

# Foundations of Interpretable Models

Pietro Barbiero & Mateo Espinosa Zarlenga

[pietro.barbiero@ibm.com](mailto:pietro.barbiero@ibm.com)

[me466@cam.ac.uk](mailto:me466@cam.ac.uk)

In collaboration with: Alberto Termine, Mateja Jamnik, Giuseppe Marra



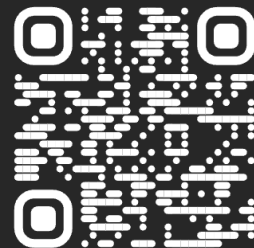
UNIVERSITY OF  
CAMBRIDGE

IBM Research



PyC

AI WITH NOTHING TO HIDE



KU LEUVEN

idsia

Istituto Dalle Molle di studi  
sull'intelligenza artificiale  
USI - SUPSI

# Who are we?



**Pietro Barbiero**

Swiss Postdoctoral Fellow  
IBM Research (Switzerland)  
pietro.barbiero@ibm.com

IBM Research



**Mateo Espinosa Zarlenga**

Final-year PhD Student  
University of Cambridge  
me466@cam.ac.uk



UNIVERSITY OF  
CAMBRIDGE

# Outline



I. What is interpretability?

II. Foundations

1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models



III. Instantiations

1. Instantiating the concept encoder  $P(C | X)$
2. Instantiating the task predictor  $P(Y | C)$

IV. Open questions

# Outline

## I. What is interpretability?

## II. Foundations

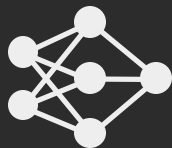
1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models

## III. Instantiations

1. Instantiating the concept encoder  $P(C \mid X)$
2. Instantiating the task predictor  $P(Y \mid C)$

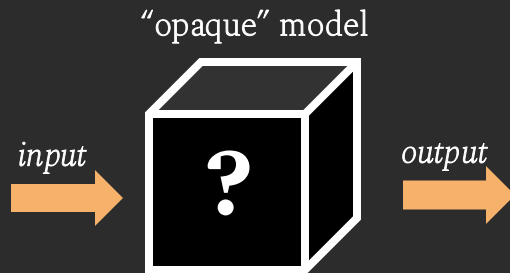
## IV. Open questions

# Why do we need interpretability in the first place?

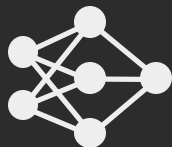


- Highly parametric
- Complicated inference steps
- Non-linearities
- Sensitive to initial states and update rules

$\approx$

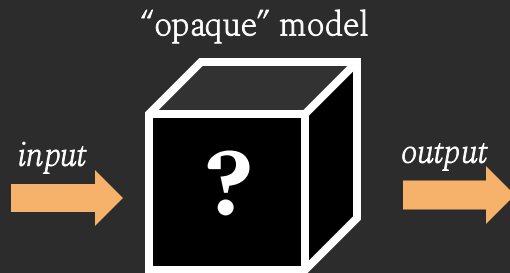


# Why do we need interpretability in the first place?



- 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*

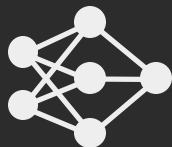
In what may be the first known case of its kind, a faulty facial recognition match led to a Michigan man's arrest for a crime he did not commit.

[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 \(2018\).](#)

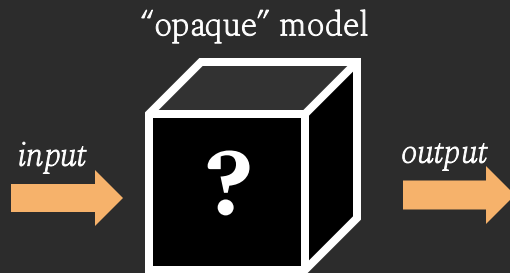
[3] [Will Douglas Heaven, "Predictive policing algorithms are racist. They need to be dismantled," MIT Technology Review \(2020\).](#)

# 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." \(2024\).](#)

# What is interpretability in AI?

*“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 \(2023\).](#)



# What is interpretability... in other fields?

*“A theory  $T$  is **interpretable** in a theory  $S$  if and only if there exists a **translation** from the language of  $T$  into the language of  $S$  such that every theorem of  $T$  is translated into a theorem of  $S$ ”*



**Formal**



**Comes with “strings attached”**



**Not contextualized in modern AI**



**Unclear consequences in modern AI**

### **Problem #1**

*Lack of formal and “actionable” definition of interpretability.*

*Research in interpretability is ill-defined.*




### **Research Question #1**

*Can we provide a general, simple & actionable definition of interpretability in AI?*

Is the model  $P(Y \mid X; m)$  interpretable?

