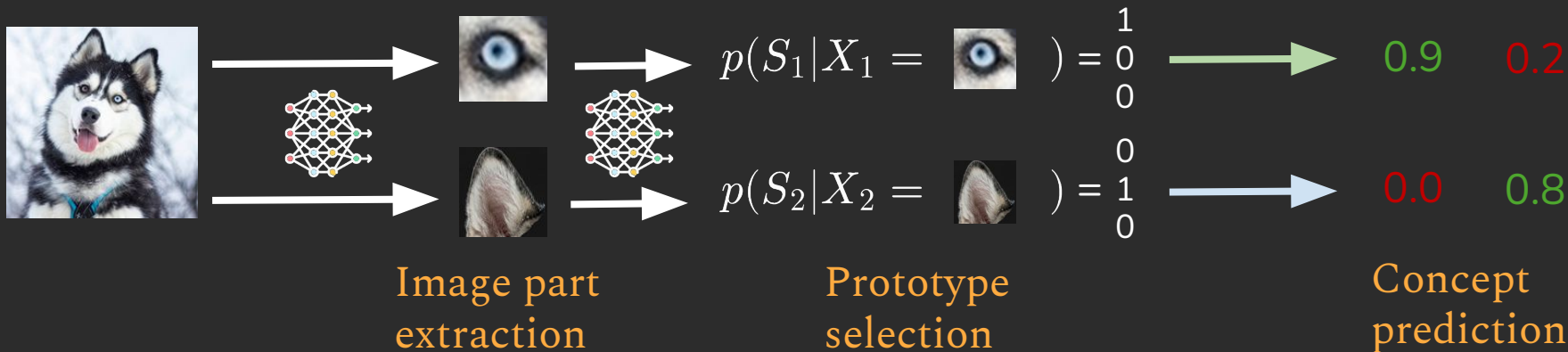
Prototype
selection

Prototype-based concept prediction



Concept
Alignment
Table

	$p(X_i S_i)$	$p(C_{i,almond\ eyes} S_i)$	$p(C_{i,triangular\ ear} S_i)$
$S_i = 1$ 		0.9	0.2
$S_i = 2$ 		0.0	0.8
$S_i = 3$ 		0.0	0.0



Verification and semantics



Each row in the table can be
verified by the human

Image	Active Concepts
	Bangs, Blond Hair, Wavy Hair Heavy Makeup, No Beard, Pale Skin
	Digit is 4
	Small, Cyan, Sphere, Metal Small, Yellow, Cube, Rubber

Verification and semantics



Each row in the table can be **verified** by the human

Image	Active Concepts
	Bangs, Blond Hair, Wavy Hair
	Heavy Makeup, No Beard, Pale Skin
	Digit is 4
	Small, Cyan, Sphere, Metal
	Small, Yellow, Cube, Rubber

Each concept's **semantics** is given by the prototypes for which it is active

Concept	Semantics
Brown Hair	 ∨ 
Pointy Nose	 ∨ 
Digit is 3	 ∨ 
Blue	 ∨  ∨ ...
Cube	 ∨  ∨ ...

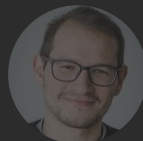
Tutorial Outline

I. Interpretability theory



~50 mins

II. Concept representations



~40 mins

III. Task predictions

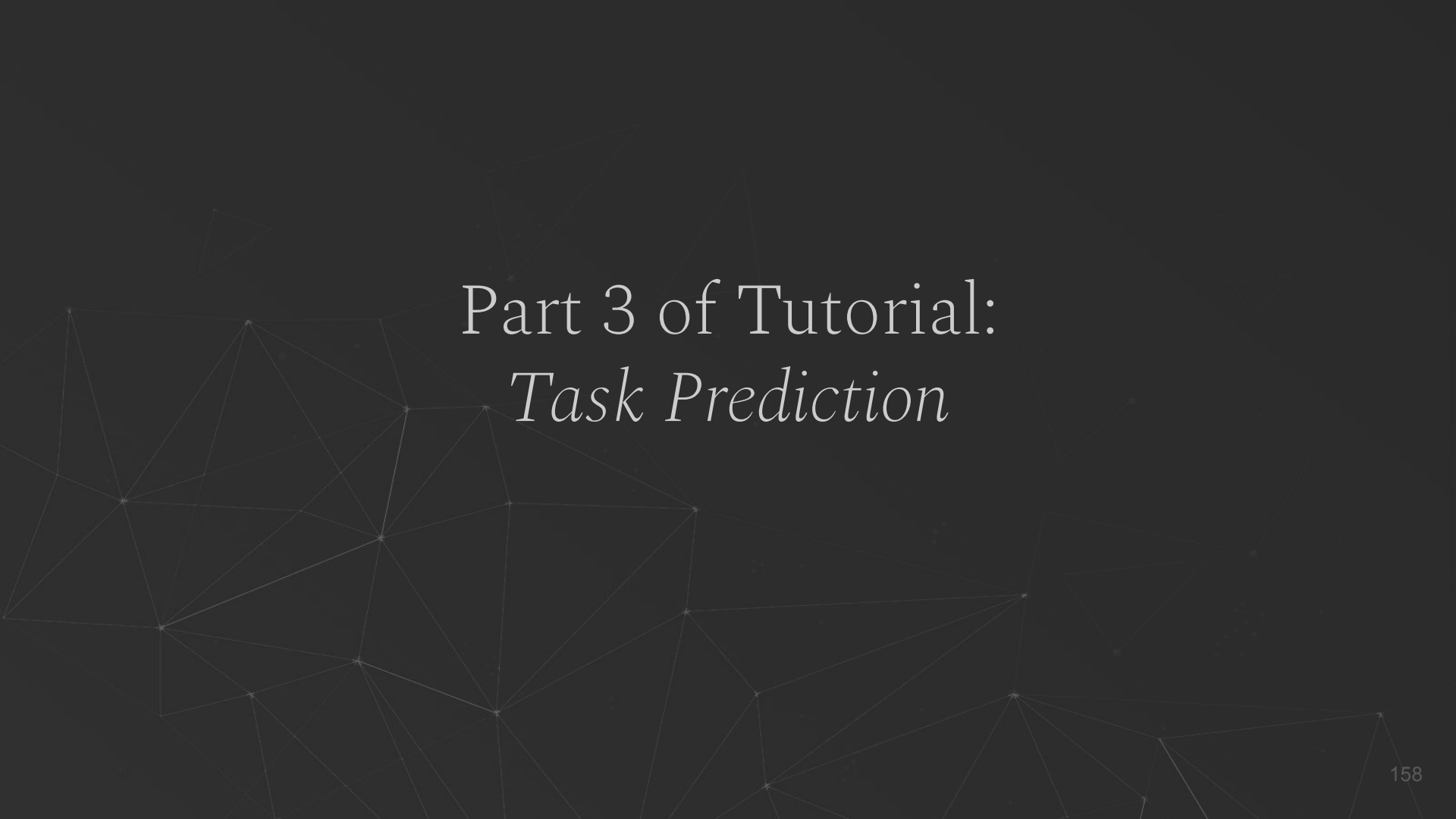


~50 mins

IV. PyTorch Concepts

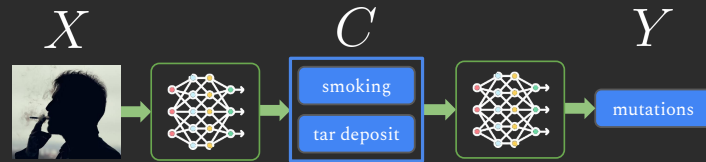


~20 mins



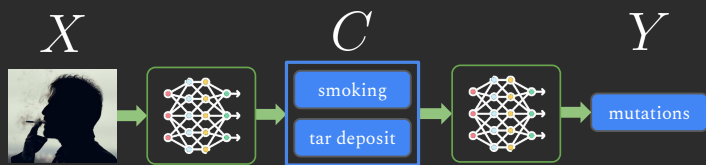
Part 3 of Tutorial: *Task Prediction*

Recap: The concept bottleneck model



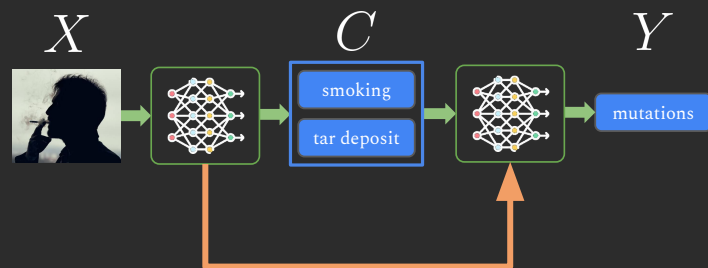
Recap: The concept bottleneck model

Ideal model



concepts = human-aligned intermediate
representation

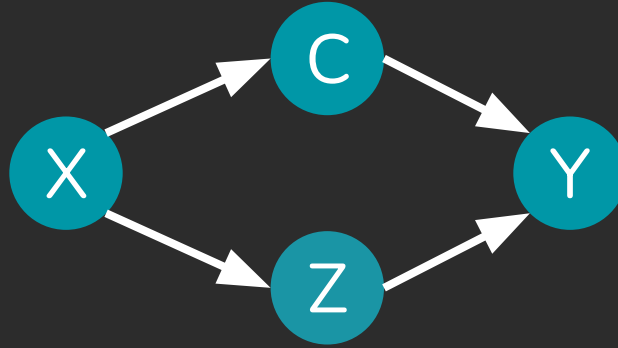
Recap: The concept bottleneck model



residual path = extra non-aligned
information

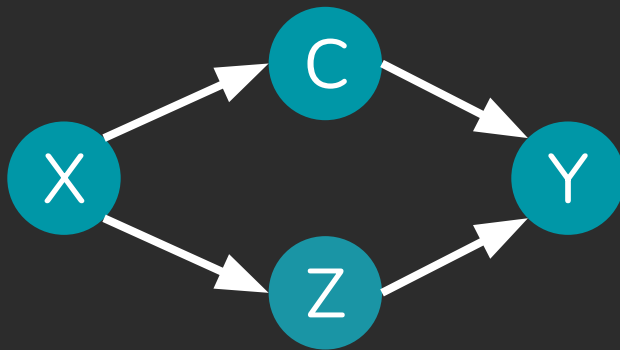
=> more accurate models

Probabilistic Graphical Model



How should we **predict** task labels $p(Y \mid C, Z)$?

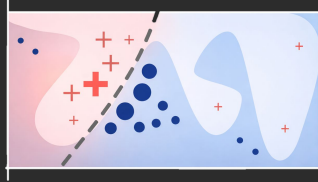
Instantiating the task predictor



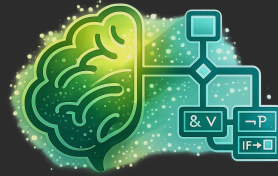
What is $p(Z \mid X)$?

What is $p(Y \mid C, Z)$?

Task predictors



Locally-linear
Predictors



Neurosymbolic
Predictors



Causal
Predictors

The concept-bottleneck model

So far, we have assumed that the task predictor $p(Y|C)$ is a logistic regressor



$$p(C = \mathbf{1} | X = \mathbf{x}) = f_{\phi}(\mathbf{x}) \quad p(Y = 1 | C = \mathbf{c}; \boldsymbol{\theta}, b) = \sigma(\boldsymbol{\theta}^{\top} \mathbf{c} + b)$$

The concept-bottleneck model

So far, we have assumed that the task predictor $p(Y|C)$ is a logistic regressor

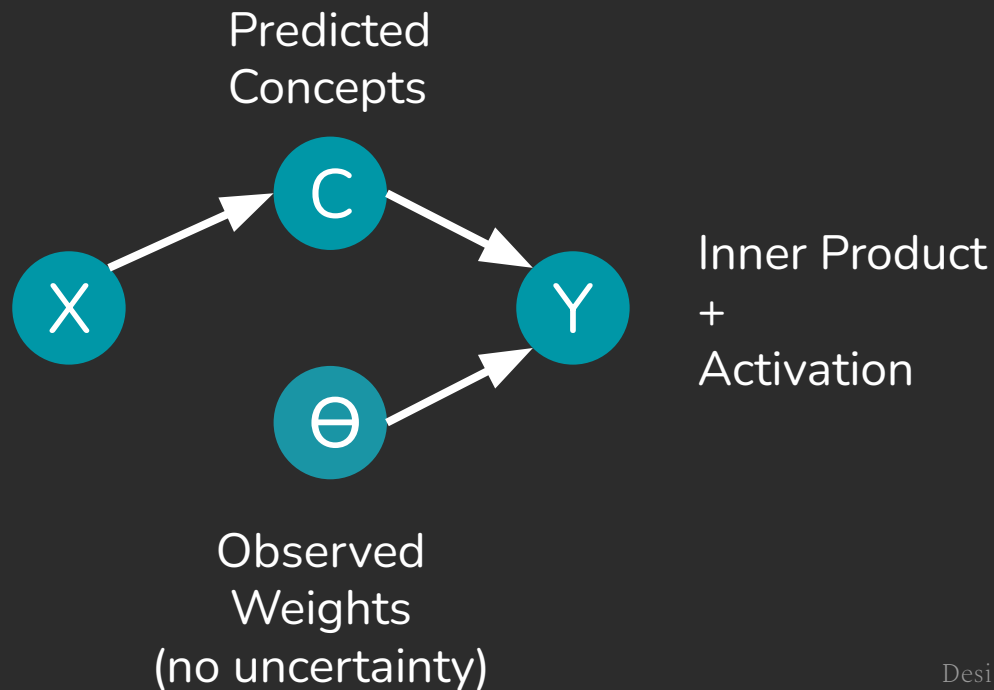


$$p(C = \mathbf{1} | X = \mathbf{x}) = f_{\phi}(\mathbf{x}) \quad p(Y = 1 | C = \mathbf{c}; \boldsymbol{\theta}, b) = \sigma(\boldsymbol{\theta}^{\top} \mathbf{c} + b)$$

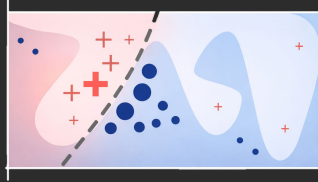
In this representation, the parameters $\boldsymbol{\theta}$ and b are implicit in $p(Y|C)$.

Reify the logistic regressor

$$p(Y = 1|C = \mathbf{c}; \boldsymbol{\theta}, b) = \sigma(\boldsymbol{\theta}^\top \mathbf{c} + b)$$



Task predictors



Locally-linear
Predictors

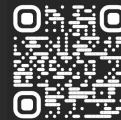


Neurosymbolic
Predictors

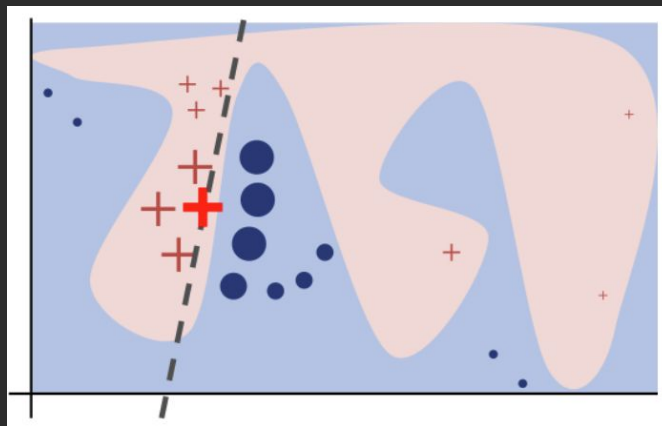


Causal
Predictors

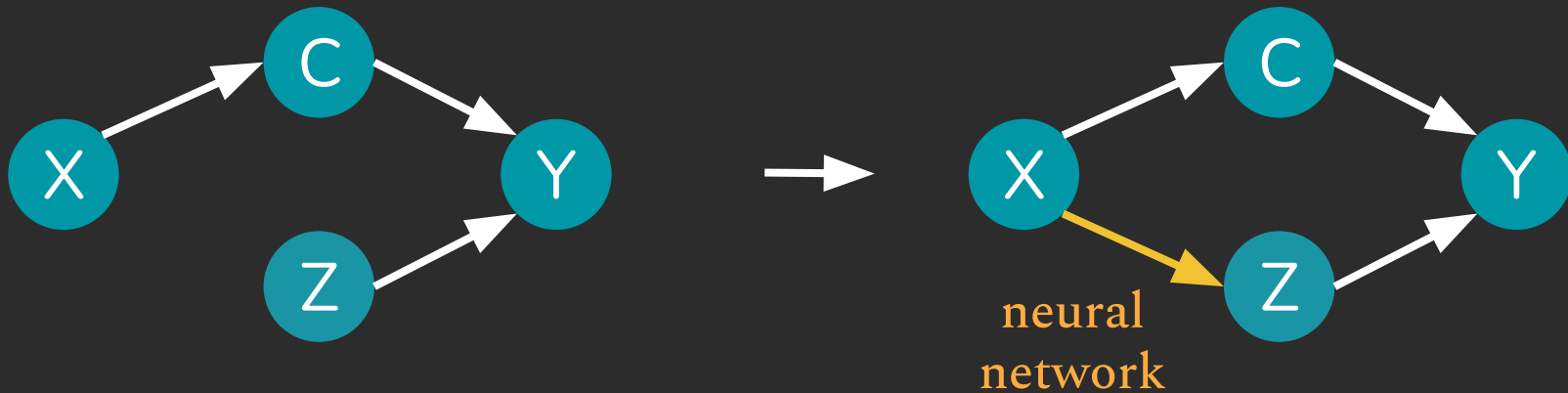
Locally-interpretable task predictors



As a first step, we can construct a non-linear task predictor $f(c) = p(Y|C=c)$ that behaves "linear-like" in the local neighborhood of c



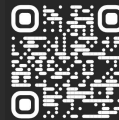
Locally-interpretable task predictors



We keep $P(Y \mid C, Z)$ a linear projection + activation

But we make weights Z a function of the input

Locally-interpretable task predictors



[Local Linearity] The model should

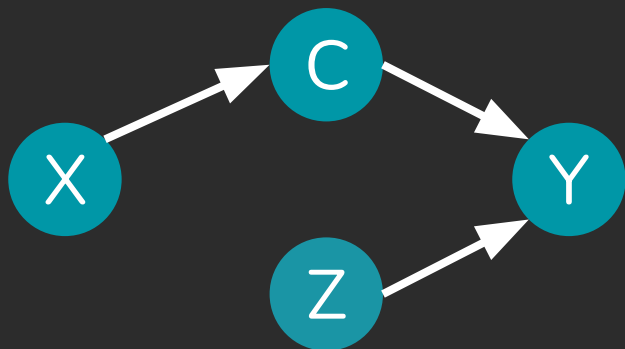
- **behave as a linear classifier**, at least in the neighborhood of a sample's concept space,
- **be locally stable**, small changes in Z should not change the prediction drastically,

This can be enforced by minimizing a loss that implicitly leads to this condition holding:

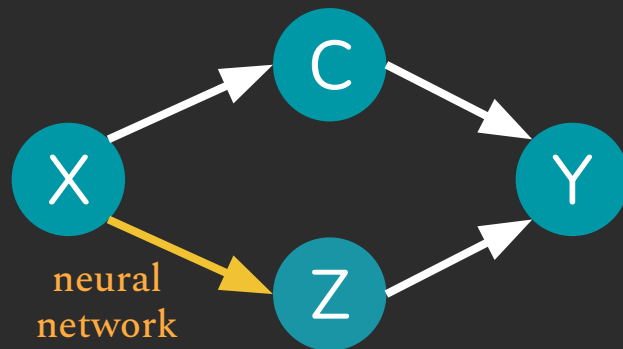
$$L(x, y) = \|\nabla_x \text{logit}(P(Y = y \mid C(x), Z(x))) - Z(x)^\top J_x C(x)\|_2^2$$

Problems with locally-linear predictors

Unfortunately, locally-linear predictors cannot provide insights about the **global behavior** of the model

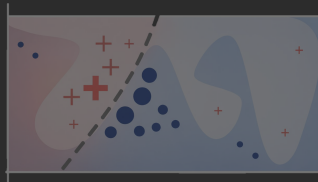


globally interpretable

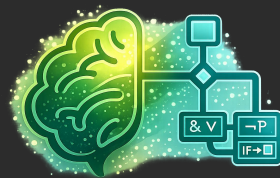


locally interpretable

Task Predictors



Locally-linear
Predictors



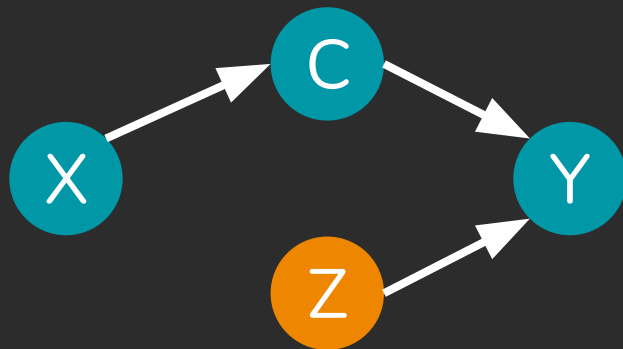
Neurosymbolic
Predictors



Causal
Predictors

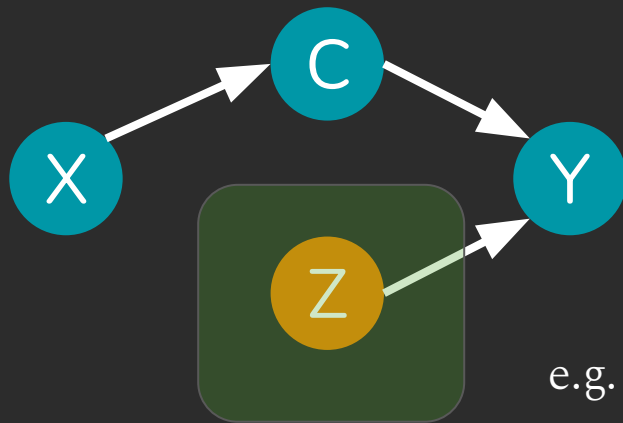
What can we use other than linear layers?

Can we use **logic rules** over the concepts to design more expressive but still interpretable task predictors?



What can we use other than linear layers?

Can we use **logic rules** over the concepts to design more expressive but still interpretable task predictors?

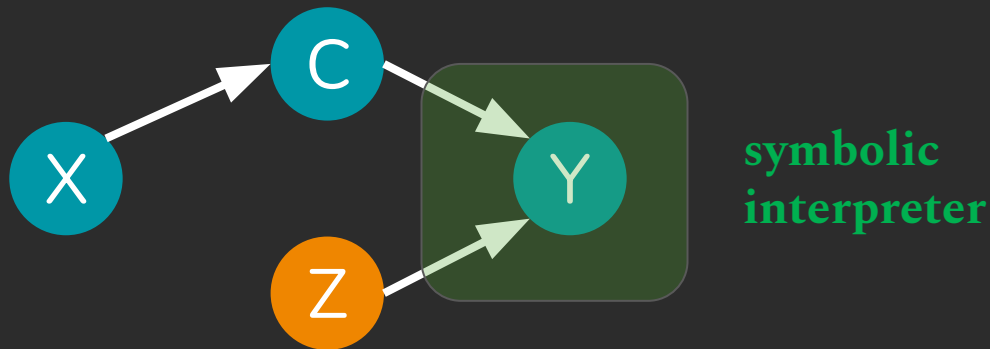


e.g. if $C_1 \wedge \neg C_2$ then Y

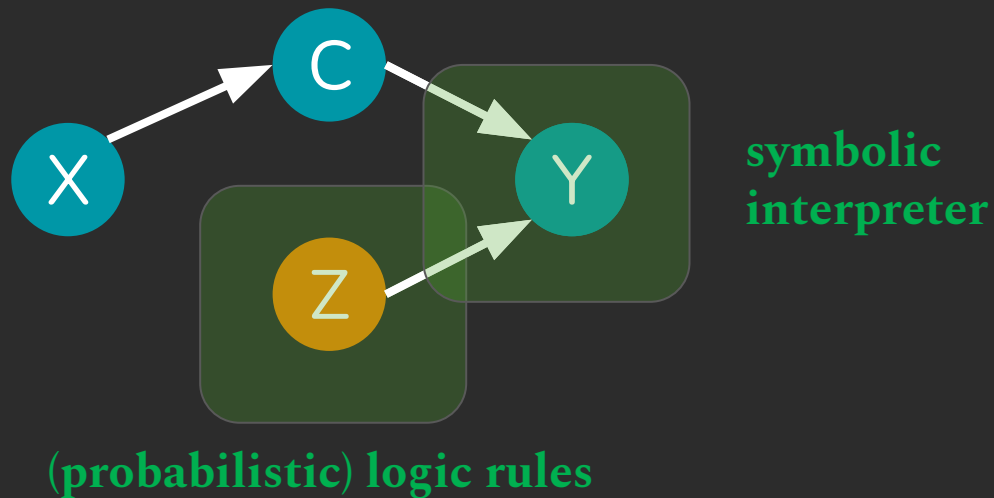
(probabilistic) logic rules

What can we use other than linear layers?

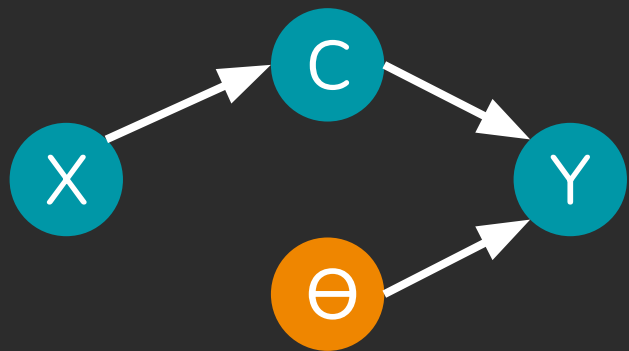
Can we use **logic rules** over the concepts to design more expressive but still interpretable task predictors?



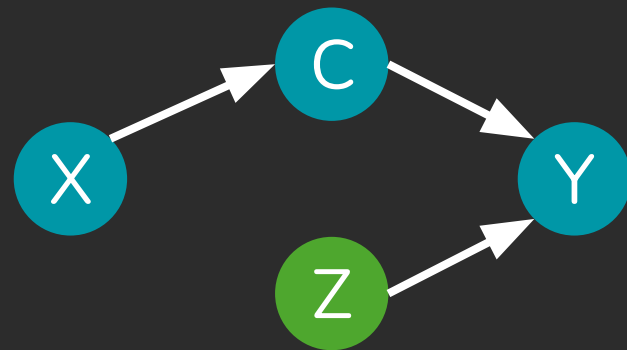
How do we learn the logic rules?



Start with a (deep) CBM, then extract rules post-hoc



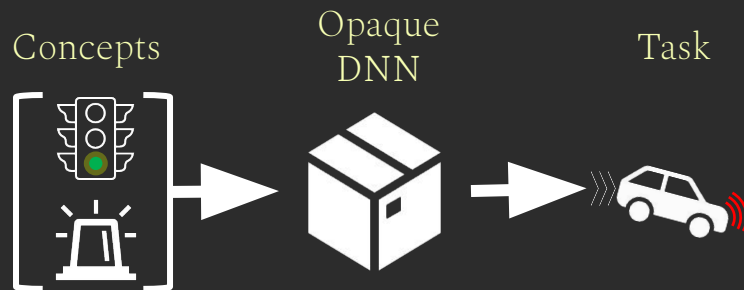
weights of a neural network



logic rules

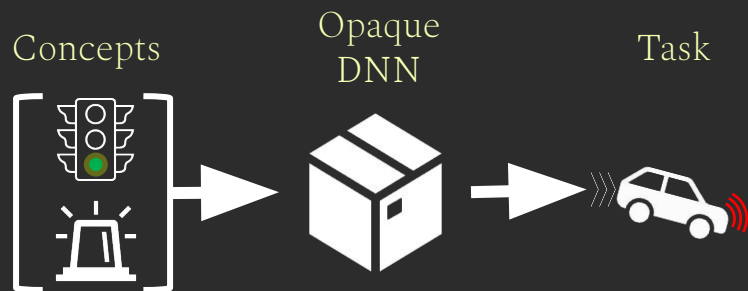
Start with a (deep) CBM, then extract rules post-hoc

We learn a DNN predicting task labels from the concepts



Start with a (deep) CBM, then extract rules post-hoc

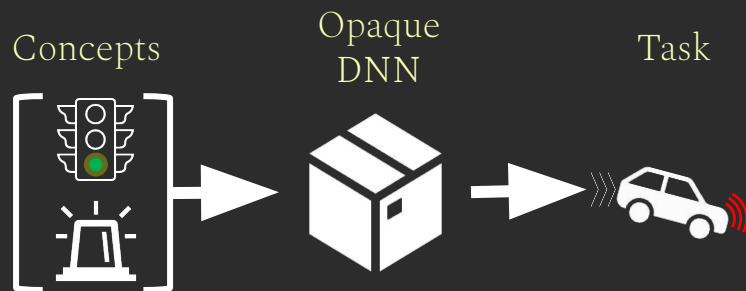
We learn a DNN predicting task labels from the concepts and predict the task from every possible input combination



		
0	0	1

Start with a (deep) CBM, then extract rules post-hoc

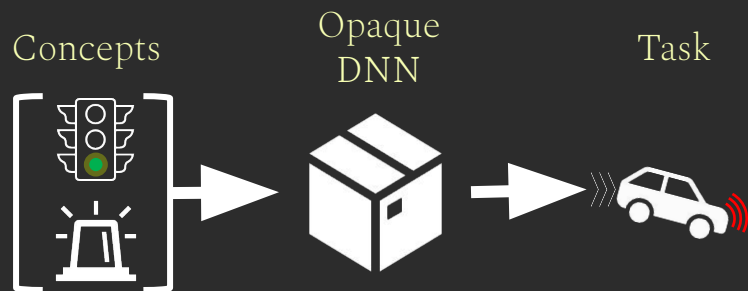
We learn a DNN predicting task labels from the concepts and predict the task from every possible input combination



		
0	0	1
1	0	0

Start with a (deep) CBM, then extract rules post-hoc

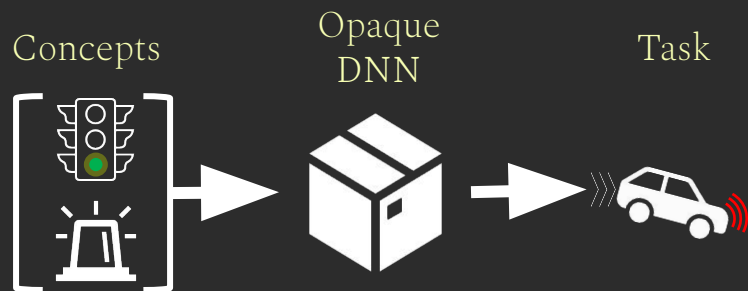
We learn a DNN predicting task labels from the concepts and predict the task from every possible input combination



		
0	0	1
1	0	0
0	1	1

Start with a (deep) CBM, then extract rules post-hoc

We learn a DNN predicting task labels from the concepts and predict the task from every possible input combination

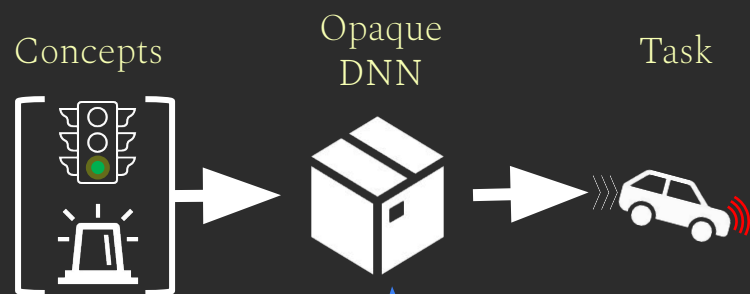


		
0	0	1
1	0	0
0	1	1
1	1	1

Start with a (deep) CBM, then extract rules post-hoc

We learn a DNN predicting task labels from the concepts and predict the task from every possible input combination

This results in a truth table where we can easily extract rules from



		
0	0	1
1	0	0
0	1	1
1	1	1



 = (NOT ) OR 



Start with a (deep) CBM, then extract rules post-hoc

We can use this approach to construct a **set of logic rules** mapping concepts to downstream tasks

This allows us to capture **non-linear relationships** between concepts and tasks using human-understandable logic rules

Start with a (deep) CBM, then extract rules post-hoc

We can use this approach to construct a **set of logic rules** mapping concepts to downstream tasks

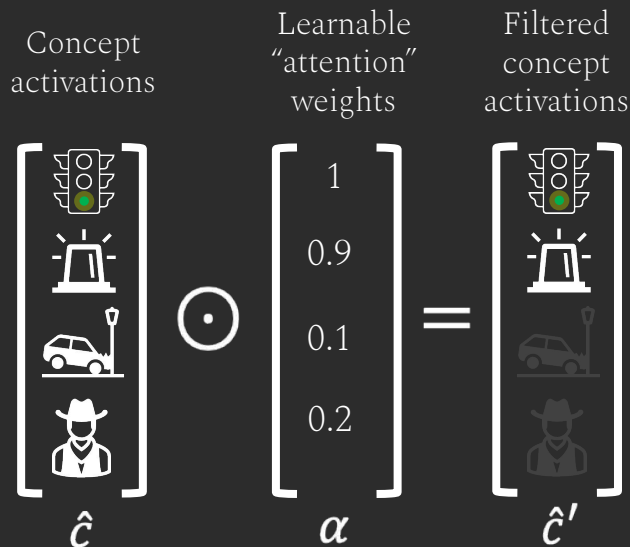
Problem: #forward passes required for this is exponential on the #concepts!



Efficiently constructing logic-based predictors

One approach for efficiently construct a logic-based predictor is:

Step 1: Filter concept activations using learnable *attention weights* α

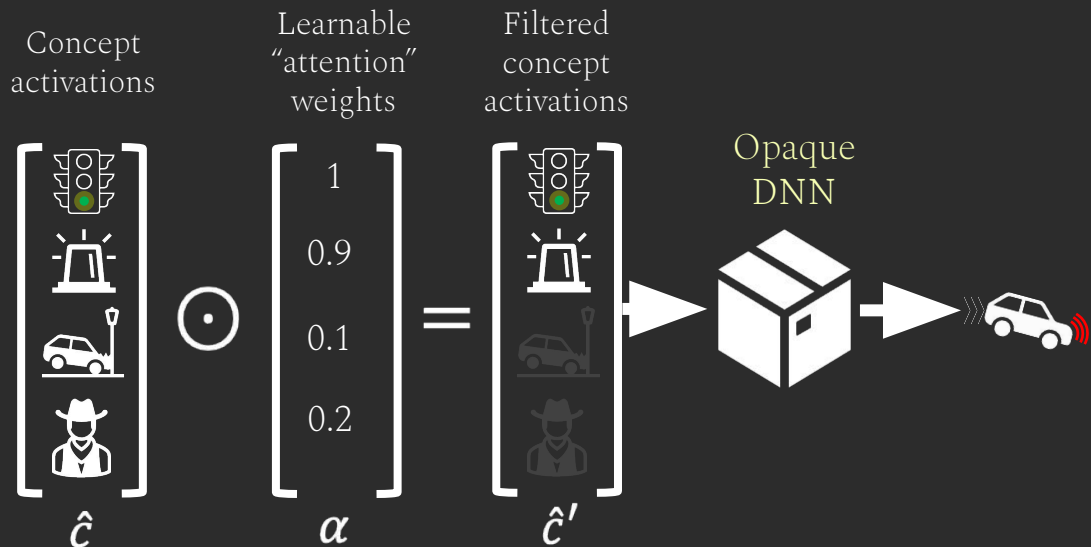




Efficiently constructing logic-based predictors

One approach for efficiently construct a logic-based predictor is:

Step 2: Solve task with the (small) set of selected concepts

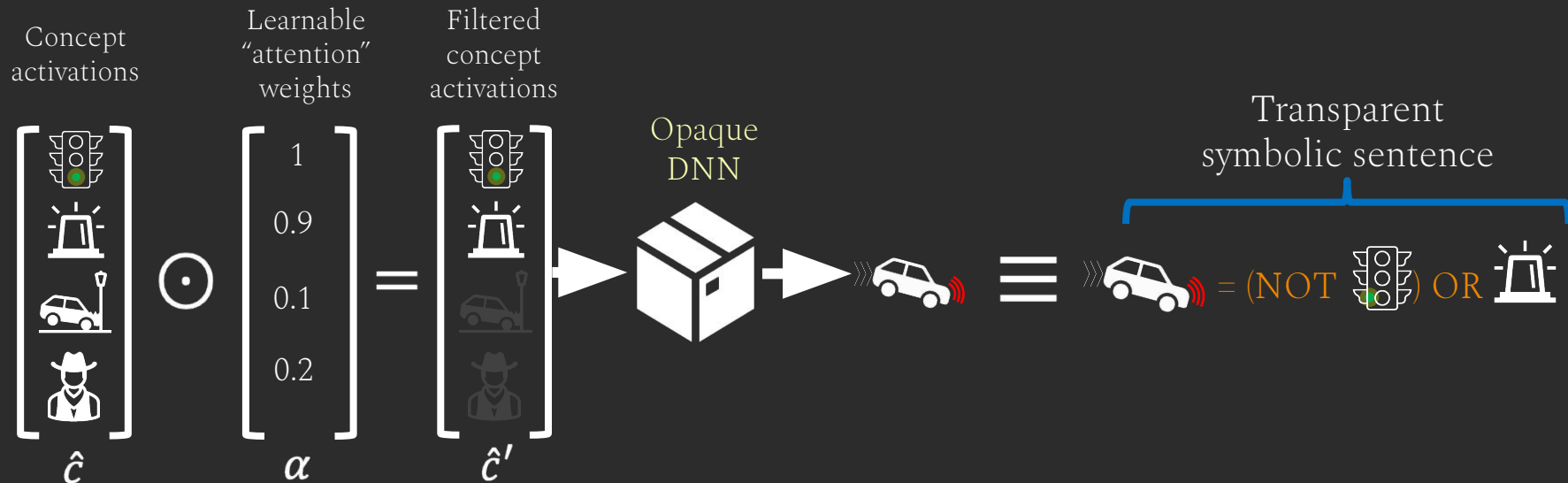




Efficiently constructing logic-based predictors

One approach for efficiently construct a logic-based predictor is:

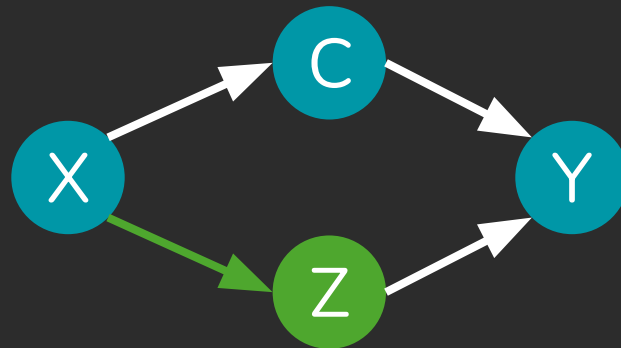
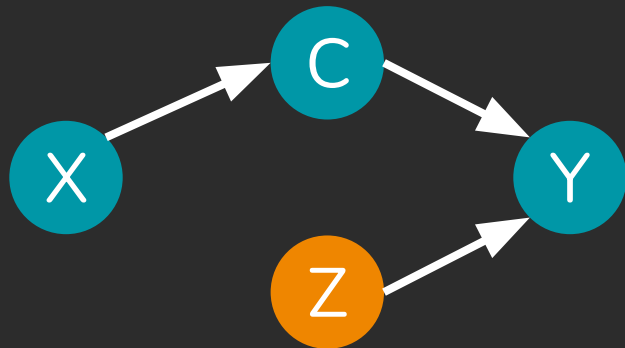
Step 3: Derive rules from the (much smaller) truth table



Neurosymbolic task predictors

So far, the logic rules were a **prior**: they are **the same for any input X**

We can also make the rules a **posterior**, i.e., **depend on the input X**



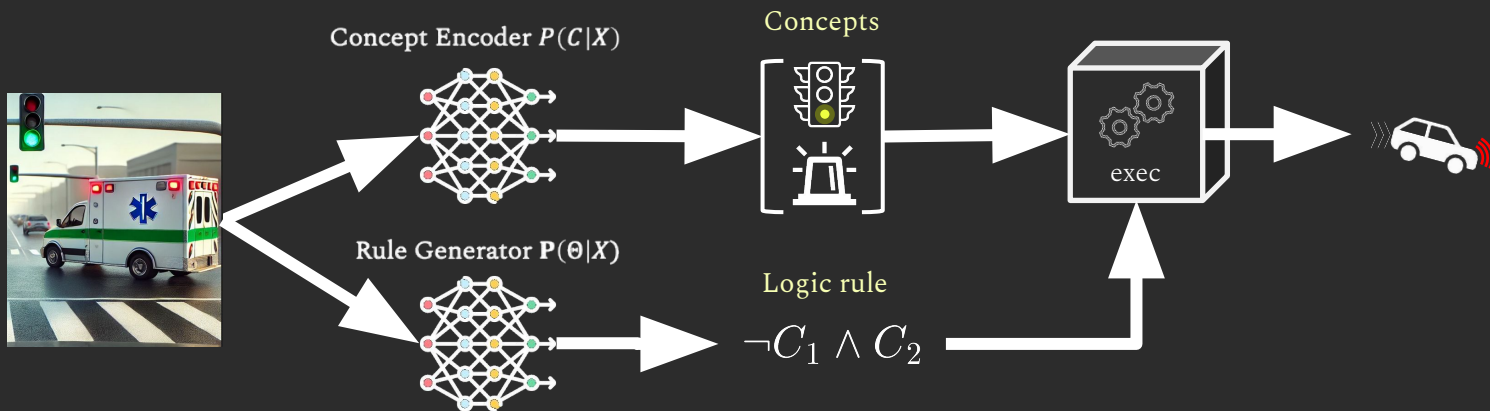
Neurosymbolic task predictors



Generally, such NeSy predictors are constructed as follows:

Step 1: DNNs parametrize both concepts & rule parameters

Step 2: A symbolic engine executes rules using predicted concepts



[1] Barbiero et al. "Interpretable neural-symbolic concept reasoning." International Conference on Machine Learning. PMLR, 2023.