<i>object</i>	<i>features</i>		<i>target</i>
$\omega$	$X_1$	$X_2$	$Y = m(X)$
	0	1	1
	0	0	1
	1	0	0

Is the model  $P(Y \mid X; m)$  interpretable?

$\omega$	$X_1$	$X_2$	$Y = m(X)$
	0	1	1
	0	0	1
	1	0	0

Let's associate:

- “one” to  $X_1$
- “red” to  $X_2$
- “even” to  $Y$



**Relation** between the **model**  
and **human knowledge**  
(number theory)!

Is the model  $P(Y \mid X; m)$  interpretable?

$\omega$	$X_1$	$X_2$	$Y = m(X)$
0	0	1	1
0	0	0	1
1	1	0	0

$$(X_1, X_2) = (0, 0) \xrightarrow{\text{unknown function } m} Y = 1$$

Is the model  $P(Y \mid X; m)$  interpretable?

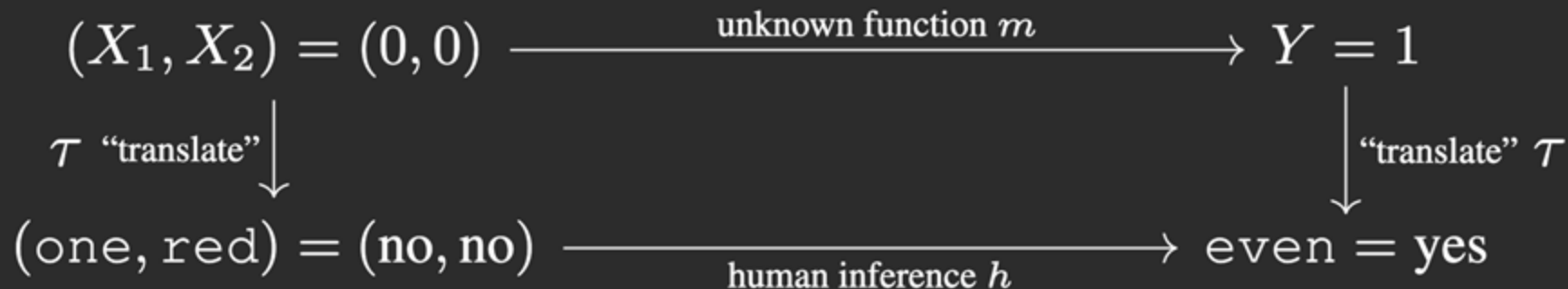
$\omega$	$X_1$	$X_2$	$Y = m(X)$
0	0	1	1
0	0	0	1
1	1	0	0

$(X_1, X_2) = (0, 0) \xrightarrow{\text{unknown function } m} Y = 1$

$(\text{one}, \text{red}) = (\text{no}, \text{no}) \xrightarrow{\text{human inference } h} \text{even} = \text{yes}$

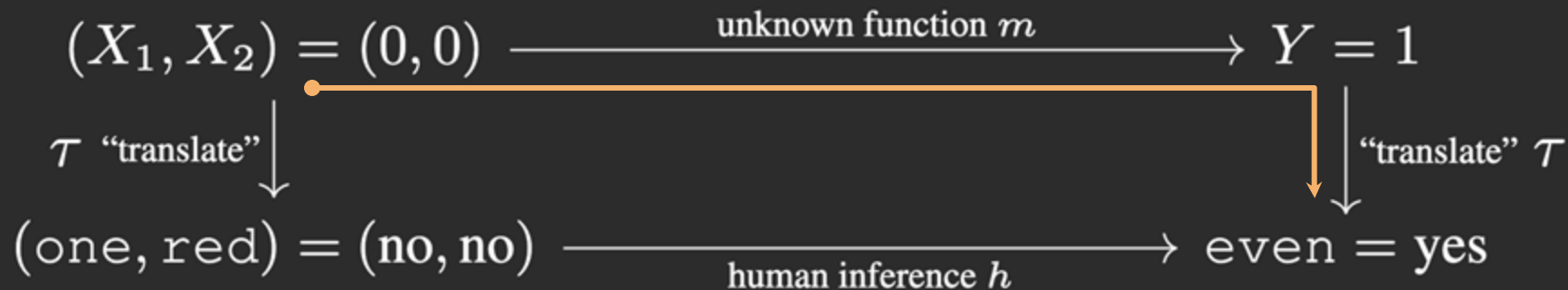
Is the model  $P(Y \mid X; m)$  interpretable?