[2] Debot et al. "Interpretable concept-based memory reasoning." NeurIPS 2024.

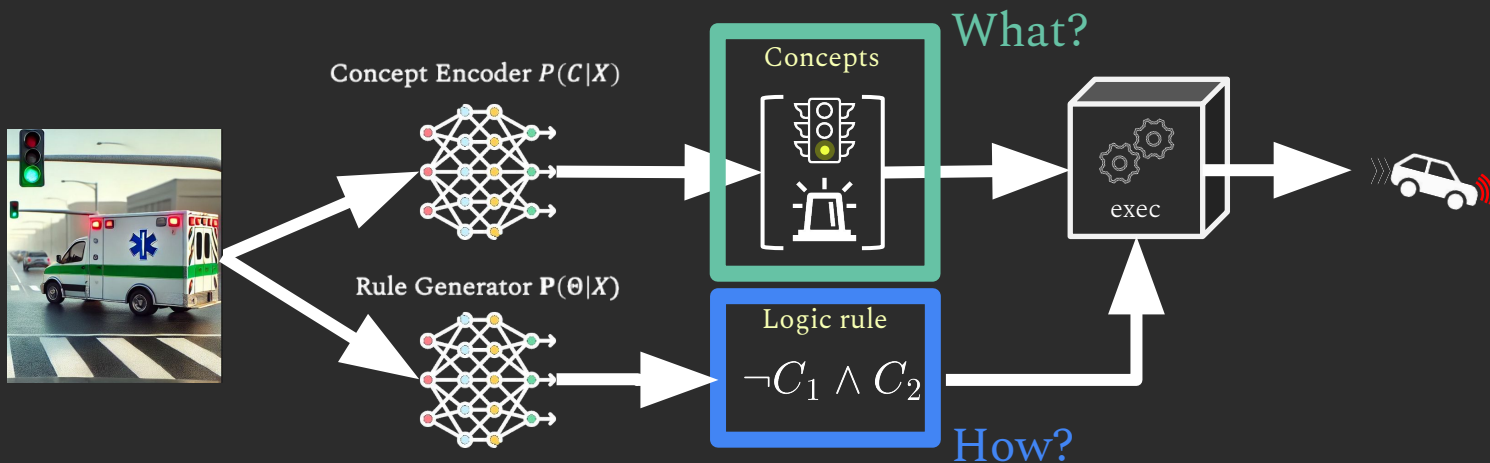
Neurosymbolic task predictors



Generally, such NeSy predictors are constructed as follows:

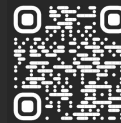
Step 1: DNNs parametrize both concepts & rule parameters

Step 2: A symbolic engine executes rules using predicted concepts



[1] Barbiero et al. "Interpretable neural-symbolic concept reasoning." International Conference on Machine Learning. PMLR, 2023.

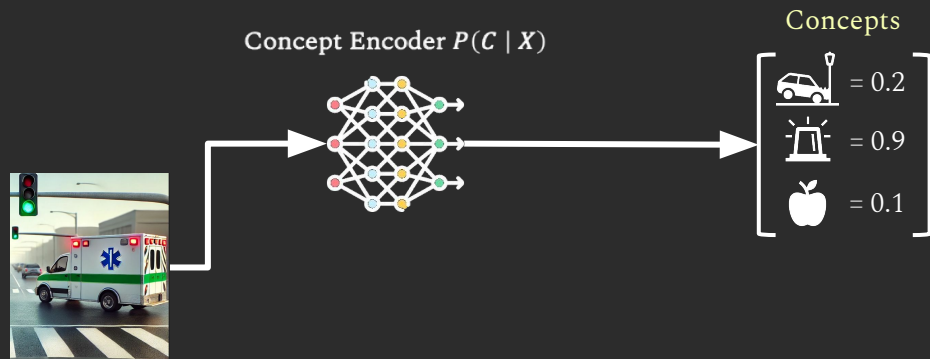
[2] Debot et al. "Interpretable concept-based memory reasoning." NeurIPS 2024.



Memory-based NeSy predictors

We can build a **Concept-based Memory Reasoner** (CMR) by:

Step 1: Predict concept probabilities $P(C|X)$ using a DNN

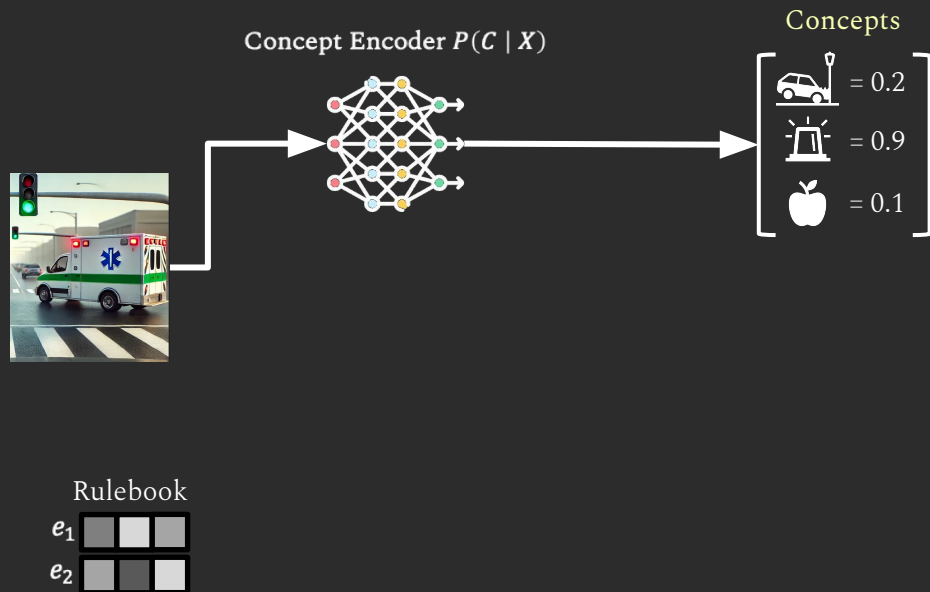




Memory-based NeSy predictors

We can build a **Concept-based Memory Reasoner** (CMR) by:

Step 2: Select a rule “embedding” from a learnable “rulebook” using a DNN

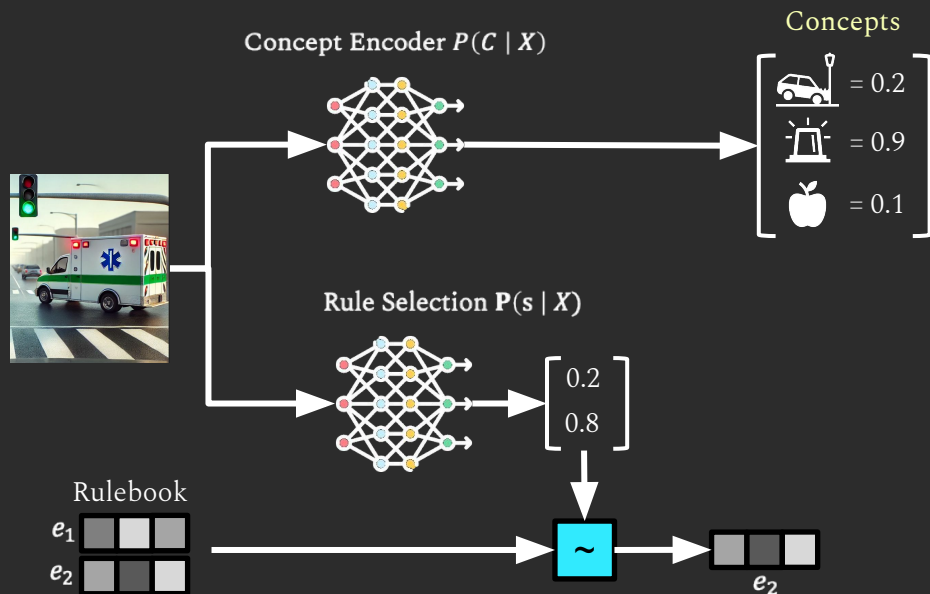




Memory-based NeSy predictors

We can build a **Concept-based Memory Reasoner** (CMR) by:

Step 2: Select a rule “embedding” from a learnable “rulebook” using a DNN

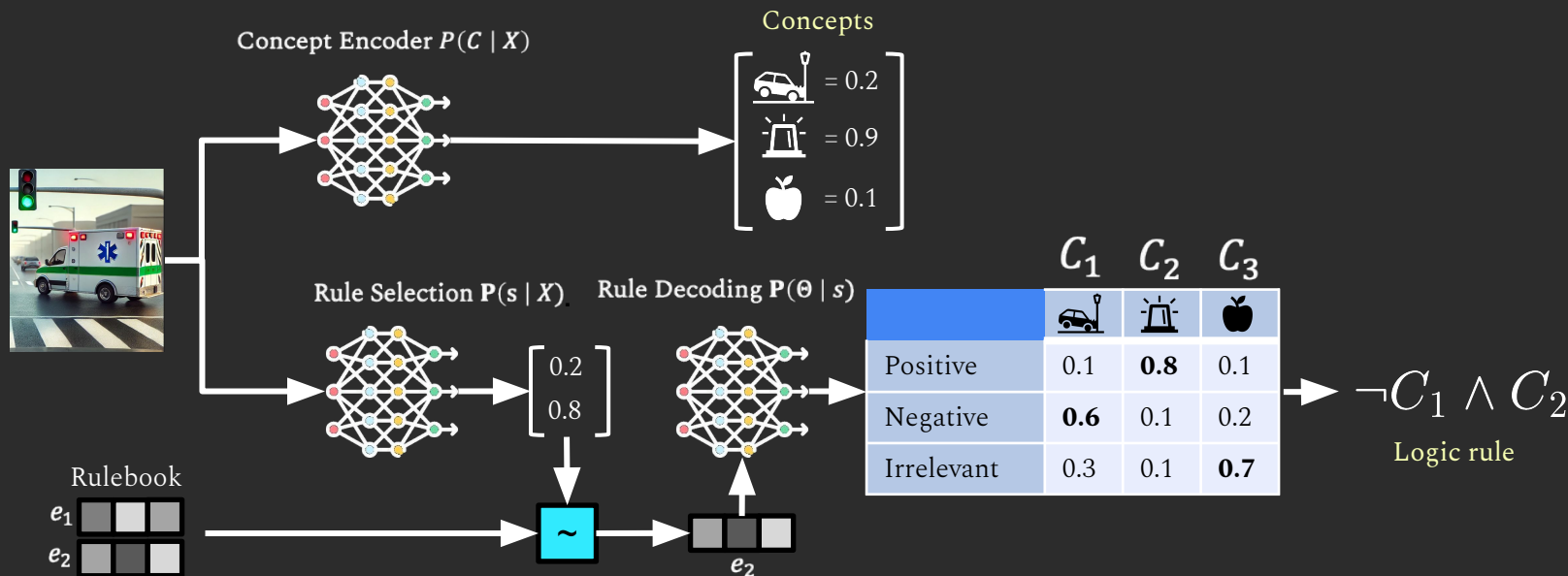




Memory-based NeSy predictors

We can build a **Concept-based Memory Reasoner** (CMR) by:

Step 3: Decode the rule embedding into three states per concept using a DNN (= rule parameters)

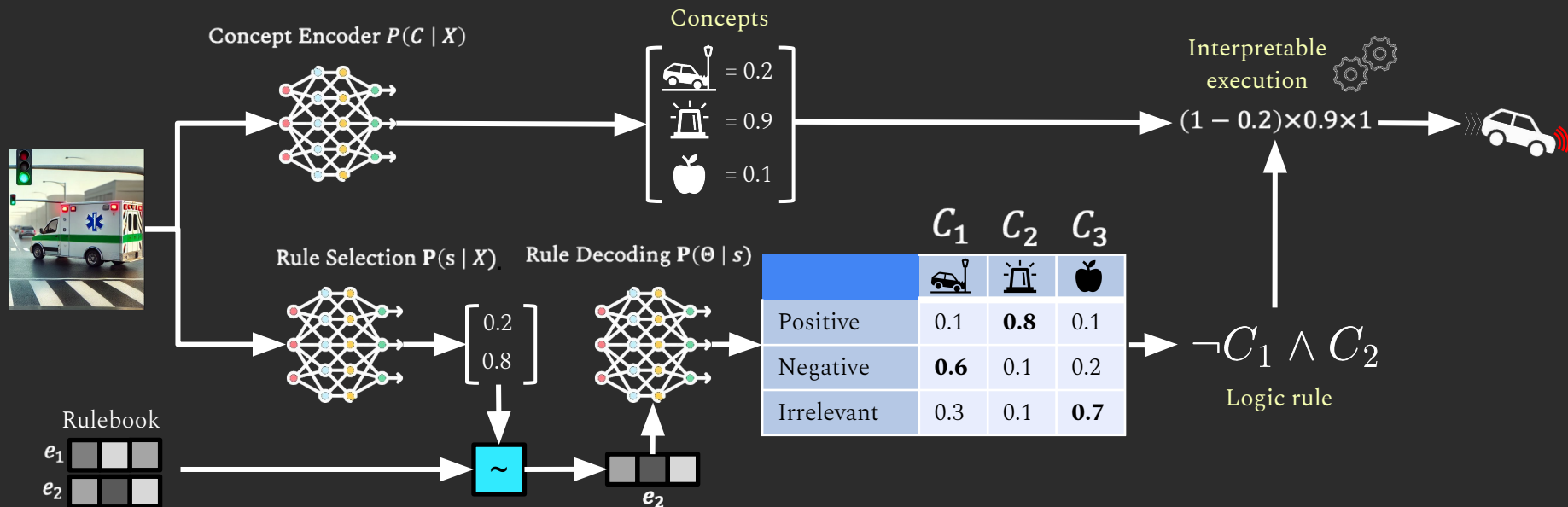


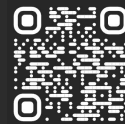


Memory-based NeSy predictors

We can build a **Concept-based Memory Reasoner** (CMR) by:

Step 4: Execute the rule combining concept states and concept activations

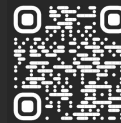




Properties of CMR

These steps enable CMR to:

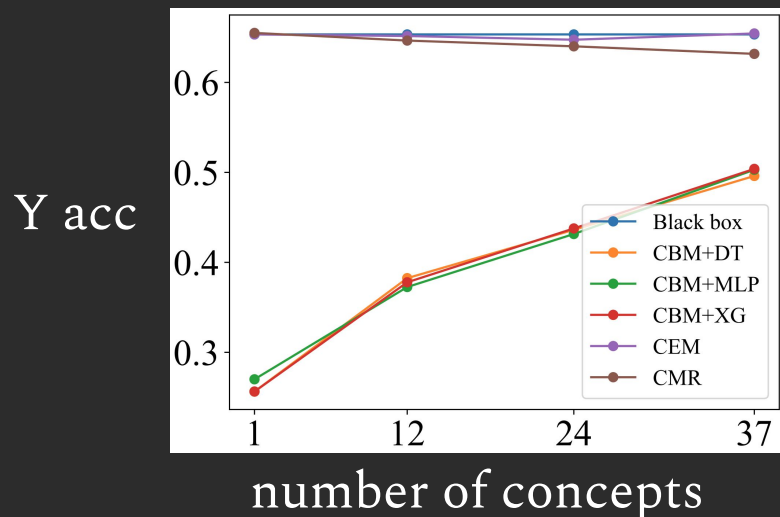
1. Become a **universal approximator** akin to opaque DNNs irrespective of the used concept set



Properties of CMR

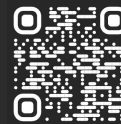
These steps enable CMR to:

1. Become a **universal approximator** akin to opaque DNNs irrespective of the used concept set



models with posterior Z (e.g. CMR)

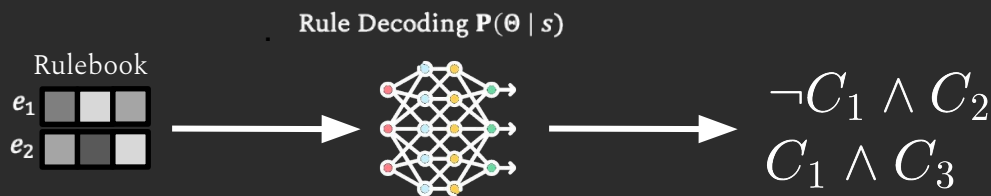
models with prior Z



Properties of CMR

These steps enable CMR to:

1. Become a **universal approximator** akin to opaque DNNs
2. Provide both **local and global interpretability**



Inference mechanisms can only be selected from a finite set of rules!



Properties of CMR

These steps enable CMR to:

1. Become a **universal approximator** akin to opaque DNNs
2. Provide both **local and global interpretability**
3. Allow **formal verification** of desirable properties

Rulebook

$$\begin{array}{l} \neg C_1 \wedge C_2 \\ C_1 \wedge C_3 \end{array}$$

+ property



SAT
Solver



holds!

"Does a property hold no matter which rule is selected?"

Task Predictors



Locally-linear
Predictors



Neurosymbolic
Predictors



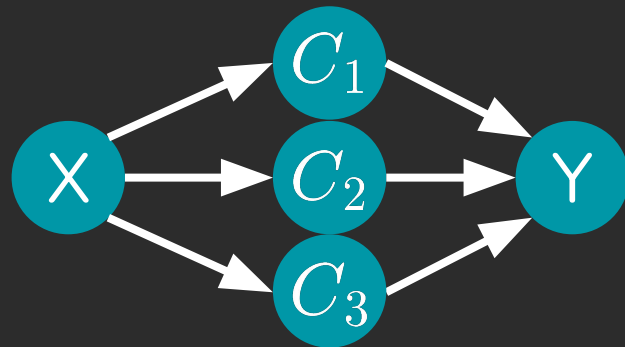
Causal
Predictors

Causal task predictors

Most CBMs make a conditional-independence assumption between concepts:

$$P(C_1, C_2|X) = P(C_1|X) \cdot P(C_2|X)$$

which is **unrealistic**



Causal task predictors

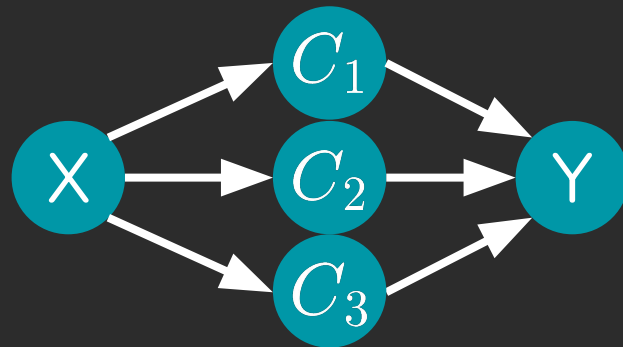
Most CBMs make a conditional-independence assumption between concepts:

$$P(C_1, C_2|X) = P(C_1|X) \cdot P(C_2|X)$$

which is **unrealistic**

Why is this a problem?

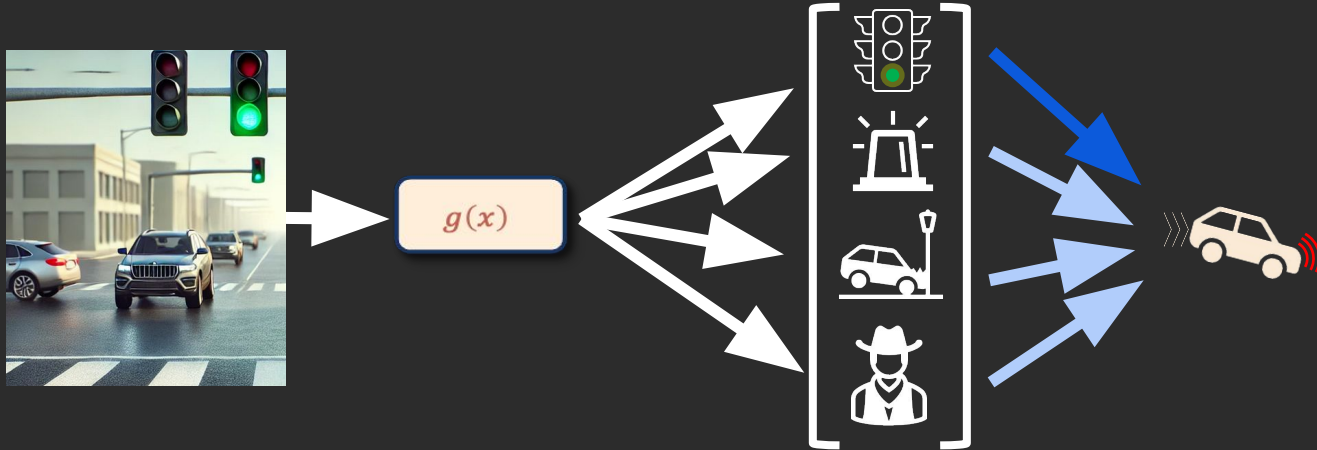
- Incorrect uncertainty quantification
- **Poor intervenability**



Causal task predictors

A **concept intervention** sets a concept prediction to some specific value

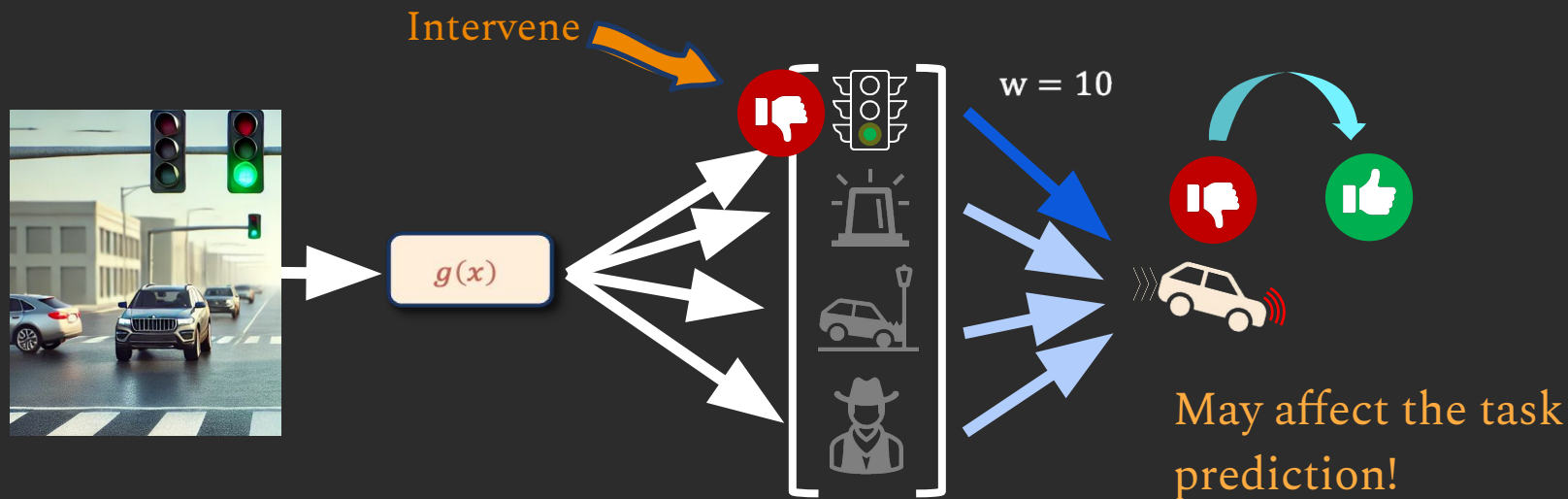
e.g. $P(Y|X, do(C_1 = 1))$



Causal task predictors

A **concept intervention** sets a concept prediction to some specific value

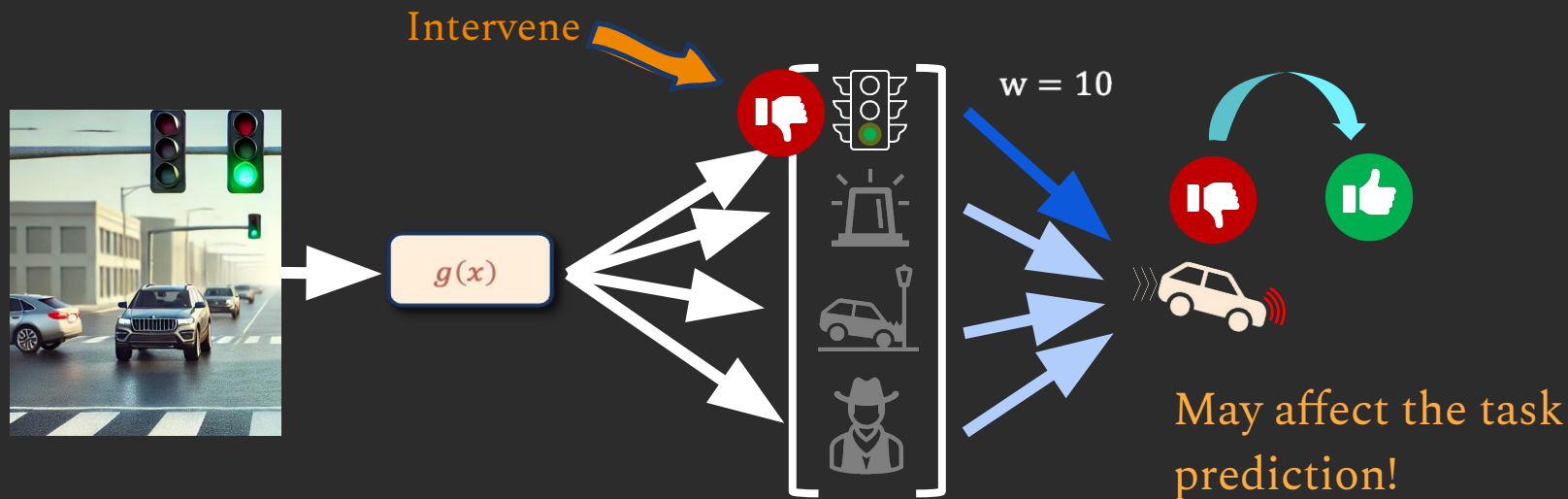
e.g. $P(Y|X, do(C_1 = 1))$



Causal task predictors

A **concept intervention** sets a concept prediction to some specific value

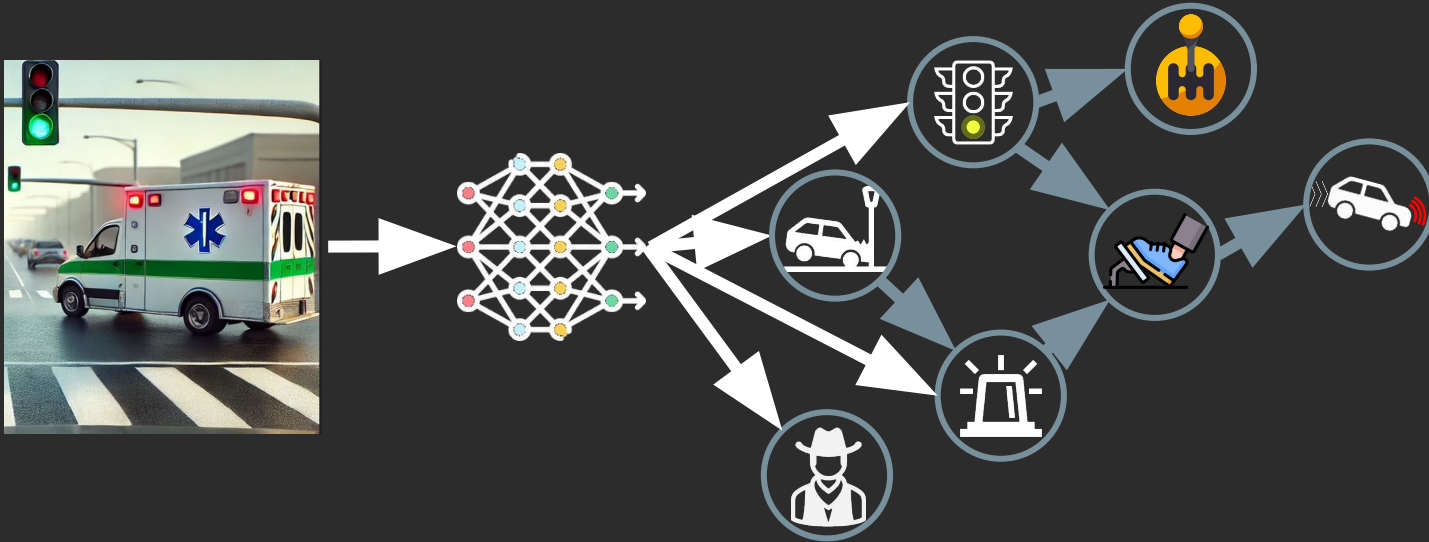
e.g. $P(Y|X, do(C_1 = 1))$



Problem: concepts do not influence each other

Causal task predictors

Causal predictors explain how concepts affect other concepts and task!



Causal opacity

However, for this it is important to understand make a distinction between:

1. Causal reliability
2. Causal opacity

Causal opacity

However, for this it is important to understand make a distinction between:

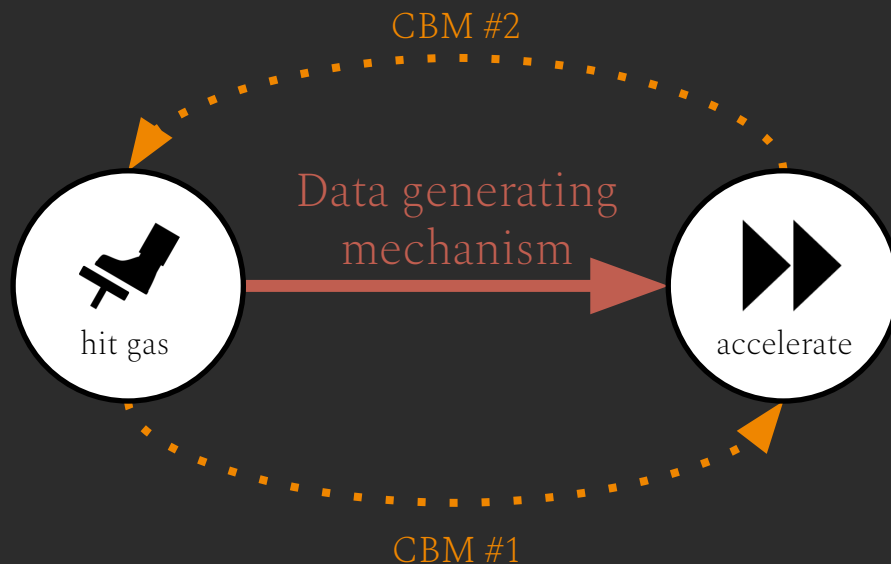
1. Causal reliability: discover causal mechanism of the data generating process



Causal opacity

However, for this it is important to understand make a distinction between:

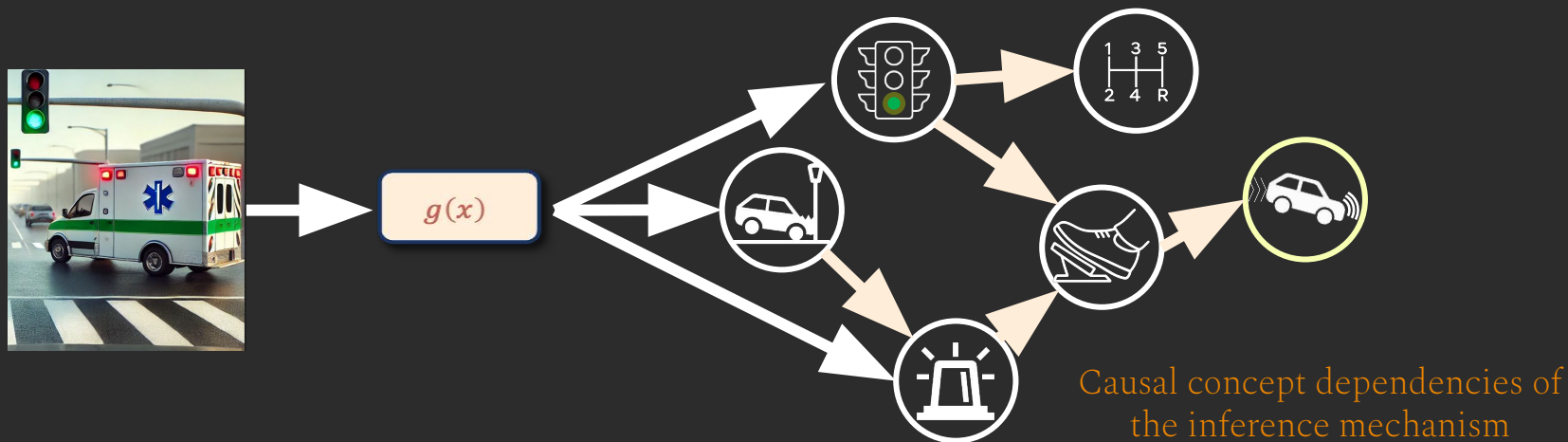
1. Causal reliability: discover causal mechanism of the data generating process
2. Causal opacity: discover causal mechanism of a model's inference process



Concept graph models



We can build a causal label predictor that explicitly uses a concept graph:

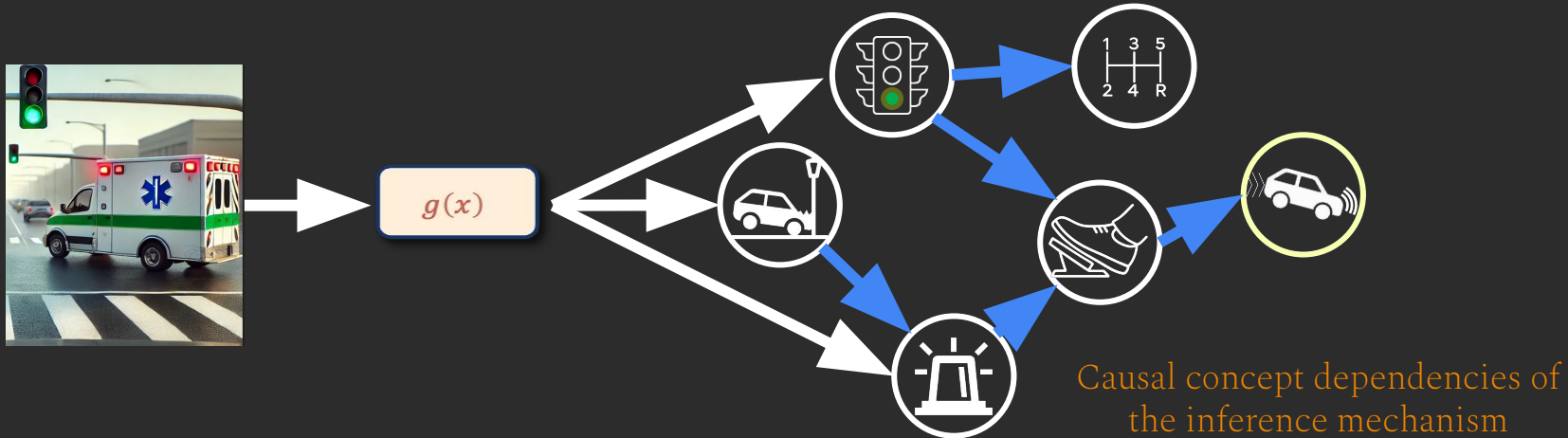


- [1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.
- [2] Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLeaR 2024.
- [3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

Concept graph models



Different approaches use different functions: e.g. **neural networks**

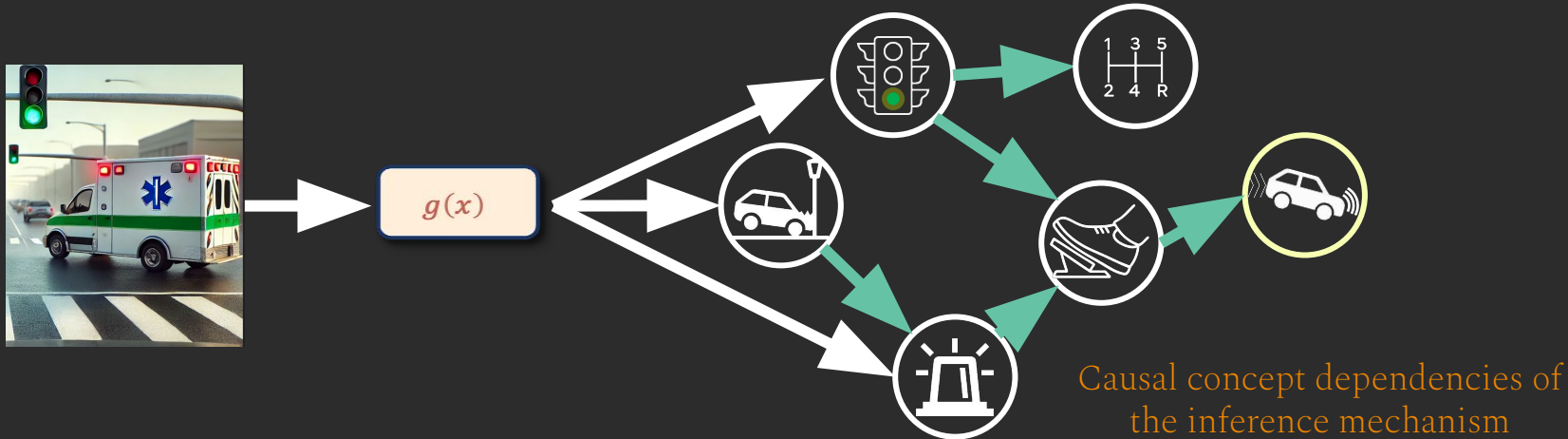


- [1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.
- [2] Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLear 2024.
- [3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

Concept graph models



Different approaches use different functions: e.g. **neural networks + logic rules**



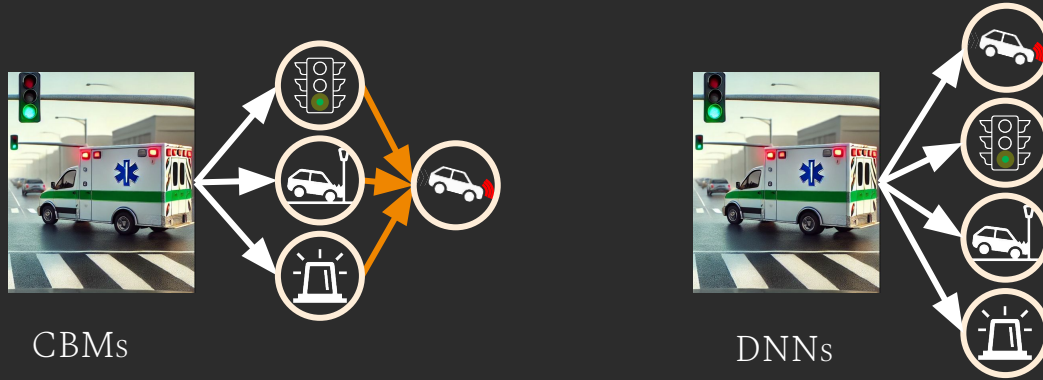
- [1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.
- [2] Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLeaR 2024.
- [3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

Concept graph models



The concept graph itself can be:

1. Given as a *prior*



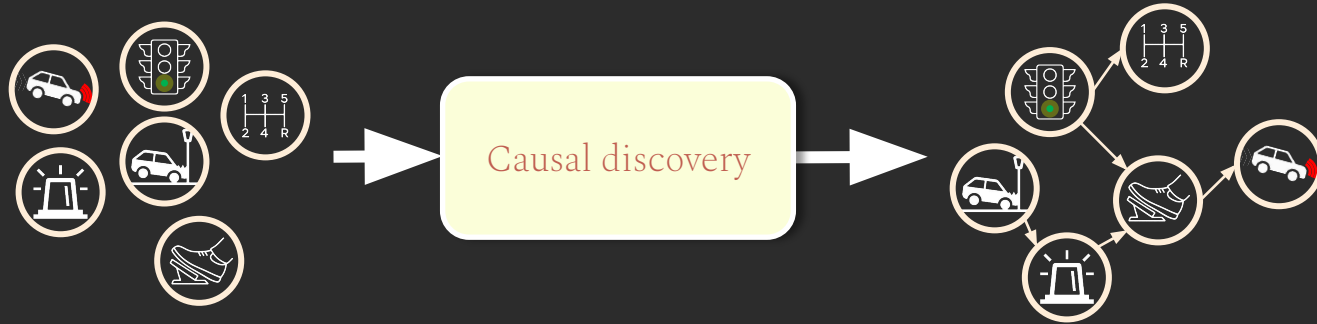
- [1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.
- [2] Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLeaR 2024.
- [3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

Concept graph models



The concept graph itself can be:

1. Given as a *prior*
2. Learnt from data via *causal discovery methods*



[1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.