$\omega$	$X_1$	$X_2$	$Y = m(X)$
0	0	1	1
0	0	0	1
1	1	0	0



Is the model  $P(Y \mid X; m)$  interpretable?

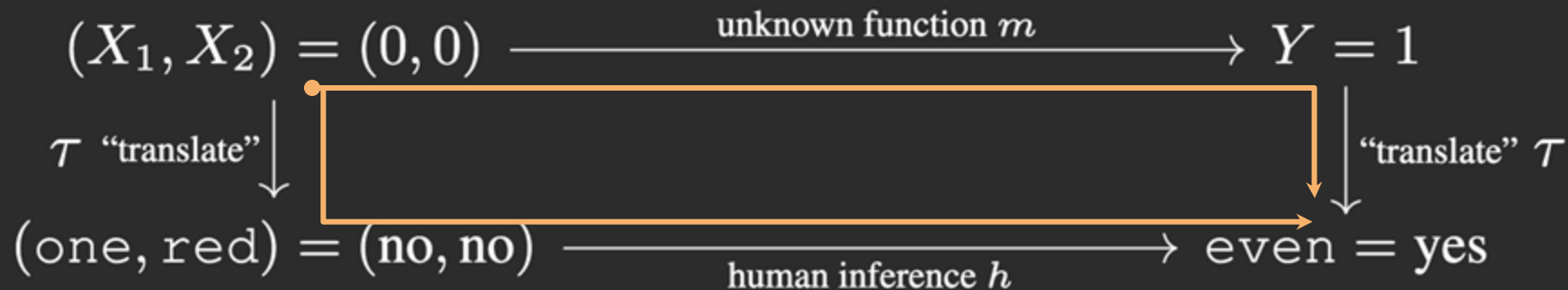
$\omega$	$X_1$	$X_2$	$Y = m(X)$
0	0	1	1
0	0	0	1
1	1	0	0





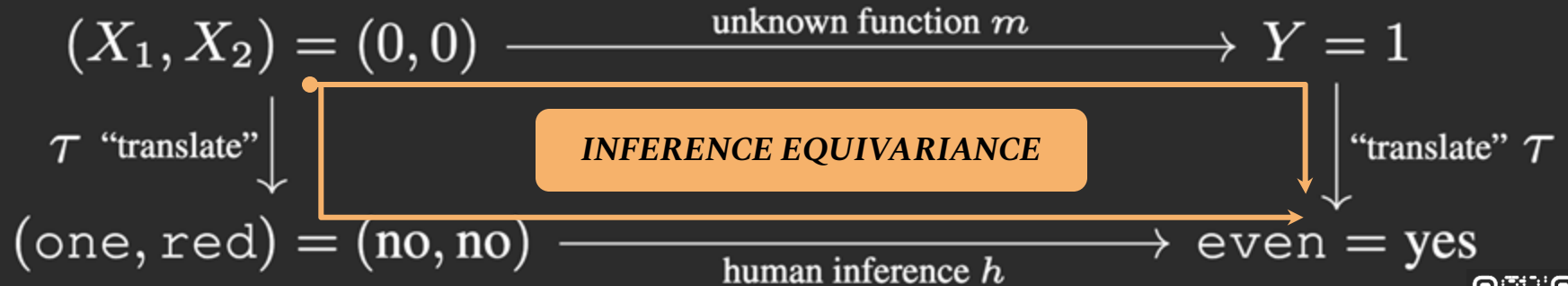
Is the model  $P(Y \mid X; m)$  interpretable?

$\omega$	$X_1$	$X_2$	$Y = m(X)$
0	0	1	1
0	0	0	1
1	1	0	0



Is the model  $P(Y \mid X; m)$  interpretable?

$\omega$	$X_1$	$X_2$	$Y = m(X)$
0	0	1	1
0	0	0	1
1	1	0	0



# Consequences:

## 1. Any function is interpretable

- But not all functions are easy to interpret by all users!

## 2. Interpretability is a **spectrum**

- Diagram may commute only for a subset of samples

## 3. Naively verifying interpretability via inference equivariance is **intractable**

- We need to scan a table with  $\mathcal{O}(\exp(D))$  entries
- With 10x10 pixel images we need more checks than #atoms in observable universe!