[2] Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLeaR 2024.

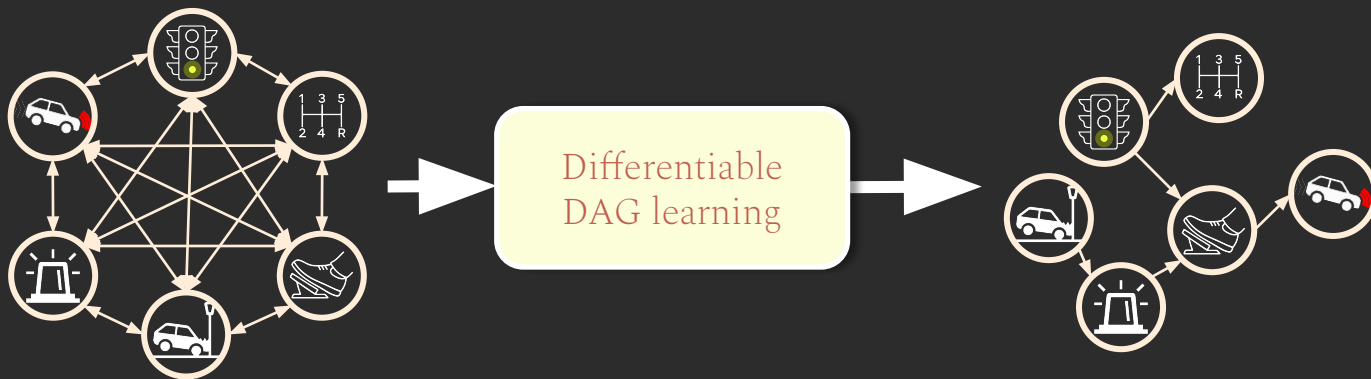
[3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

Concept graph models



The concept graph itself can be:

1. Given as a *prior*
2. Learnt from data via *causal discovery methods*
3. Learnt end-to-end via *(differentiable) DAG learning*

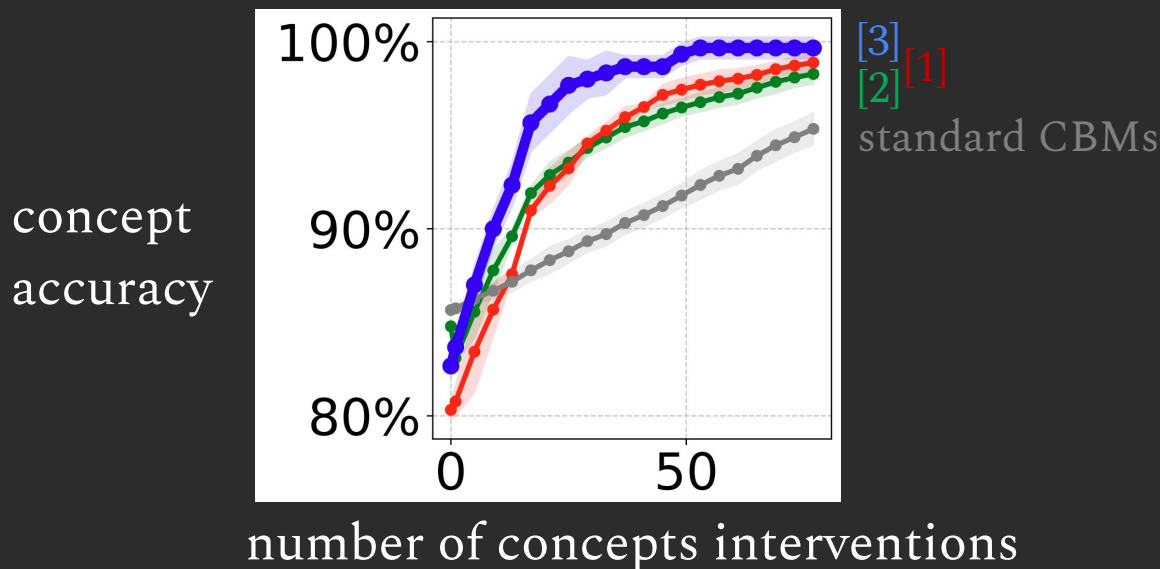


[1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.

[2] Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLeaR 2024.

[3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

Concept graph models and intervenability



[1] Dominici et al. "Causal Concept Graph Models: Beyond Causal Opacity in Deep Learning." ICLR 2025.

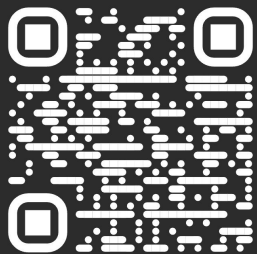
[2] Vandenhirtz et al. "Stochastic Concept Bottleneck Models." NeurIPS 2024.

[3] Debot et al. "Interpretable Hierarchical Concept Reasoning through Attention-Guided Graph Learning." Arxiv 2025.

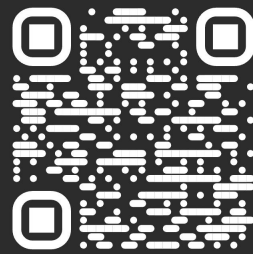
Further causal task predictors

Recent causal task predictors include:

DiConstruct



Causally Interpretable Models



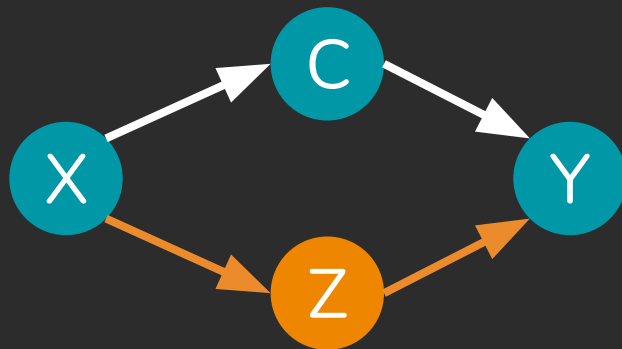
[1] [Moreira et al. "Diconstruct: Causal concept-based explanations through black-box distillation." CLeaR \(2024\).](#)

[2] [Hwang et al. "From black-box to causal-box: Towards building more interpretable models." NeurIPS \(2025\).](#)

Sidechannels

Many models replace the **prior** $p(Z)$ with a **posterior** $p(Z|X)$ (a "sidechannel"), which makes the model significantly more expressive and accurate

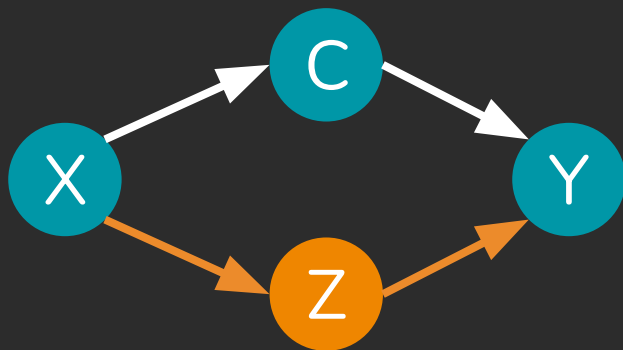
Is there no downside to this?



Influence of sidechannels

Downside: predictions are made *not only* using human-understandable concepts, but also using **uninterpretable latent information**

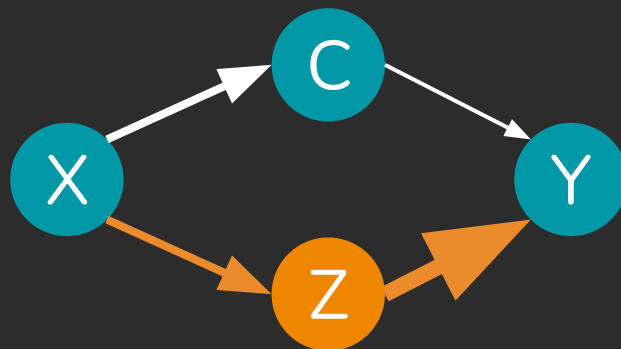
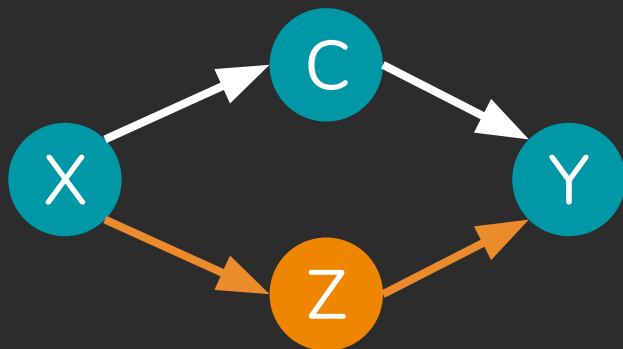
This hurts interpretability!



Influence of sidechannels

Downside: predictions are made *not only* using human-understandable concepts, but also using **uninterpretable latent information**

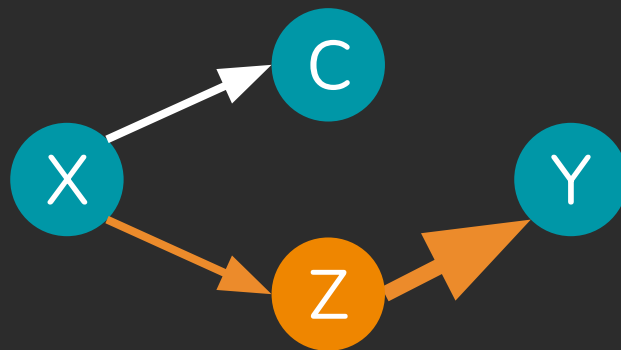
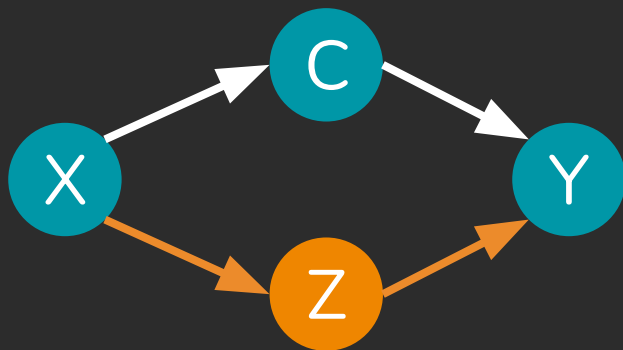
This hurts interpretability!



Influence of sidechannels

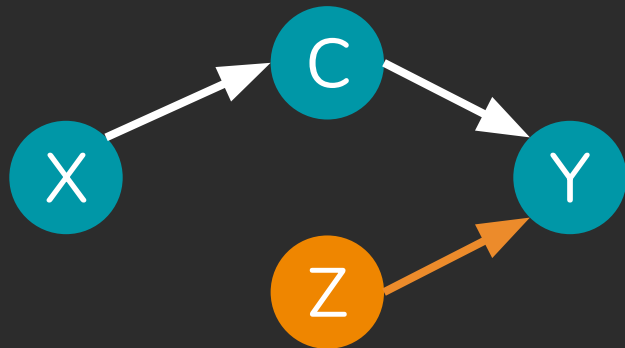
Downside: predictions are made *not only* using human-understandable concepts, but also using **uninterpretable latent information**

This hurts interpretability!

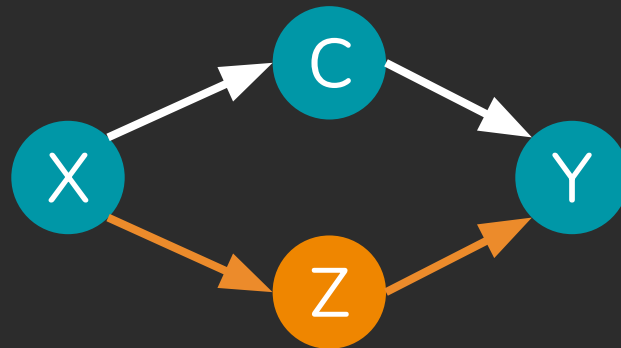


Can we compare the two?

We consider this
fully interpretable



We don't know how
interpretable this is



A metric for sidechannel reliance

The more a model relies on this sidechannel, the less interpretable it is

$$\text{SIS} = \mathbb{E}_{x \sim p(X)} [\mathbb{1}[\textcolor{teal}{y}_x = \textcolor{red}{\bar{y}}_x]]$$

"How often does the sidechannel not influence the model's decision?"

A metric for sidechannel reliance

The more a model relies on this sidechannel, the less interpretable it is

$$\text{SIS} = \mathbb{E}_{x \sim p(X)} [\mathbb{1}[\mathbf{y}_x = \bar{\mathbf{y}}_x]]$$

"How often does the sidechannel not influence the model's decision?"

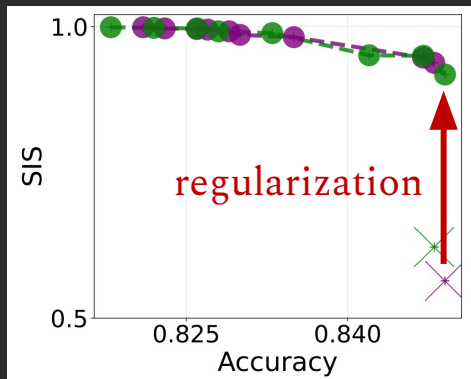
$$\mathbf{p}(\mathbf{y}|\mathbf{x}) = \sum_{c,z} p(c|\mathbf{x}) \cdot \underline{p(z|\mathbf{x})} \cdot p(\mathbf{y}|c, z) \quad \text{standard inference}$$

$$\bar{\mathbf{p}}(\mathbf{y}|\mathbf{x}) = \sum_{c,z} p(c|\mathbf{x}) \cdot \underline{p(z)} \cdot p(\mathbf{y}|c, z) \quad \text{inference with sidechannel prior (learned or marginalized)}$$

Metric can also be **regularized** for during training!

Sidechannels in practice

Expressive $p(Y|C,Z)$
incomplete concepts

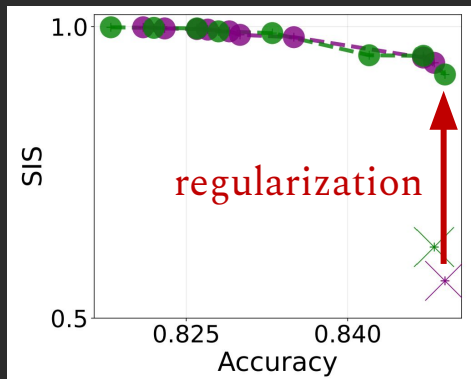


Standard CBM training results
in uninterpretable CBMs

A bit of regularization vastly
improves interpretability

Sidechannels in practice

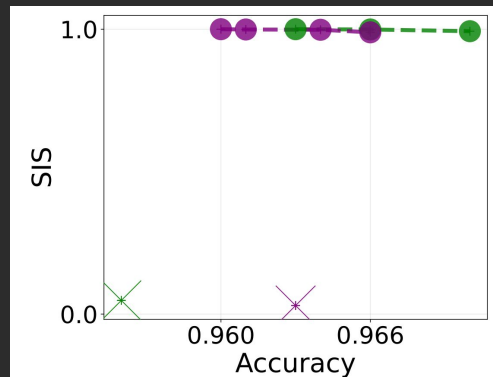
Expressive $p(Y|C,Z)$
incomplete concepts



Standard CBM training results
in uninterpretable CBMs

A bit of regularization vastly
improves interpretability

Expressive $p(Y|C,Z)$
complete concepts

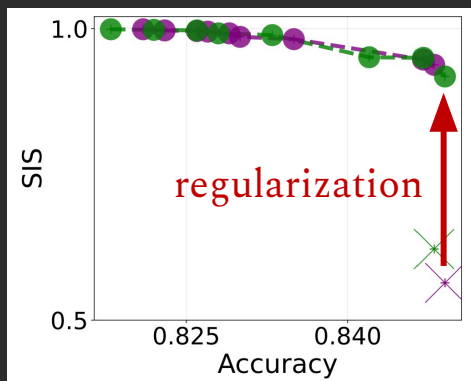


CBMs rely on the sidechannel
even when unnecessary

Not when regularizing for SIS

Sidechannels in practice

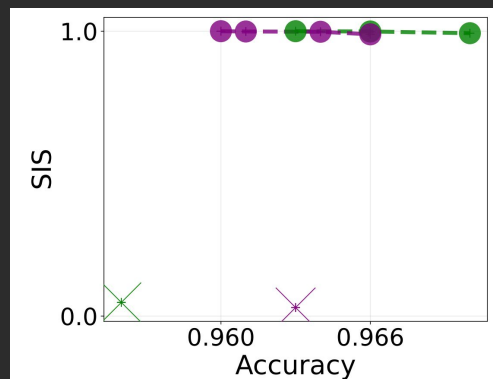
Expressive $p(Y|C,Z)$
incomplete concepts



Standard CBM training results
in uninterpretable CBMs

A bit of regularization vastly
improves interpretability

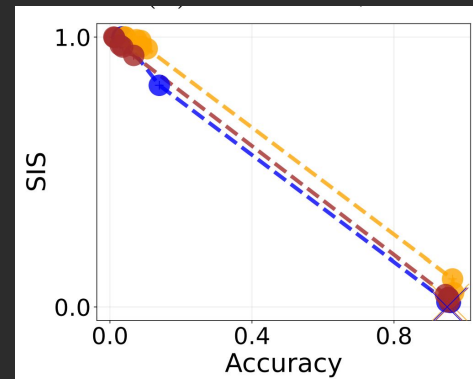
Expressive $p(Y|C,Z)$
complete concepts



CBMs rely on the sidechannel
even when unnecessary

Not when regularizing for SIS

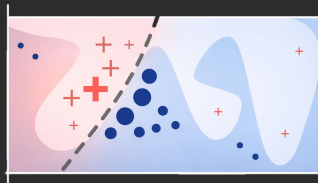
Non-expressive $p(Y|C,Z)$
complete concepts



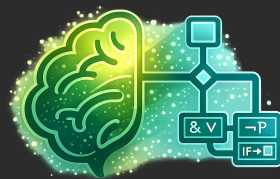
Do not use non-expressive task
predictors on expressive tasks

Accuracy and interpretability
are mutually exclusive then

Thank you



Locally-linear
Predictors



Neurosymbolic
Predictors



Causal
Predictors

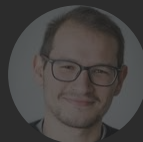
Tutorial Outline

I. Interpretability theory



~50 mins

II. Concept representations



~40 mins

III. Task predictions

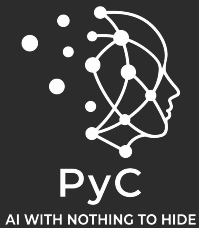


~50 mins

IV. PyTorch Concepts



~20 mins



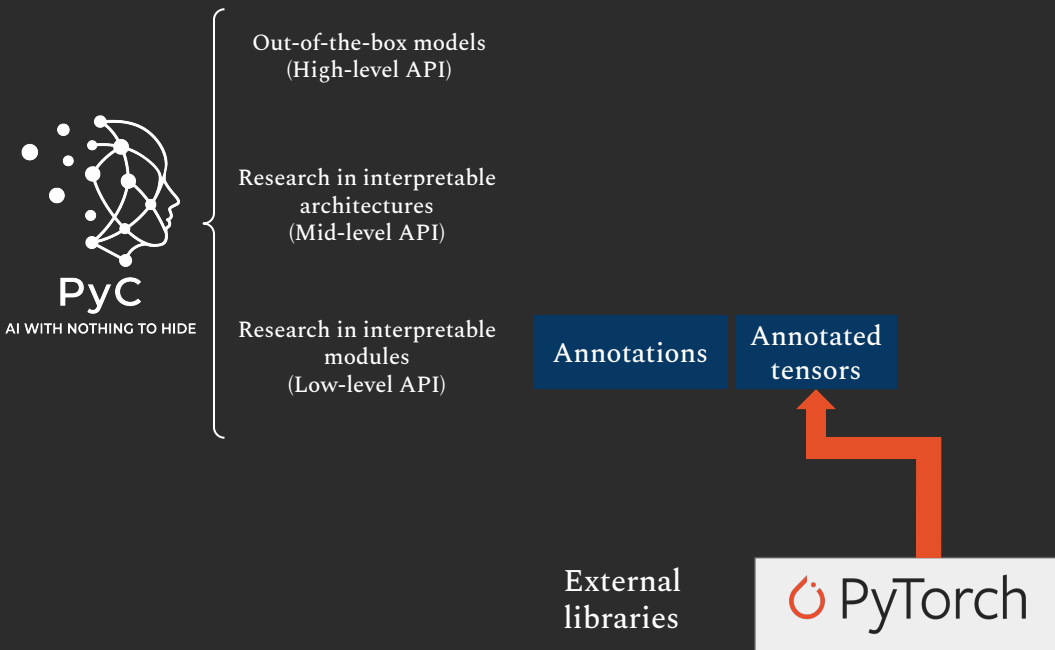
Out-of-the-box models
(High-level API)

Research in interpretable
architectures
(Mid-level API)

Research in interpretable
modules
(Low-level API)

External
libraries

 PyTorch



Annotated tensors

Annotations: define concept names and domains

smoking	genotype			tar
	A	B	C	

```
annotations = pyc.Annotations(  
    labels=["smoking", "genotype", "tar"],  
    cardinalities=[1, 3, 1],  
    types=["binary", "categorical", "continuous"],  
)
```

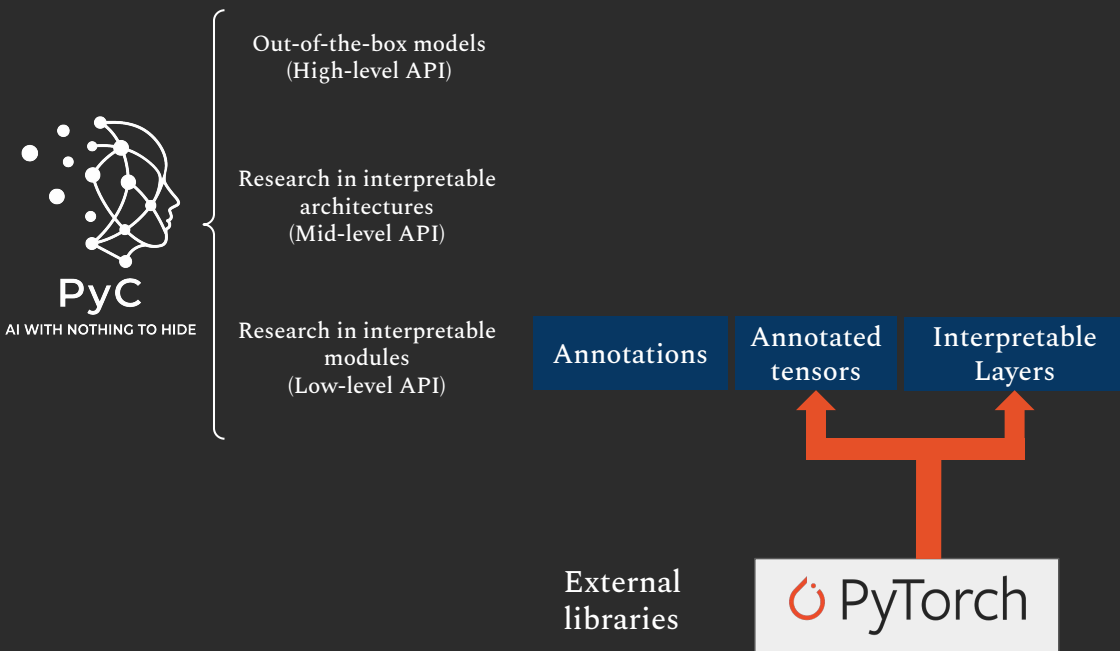
Annotated tensors

Annotations: define concept names and domains

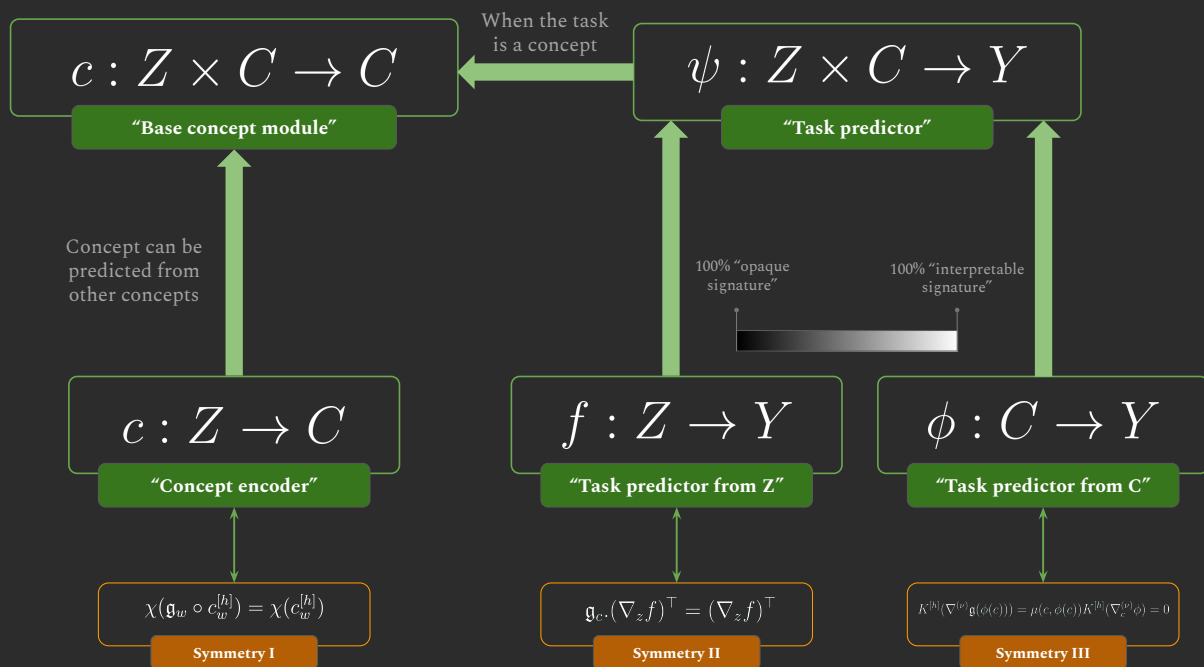
Annotated tensors: enable concept-based operations on torch tensors

smoking	genotype			tar
	A	B	C	
0	0	1	0	-3.1
1	1	0	0	4.2
0	1	0	0	2.3

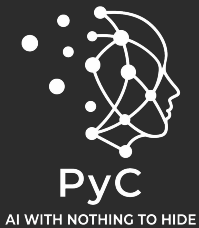
```
annotations = pyc.Annotations(  
    labels=["smoking", "genotype", "tar"],  
    cardinalities=[1, 3, 1],  
    types=["binary", "categorical", "continuous"],  
)  
  
tensor = pyc.AnnotatedTensor(  
    data=torch.randn(10, 5), # (batch_size, sum(cardinalities))  
    annotation=annotations,  
)  
  
smoking = tensor["smoking"] # slice by concept name  
binary = tensor.split_by_type("binary") # slice by concept type
```



Building blocks of interpretable models



```
class BaseConceptLayer(ABC, torch.nn.Module):
    """Abstract base class for concept layers..."""
    def __init__(
        self,
        out_concepts: Union[int, AxisAnnotation],
        in_concepts: Union[int, AxisAnnotation] = None,
        in_embeddings: Union[int, AxisAnnotation] = None,
        *args,
        **kwargs,
    ):
        pass
```



Out-of-the-box models
(High-level API)

Research in interpretable
architectures
(Mid-level API)

Research in interpretable
modules
(Low-level API)

Annotations

Annotated
tensors

Interpretable
Layers

Interventions

External
libraries

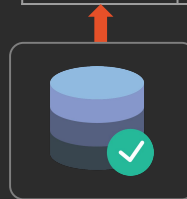
 PyTorch

Intervention strategies = how to intervene

smoking	genotype			tar
	A	B	C	
0.4	0	1	0	0.1
0.3	1	0	0	0.9
0.8	1	0	0	0.5



smoking	genotype			tar
	A	B	C	
0	0	1	0	0.1
1	1	0	0	0.9
0	1	0	0	0.5



ground truth data

Intervention strategies = how to intervene

smoking	genotype			tar
	A	B	C	
0.4	0	1	0	0.1
0.3	1	0	0	0.9
0.8	1	0	0	0.5



smoking	genotype			tar
	A	B	C	
0	0	1	0	1
1	1	0	0	1
0	1	0	0	1



ground truth data

$do(c_{\text{tar deposit}} = 1)$

Intervention policies = where to intervene

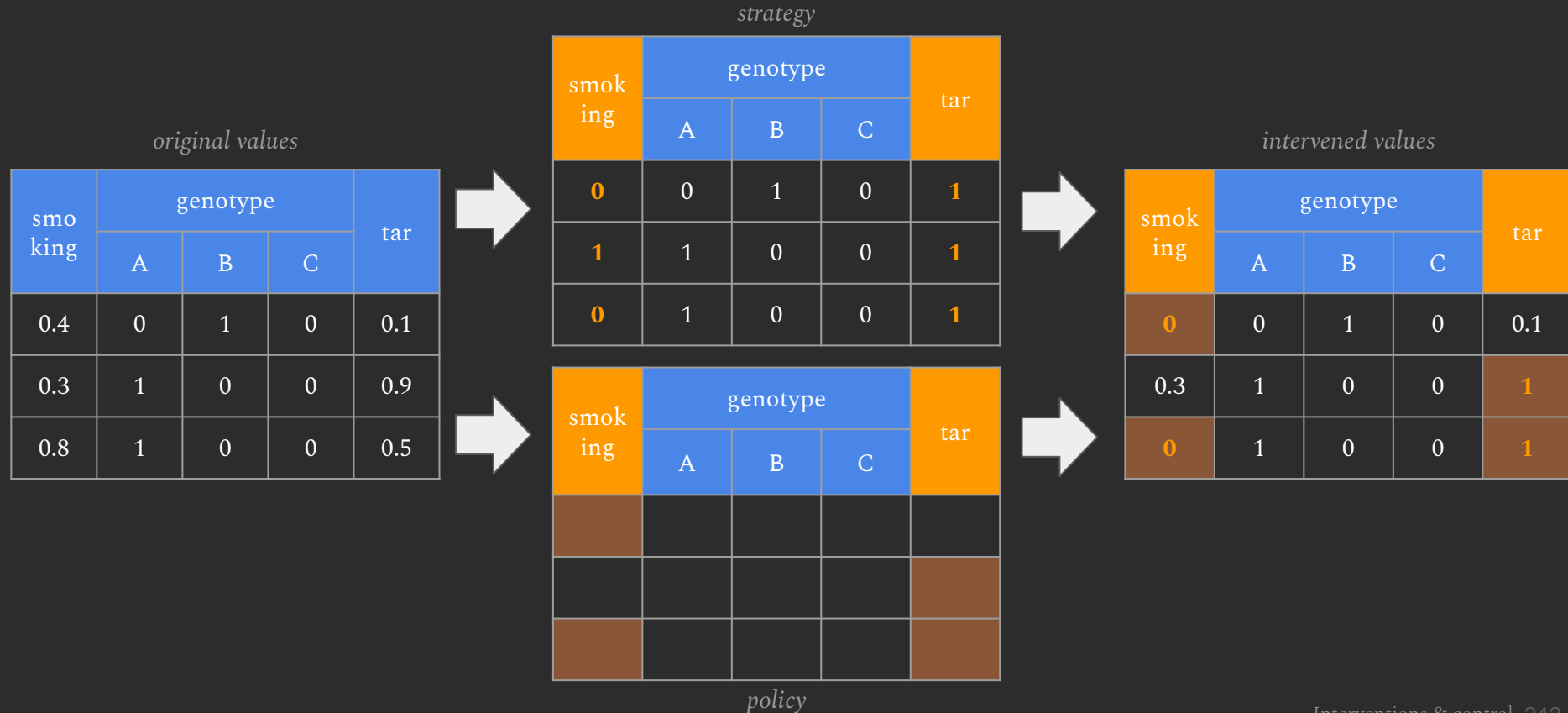
smoking	genotype			tar
	A	B	C	
0.4	0	1	0	0.1
0.3	1	0	0	0.9
0.8	1	0	0	0.5



smoking	genotype			tar
	A	B	C	

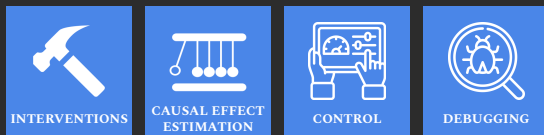
Random policy

Intervention = strategy + policy

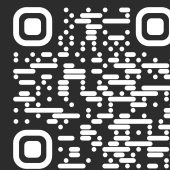


Interventions & control

INTERPRETABLE MODELS SUPPORT

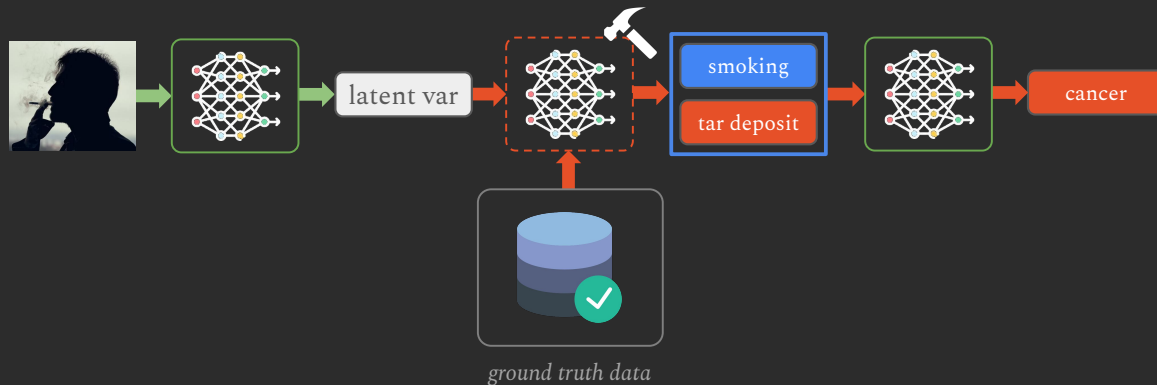


```
embedding_encoder = torch.nn.Sequential(  
    torch.nn.Linear(n_features, embedding_dims),  
    torch.nn.LeakyReLU(),  
)  
c_encoder = pyc.nn.LinearEmbeddingToConcept(  
    in_embeddings=embedding_dims,  
    out_concepts=annotations  
)  
y_predictor = pyc.nn.LinearConceptToConcept(  
    in_concepts=concept_dims,  
    out_concepts=task_dims  
)  
model = torch.nn.ModuleDict({  
    "embedding_encoder": embedding_encoder,  
    "concept_encoder": c_encoder,  
    "task_predictor": y_predictor  
})  
embeddings = model["embedding_encoder"](x_train)  
c_pred = model["concept_encoder"](embeddings)  
y_pred = model["task_predictor"](c_pred)
```

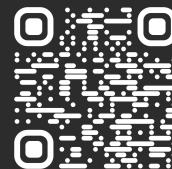


Interventions & control

INTERPRETABLE MODELS SUPPORT

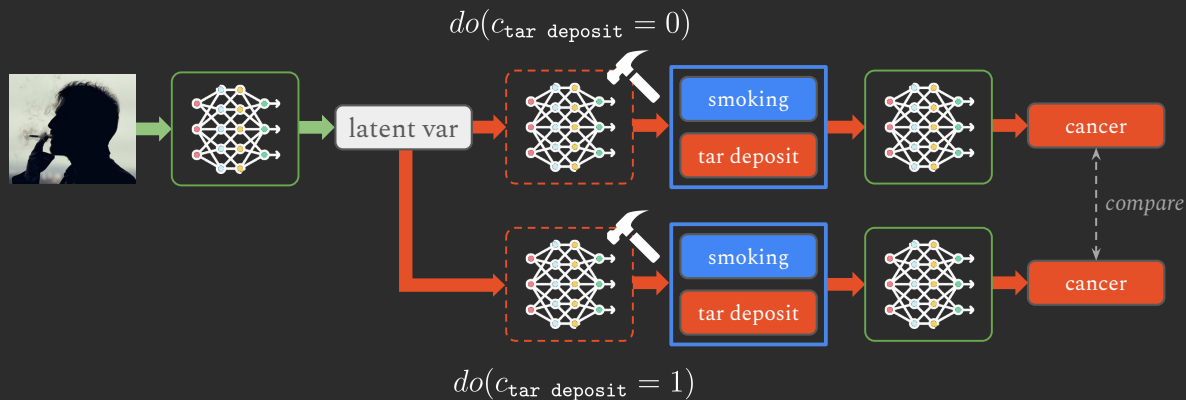


```
concept_encoder_intervened = pyc.nn.InterventionModule(  
    model["concept_encoder"],  
    intervention_strategy=pyc.nn.GroundTruthIntervention(c_train),  
    intervention_policy=pyc.nn.UniformPolicy(),  
    out_concepts_to_intervene_on=["tar"]  
)  
c_pred_intervened = concept_encoder_intervened(embeddings)
```



Interventions & control

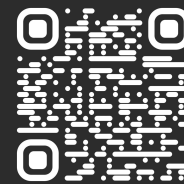
INTERPRETABLE MODELS SUPPORT



```
concept_encoder_intervened_1 = pyc.nn.InterventionModule(
    model["concept_encoder"],
    intervention_strategy=pyc.nn.DoIntervention(constants=1.0),
    intervention_policy=pyc.nn.UniformPolicy(),
    out_concepts_to_intervene_on=["tar"]
)
c_pred_intervened_1 = concept_encoder_intervened_1(embeddings)

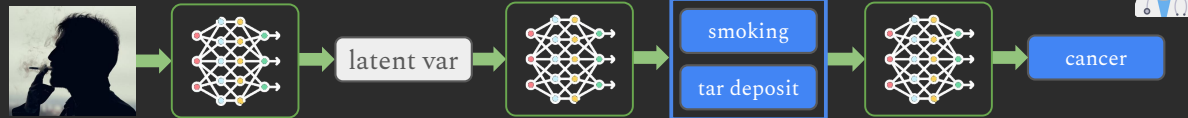
concept_encoder_intervened_0 = pyc.nn.InterventionModule(
    model["concept_encoder"],
    intervention_strategy=pyc.nn.DoIntervention(constants=0.0),
    intervention_policy=pyc.nn.UniformPolicy(),
    out_concepts_to_intervene_on=["tar"]
)
c_pred_intervened_0 = concept_encoder_intervened_0(embeddings)

avg_treatment_effect = pyc.nn.functional.cace_score(
    c_pred_intervened_0,
    c_pred_intervened_1
)
```



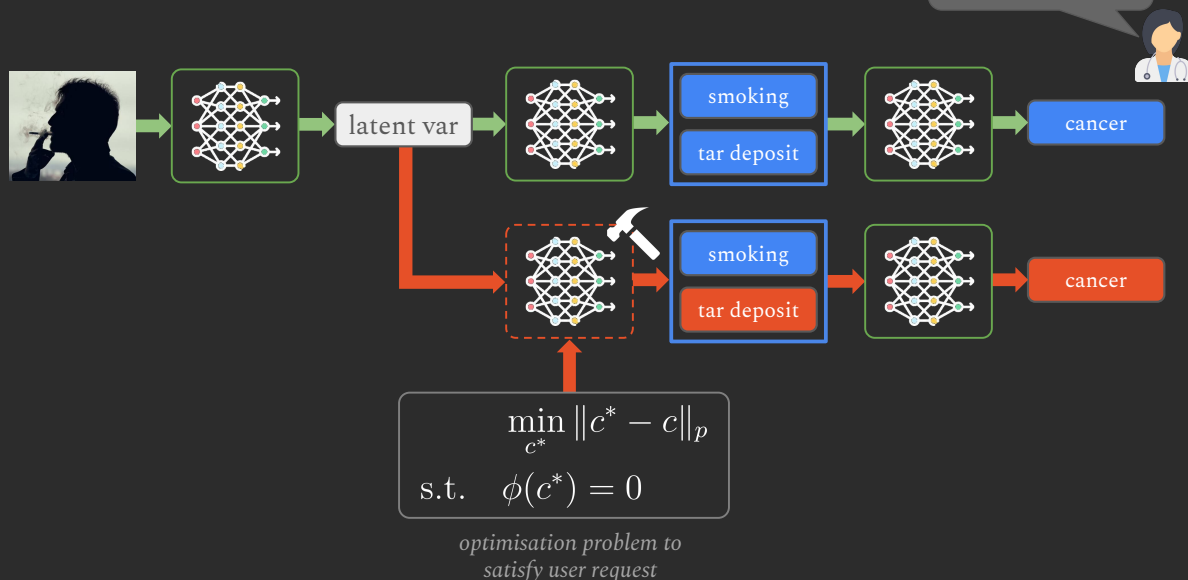
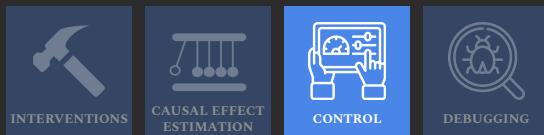
Interventions & control

INTERPRETABLE MODELS SUPPORT



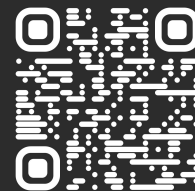
Interventions & control

INTERPRETABLE MODELS SUPPORT



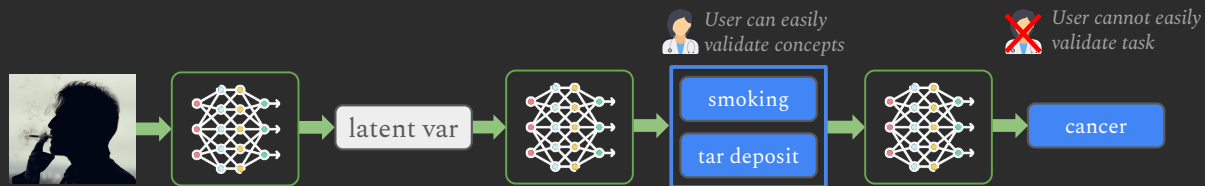
```

y_target = torch.ones(batch_size, out_module_size)
concept_encoder_intervened = pyc.nn.InterventionModule(
    model["concept_encoder"],
    pyc.nn.ContrastiveIntervention(
        out_target_tensor=y_target,
        x_to_out_module=model["task_predictor"],
    ),
    pyc.nn.UniformPolicy(),
)
c_pred_contrastive = concept_encoder_intervened(
    embeddings[:batch_size]
)
y_pred_contrastive = model["task_predictor"](
    c_pred_contrastive
)
    
```



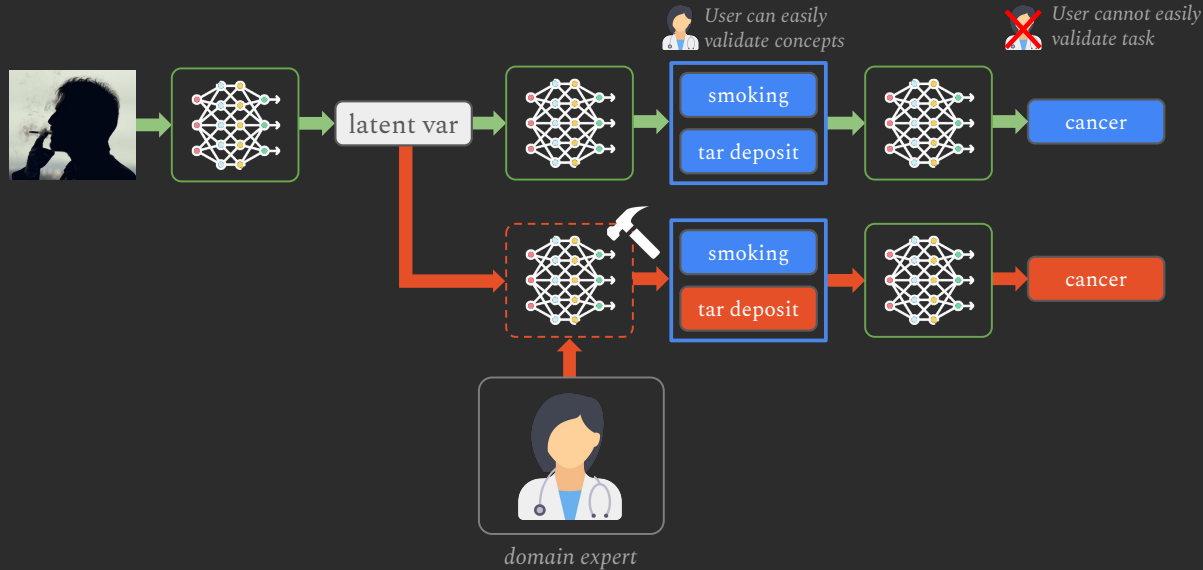
Interventions & control

INTERPRETABLE MODELS SUPPORT

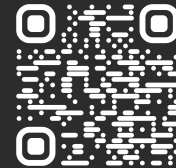


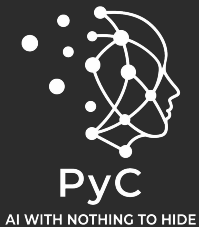
Interventions & control

INTERPRETABLE MODELS SUPPORT



```
concept_encoder_intervened = pyc.nn.InterventionModule(  
    model["concept_encoder"],  
    intervention_strategy=pyc.nn.GroundTruthIntervention(c_train),  
    intervention_policy=pyc.nn.UniformPolicy(),  
    out_concepts_to_intervene_on=["tar"]  
)  
c_pred_intervened = concept_encoder_intervened(embeddings)
```

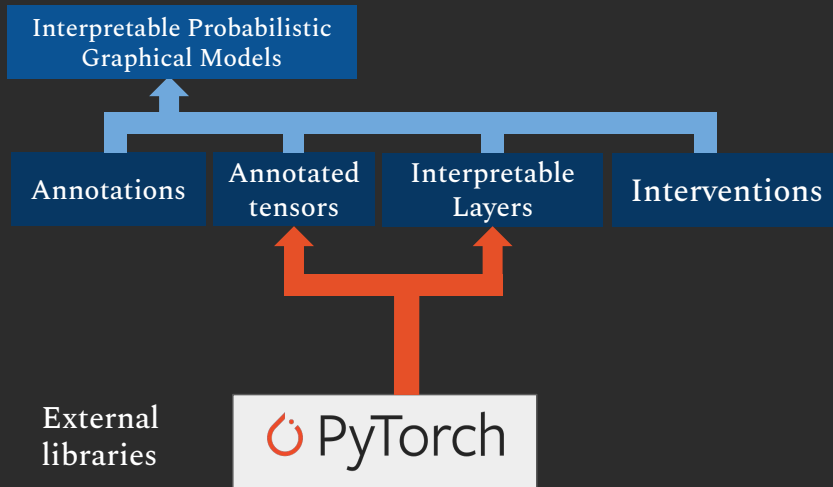




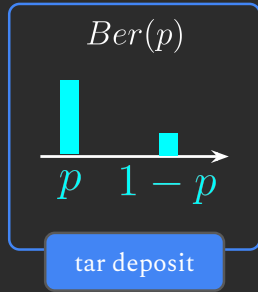
Out-of-the-box models
(High-level API)

Research in interpretable
architectures
(Mid-level API)

Research in interpretable
modules
(Low-level API)

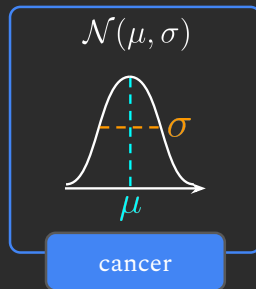
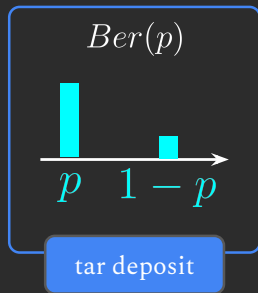


Random variables



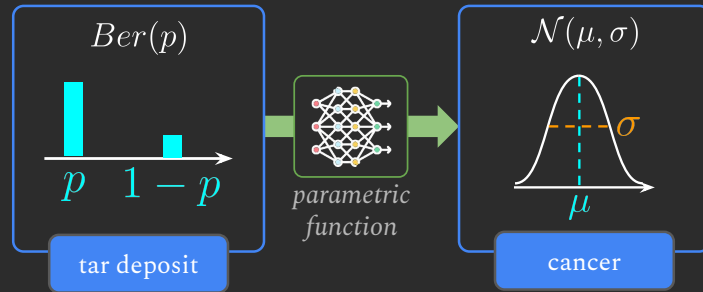
```
tar = pyc.ConceptVariable(  
    names: "tar",  
    distribution=Bernoulli  
)
```

Random variables

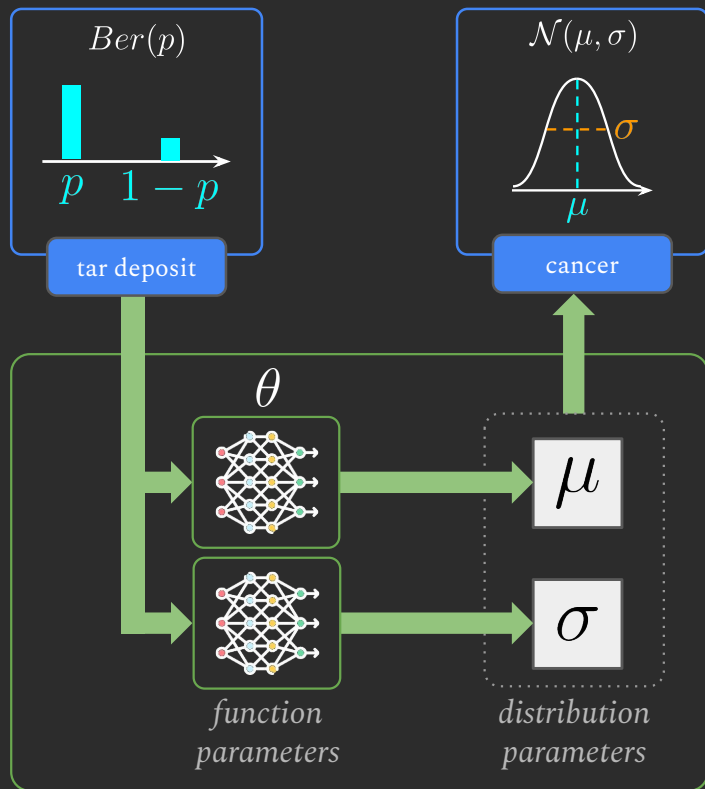


```
tar = pyc.ConceptVariable(  
    names: "tar",  
    distribution=Bernoulli  
)  
mutations = pyc.ConceptVariable(  
    names: "cancer",  
    distribution=Normal  
)
```


Parametric conditional probability distributions

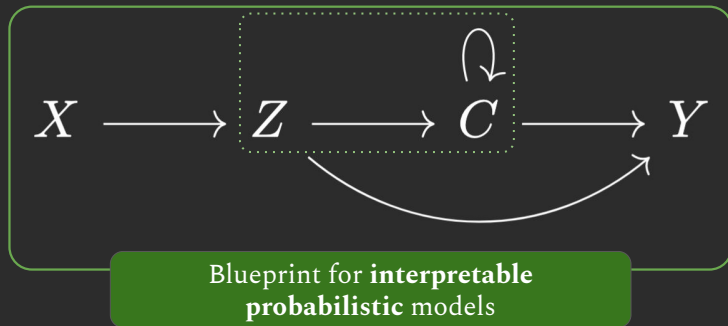


Parametric conditional probability distributions

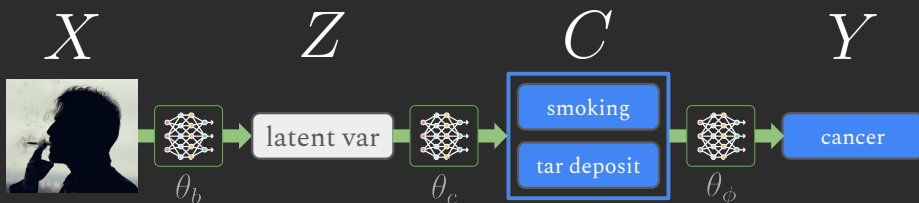


```
cancer_cpd = pyc.nn.ParametricCPD(  
    cancer,  
    parents=[tar, smoking],  
    parametrization={  
        "loc": pyc.nn.LinearConceptToConcept(in_concepts=2, out_concepts=1),  
        "scale": pyc.nn.Sequential(  
            *args: pyc.nn.LinearConceptToConcept(in_concepts=2, out_concepts=1),  
            torch.nn.Softplus()  
        )  
    }  
)
```

Interpretable probabilistic models



PROBABILISTIC CONCEPT BOTTLENECK MODEL



$$p(y, c \mid x; \theta_\phi, \theta_c, \theta_b) = \int p(y \mid c; \theta_\phi) p(c \mid z; \theta_c) p(z \mid x; \theta_b) dz$$

```

x_var = pyc.nn.EmbeddingVariable(
    names: "x",
    distribution=Delta,
    size=n_features
)

c_var = pyc.nn.ConceptVariable(
    names: "concepts",
    members=["tar", "smoking"],
    distribution=Bernoulli
)

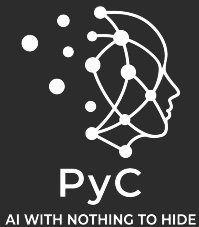
y_var = pyc.nn.ConceptVariable( names: "cancer", distribution=Normal)

input_cpd = pyc.nn.ParametricCPD(
    x_var, parents=[],
    parametrization=pyc.nn.LearnablePrior(size=1)
)

concepts_cpd = pyc.nn.ParametricCPD(
    c_var, parents=[x_var],
    parametrization={
        'logits': pyc.nn.LinearEmbeddingToConcept(n_features, out_concepts=2)
    }
)

cancer_cpd = pyc.nn.ParametricCPD(
    cancer,
    parents=[c_var],
    parametrization={
        "loc": pyc.nn.LinearConceptToConcept(in_concepts=2, out_concepts=1),
        "scale": pyc.nn.Sequential(
            *args: pyc.nn.LinearConceptToConcept(in_concepts=2, out_concepts=1),
            torch.nn.Softplus()
        )
    }
)

model = pyc.nn.BayesianNetwork(
    variables=[x_var, c_var, y_var],
    factors=[input_cpd, concepts_cpd, cancer_cpd]
)
    
```

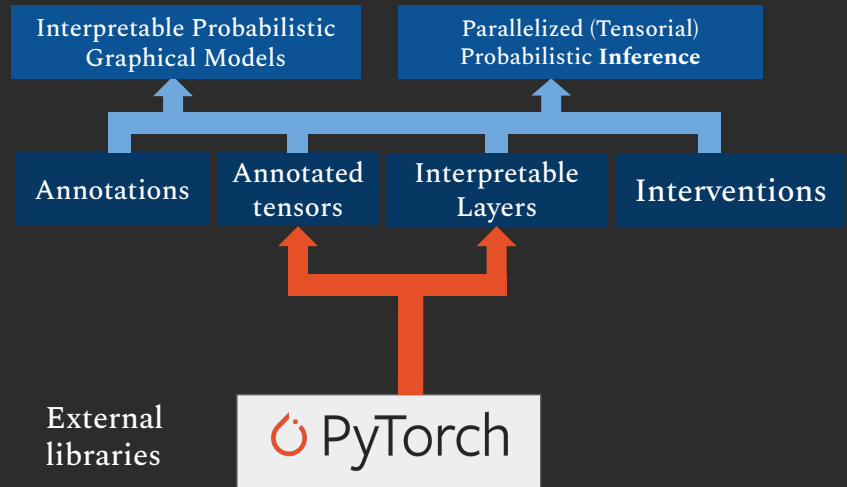


AI WITH NOTHING TO HIDE

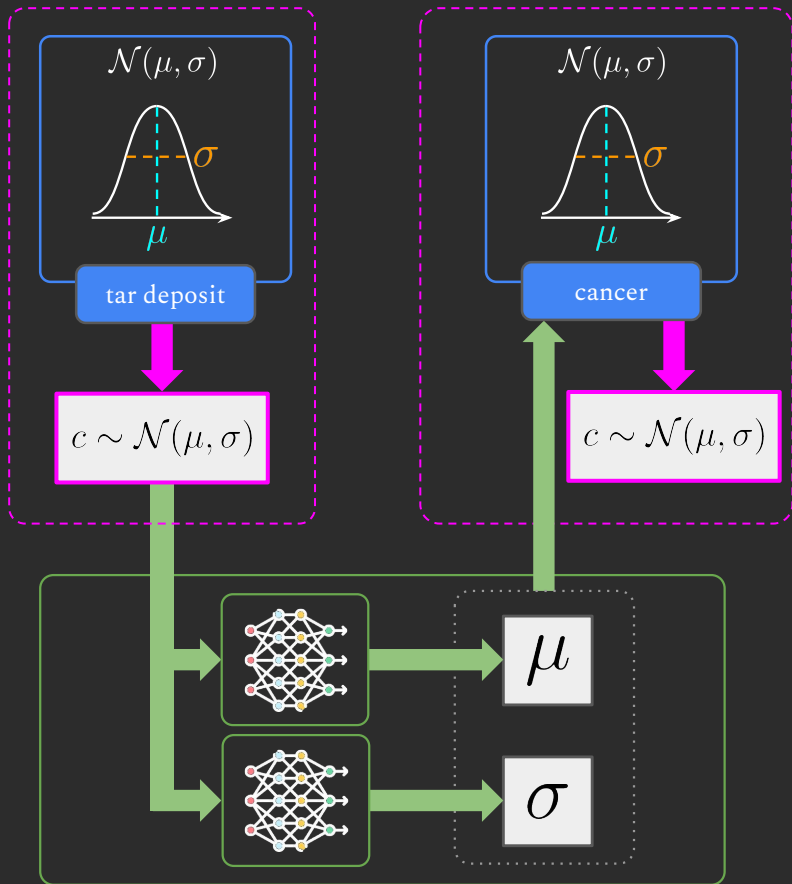
Out-of-the-box models
(High-level API)

Research in interpretable
architectures
(Mid-level API)

Research in interpretable
modules
(Low-level API)



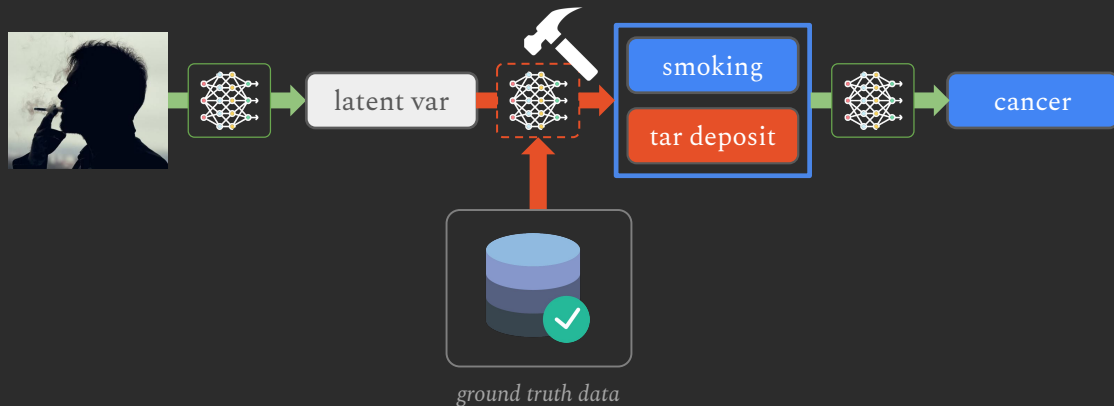
Ancestral sampling



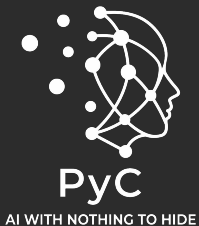
```
model = pyc.nn.BayesianNetwork(  
    variables=[x_var, c_var, y_var],  
    factors=[input_cpd, concepts_cpd, cancer_cpd]  
)  
  
inference = pyc.nn.AncestralSamplingInference(model)  
cancer_pred_sampled = inference.query(  
    query=["cancer"],  
    evidence={  
        "tar": c_true["tar"].tensor,  
    }  
)  
  
inference = pyc.nn.DeterministicInference(model)  
cancer_pred_deterministic = inference.query(  
    query=["cancer"],  
    evidence={  
        "tar": c_true["tar"].tensor,  
    }  
)
```

Interventional inference

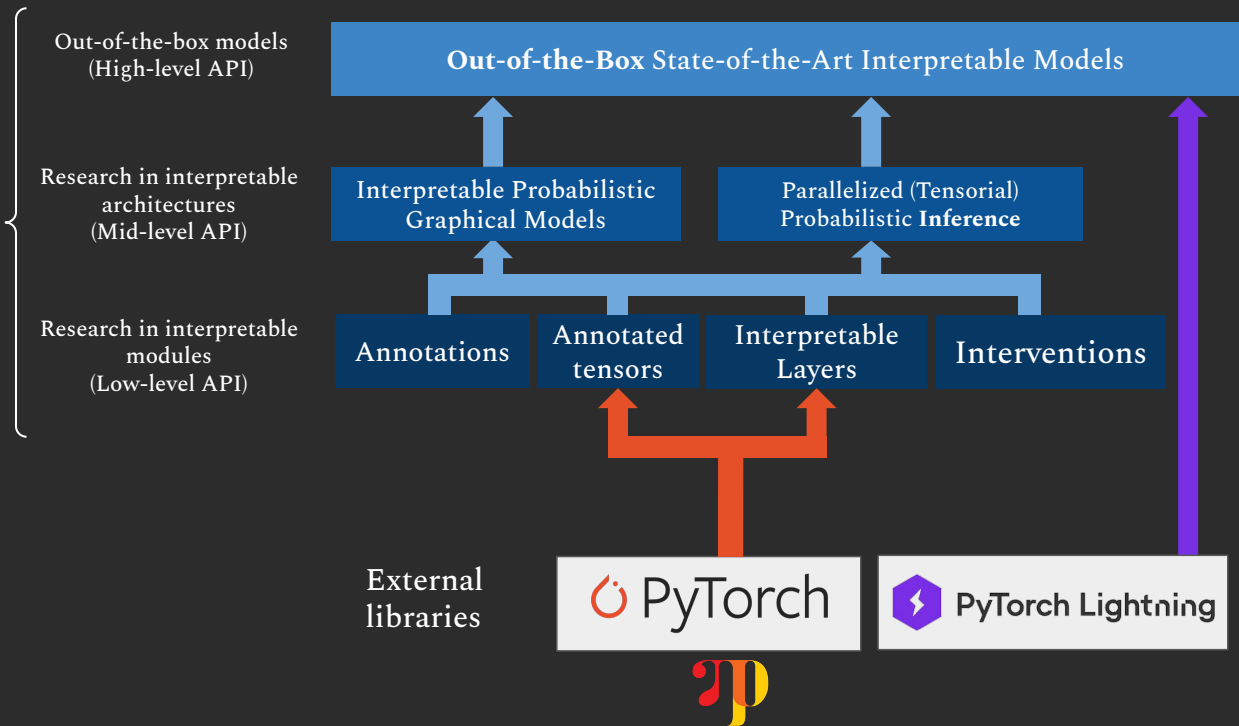
INTERPRETABLE MODELS SUPPORT



```
with pyc.nn.intervention(  
    model,  
    pyc.nn.GroundTruthIntervention(c_true["tar"].tensor),  
    pyc.nn.UniformPolicy(),  
    variable_to_intervene_on="concepts",  
    parameter_to_intervene_on="logits",  
    members_to_intervene_on=["tar"]  
):  
    cancer_pred_intervened = inference.query(  
        query=["cancer"],  
        evidence={  
            "tar": c_true["tar"].tensor,  
        }  
    )
```

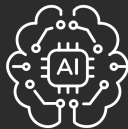


AI WITH NOTHING TO HIDE



High-level API

Torch-style



Out-of-the-box Models

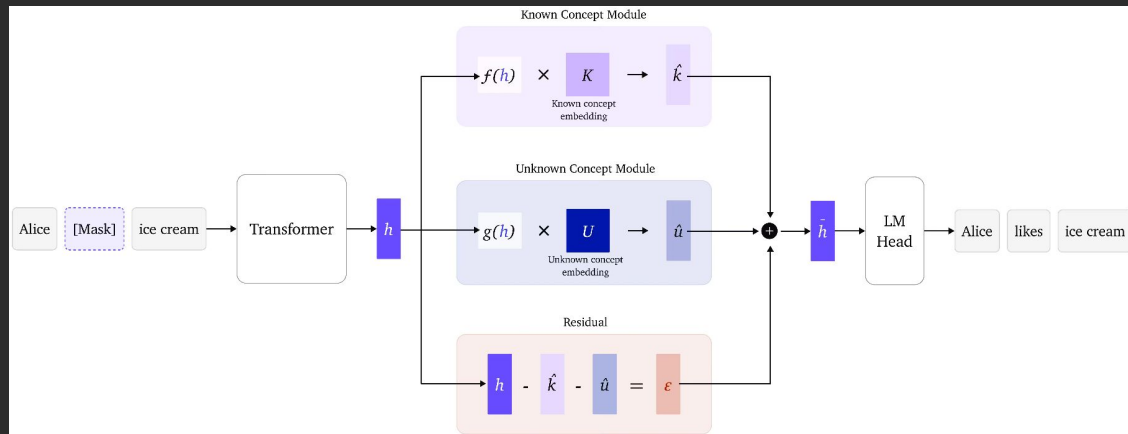
Lightning-style



```
# Initialize the CBM
model = ConceptBottleneckModel(
    input_size=n_features,
    annotations=concept_annotations,
    variable_distributions=variable_distributions,
    task_names=task_names,
    latent_encoder_kwargs={'hidden_size': 16, 'n_layers': 1}
)
```

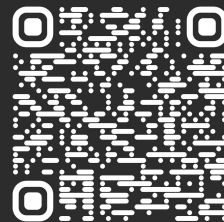
```
model = ConceptBottleneckModel(
    input_size=n_features,
    annotations=annotations,
    variable_distributions=variable_distributions,
    task_names=['xor'],
    latent_encoder_kwargs={'hidden_size': 16, 'n_layers': 1},
    # Inference engines (both default to DeterministicInference)
    inference=DeterministicInference,
    train_inference=DeterministicInference,
    # Lightning kwargs
    lightning=True,
    loss=loss,
    optim_class=optim,
    optim_kwargs=optim_kwargs
)
```


Steerling-8B



```
from torch_concepts.steerling import SteerlingModel

steerling_model = SteerlingModel(
    pretrained_components=['backbone', 'known_head', 'unknown_head', 'lm_head'],
    freeze_components=['backbone', 'known_head', 'unknown_head', 'lm_head'],
    use_unknown=True,
    use_epsilon_correction=False,
)
```

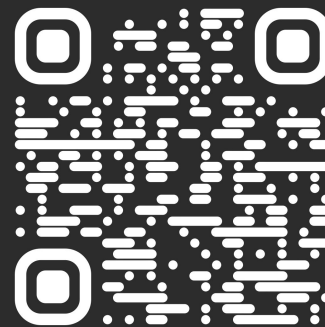
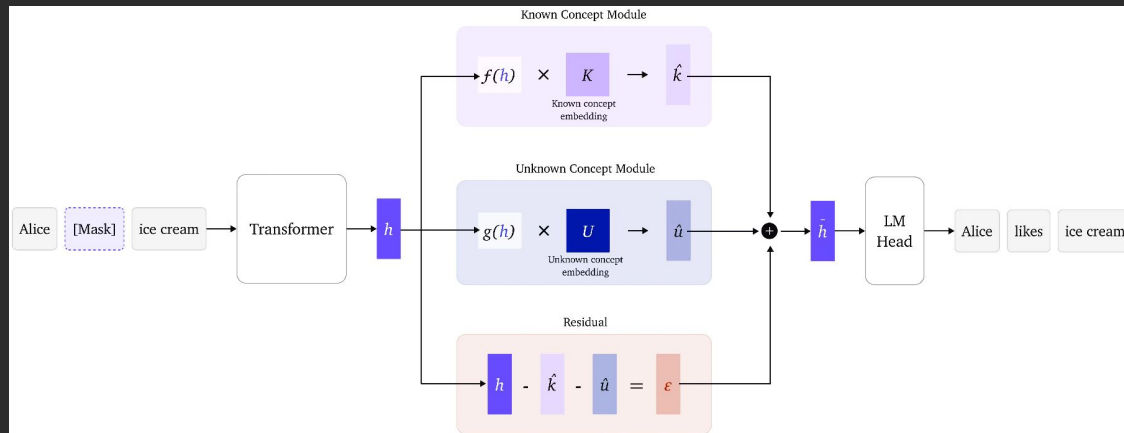


Concept-based LLM with 8Bn parameter
by



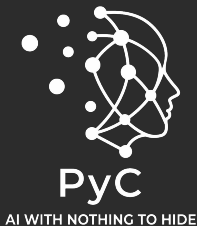
GuideLabs

Baby Steerling

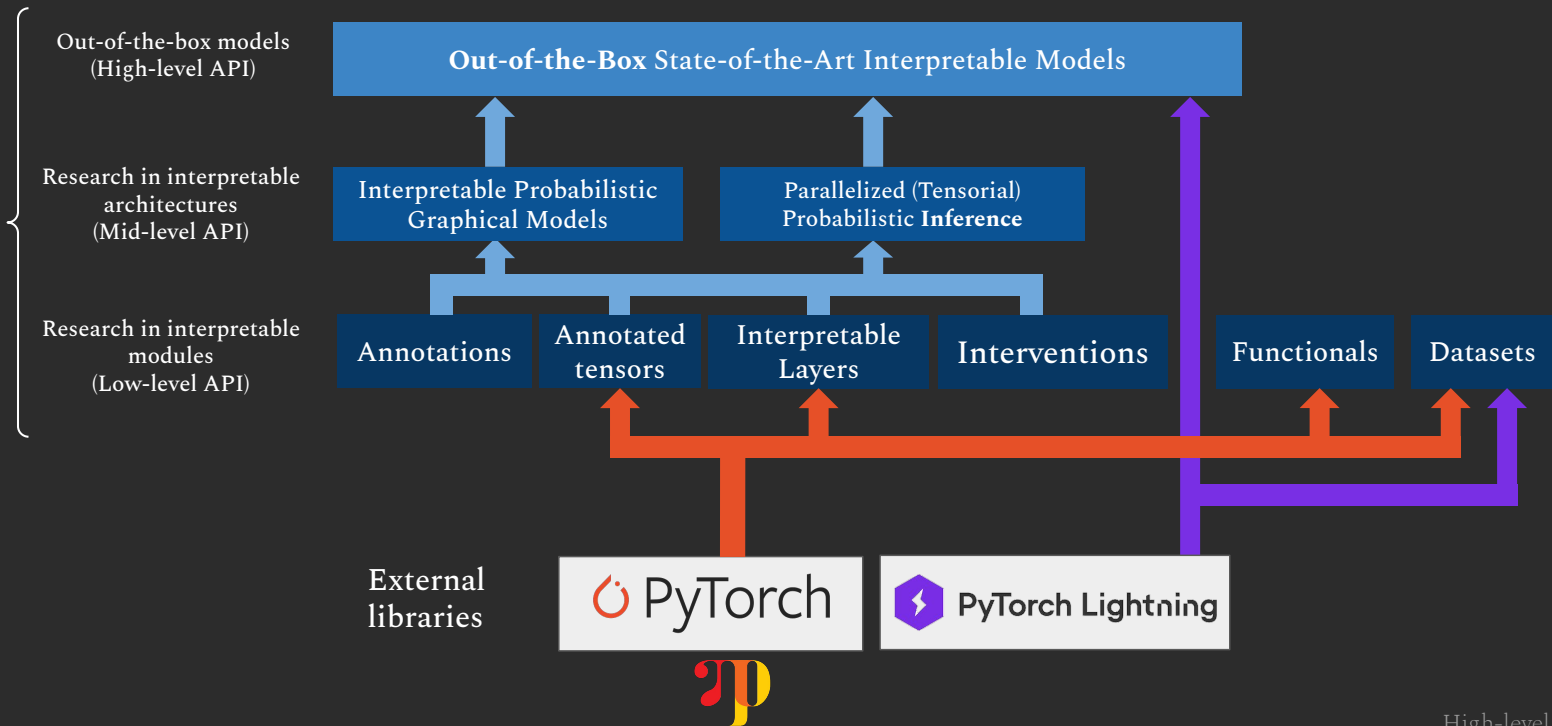


Concept-based LLM trainable on a laptop in few mins
by





AI WITH NOTHING TO HIDE



Loss

Automatic type-aware routing:

```
loss = ConceptLoss(  
    annotations=annotations,  
    binary=[  
        nn.BCEWithLogitsLoss(),  
        L1LogitRegularizer(scale=0.01)  
    ],  
    binary_weights = [1.0, 0.5],  
    categorical = nn.CrossEntropyLoss(),  
    continuous = nn.MSELoss()  
)
```

Pass it to the model:

```
# Initialize the CBM  
model = ConceptBottleneckModel(  
    input_size=n_features,  
    annotations=annotations,  
    variable_distributions=variable_distributions,  
    task_names=['xor'],  
    latent_encoder_kwargs={'hidden_size': 16, 'n_layers': 1},  
    lightning=True,  
    loss=loss_fn,  
    metrics=metrics,  
    optim_class=torch.optim.AdamW,  
    optim_kwargs={'lr': 0.02}  
)
```

Metrics

Support for:



PyTorch metrics
(concept accuracy, ...)



PyC metrics
(interventions, leakage, ...)

Automatic type-aware routing:

```
metrics = ConceptMetrics(  
    annotations=annotations,  
    summary=True,      # enable averaged metrics across all concepts of the same type  
    per_concept=True,  # enable per-concept metrics  
    binary={  
        # Pre-instantiated  
        'accuracy': torchmetrics.classification.BinaryAccuracy(),  
        # Class + kwargs  
        'f1': (torchmetrics.classification.BinaryF1Score, {'threshold': 0.5}),  
        # Class only  
        'precision': torchmetrics.classification.BinaryPrecision  
    },  
    categorical={  
        # Class + kwargs  
        'accuracy': (torchmetrics.classification.MulticlassAccuracy, {'average': 'micro'}),  
        # Class only  
        'f1': torchmetrics.classification.MulticlassF1Score  
    }  
)
```



Applications & benchmarking
(No-code API)

Out-of-the-box models
(High-level API)

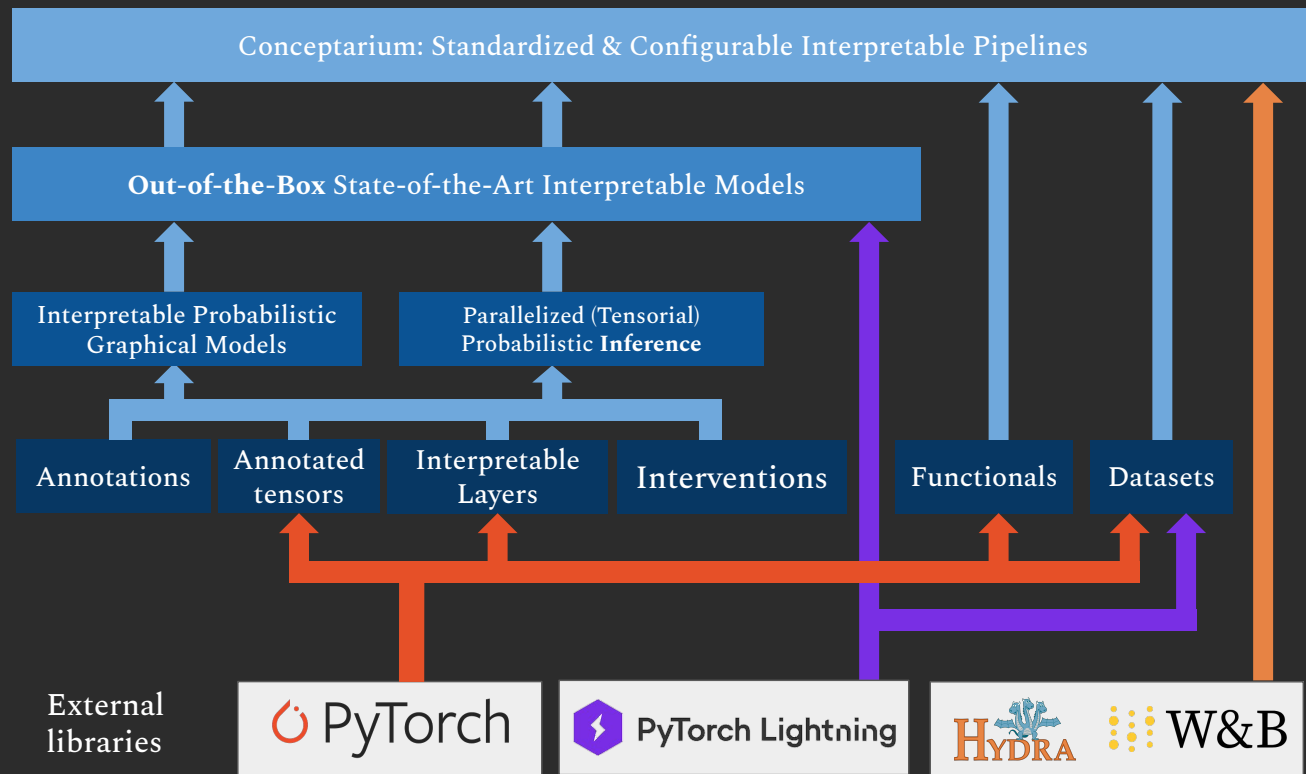
Research in interpretable
architectures
(Mid-level API)

Research in interpretable
modules
(Low-level API)



PyC

AI WITH NOTHING TO HIDE





conceptarium



Main cfg:



```
defaults:
  - _default
  - _self_

hydra:
  job:
    name: test
  sweeper:
    # standard grid search
    params:
      seed: 1,2,3,4,5
      dataset: celeba, asia, sachs
      model: cbm, cem
      loss: weighted
      loss.concept_weight: 8
      loss.task_weight: 1
```

Model cfg:



```
defaults:
  - _commons
  - _self_

# wrapper for 'joint' training mode
_target_: "torch_concepts.nn.ConceptBottleneckModel"

task_names: ${dataset.default_task_names}

inference:
  _target_: "torch_concepts.nn.DeterministicInference"
  _partial_: true
```

Dataset cfg:



```
defaults:
  - _self_

_target_: torch_concepts.data.datamodules.celeba.CelebADataModule

name: celeba

# backbone handling and embedding precomputation
backbone: facebook/dinov2-base
precompute_embs: true
force_recompute: false

# Task label - which CelebA attribute to predict
default_task_names: [Attractive]

# splitter
splitter:
  _target_: torch_concepts.data.splitters.native.NativeSplitter

# all CelebA attributes are binary facial features
label_descriptions: null
```

...

And then...

```
CUDA_VISIBLE_DEVICES=1 python conceptarium/run_experiment.py --config-name sweep
```

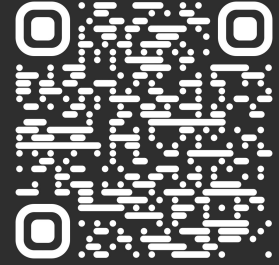
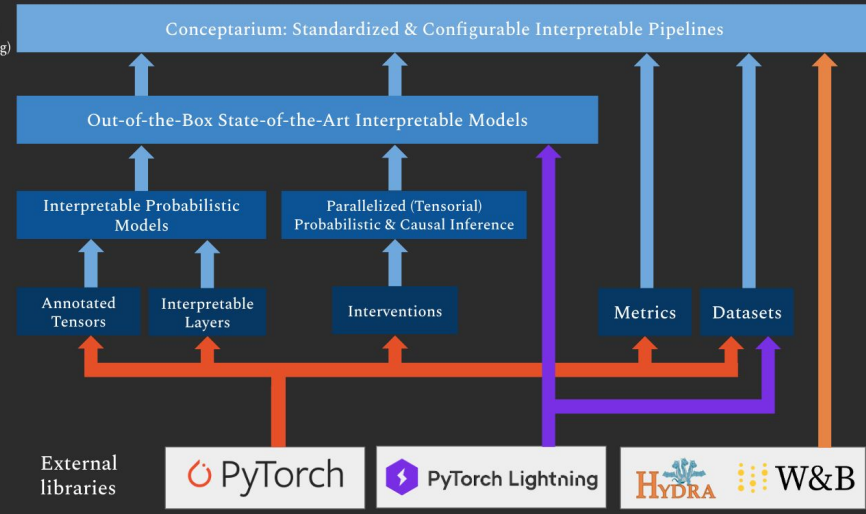


No-code APIs
(end2end platforms for AI
applications & benchmarking)

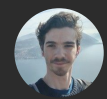
High-level APIs
(plug & play interpretable
models)

Mid-level APIs
(research in interpretable
architectures)

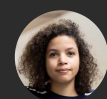
Low-level APIs
(research in interpretable
modules)



github link



Giovanni De Felice (PI)
Università della Svizzera Italiana (CH)



Arianna Casanova
University of Liechtenstein (LI)



Gabriele Ciravegna
Intesa Sanpaolo Innovation Center (IT)



David Debot
KU Leuven (BE)



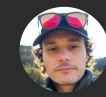
Giuseppe Marra
KU Leuven (BE)



Francesco De Santis
Politecnico di Torino (IT)



Mateo Espinosa Zarlenga
University of Oxford (UK)



Edoardo Gabrielli
Sapienza University of Rome (IT)



Tuomas Oikarinen
UC San Diego (US)



Pietro Barbiero
IBM Research