## 4. Many translations exist, but some are “not sound”

<i>sound</i>		<i>not sound</i>
$X_1(\text{blue circle}) = 0 \xrightarrow{\tau} \text{unum} = no$	$X_1(\text{blue circle}) = 0 \xrightarrow{\tau} \text{baz} = no$	$X_1(\text{blue circle}) = 0 \xrightarrow{\tau} \text{one} = yes$
$X_1(\text{blue square}) = 1 \xrightarrow{\tau} \text{unum} = yes$	$X_1(\text{blue square}) = 1 \xrightarrow{\tau} \text{baz} = yes$	$X_1(\text{blue square}) = 1 \xrightarrow{\tau} \text{one} = yes$

# Outline

I. What is interpretability?

## II. Foundations

1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models

## III. Instantiations

1. Instantiating the concept encoder  $P(C \mid X)$
2. Instantiating the task predictor  $P(Y \mid C)$

## IV. Open questions

## **Problem #2**

*Naively verifying inference equivariance is intractable.*

## **Research Question #2**

*Can we identify assumptions & principles making inference equiv. tractable?*

*Can we make interpretability tractable by compression?*

$$X \subseteq \mathbb{R}^D$$



64x64 pixels

$$Y \subseteq \mathbb{N}$$

even

*Can we make interpretability tractable by compression?*

$$X \subseteq \mathbb{R}^D$$



64x64 pixels

$$C \subseteq \mathbb{R}^K$$

$$K \ll D$$

compression

$$I(Y; C) \approx I(Y; X)$$

preservation of information

$$Y \subseteq \mathbb{N}$$

even

*Can we make interpretability tractable by compression?*

$$X \subseteq \mathbb{R}^D$$



64x64 pixels

$$C \subseteq \mathbb{R}^K$$

$$K \ll D$$

compression

$$I(Y; C) \approx I(Y; X)$$

preservation of information

$$Y \subseteq \mathbb{N}$$

even

$$I(Y; X_j \mid \{C_i\}_{i=1}^K) = 0 \quad \forall X_j \notin \{C_i\}_{i=1}^K$$

conditional interpretability principle

[1] [Cayton "Algorithms for manifold learning" Univ. of California at San Diego Tech \(2005\).](#)

[2] [Pearl "Probabilistic reasoning in intelligent systems: networks of plausible inference" Elsevier \(1988\).](#)



*Can we make interpretability tractable by compression?*

$$X \subseteq \mathbb{R}^D$$



64x64 pixels

$$C \subseteq \mathbb{R}^K$$

$$Y \subseteq \mathbb{N}$$

$$K \ll D$$

compression

$$I(Y; C) \approx I(Y; X)$$

preservation of information

even

$$I(Y; X_j \mid \{C_i\}_{i=1}^K) = 0 \quad \forall X_j \notin \{C_i\}_{i=1}^K$$

conditional interpretability principle



$$P(Y, C, X) = P(Y \mid C)P(C \mid X)$$

[1] [Cayton "Algorithms for manifold learning" Univ. of California at San Diego Tech \(2005\).](#)

[2] [Pearl "Probabilistic reasoning in intelligent systems: networks of plausible inference" Elsevier \(1988\).](#)

[3] [Koh et al. "Concept bottleneck models" ICML \(2020\).](#)

*Can we make interpretability tractable by compression?*

$$X \subseteq \mathbb{R}^D$$



64x64 pixels

$$C \subseteq \mathbb{R}^K$$

$$Y \subseteq \mathbb{N}$$

$$K \ll D$$

compression

$$I(Y; C) \approx I(Y; X)$$

preservation of information

even

$$I(Y; X_j \mid \{C_i\}_{i=1}^K) = 0 \quad \forall X_j \notin \{C_i\}_{i=1}^K$$

conditional interpretability principle



$$P(Y, C, X) = P(Y \mid C)P(C \mid X)$$



Complexity (entries in the table) go from  $\mathcal{O}(\exp(D))$  to  $\mathcal{O}(\exp(K))$

[1] [Cayton "Algorithms for manifold learning" Univ. of California at San Diego Tech \(2005\).](#)

[2] [Pearl "Probabilistic reasoning in intelligent systems: networks of plausible inference" Elsevier \(1988\).](#)

[3] [Koh et al. "Concept bottleneck models" ICML \(2020\).](#)

### ***Problem #3***

*Translations may not be sound.*

### ***Research Question #3***

*Can we characterize **sound** translations?*

### ***Problem #3***

*Translations may not be sound.*






### ***Research Question #3***

*Can we characterize **sound** translations?*

**A: Yes! We can use “concepts”!**

*Research Question #3.1: What is a “concept”?*

# What is a concept?

	red	one $\wedge$ $\neg$ fruit	zero	even
	1	1	0	0
	0	0	1	1
	1	0	1	1
	0	1	0	0
	1	0	0	0

Note that:






$$\beta(\{\text{red}\}) = \{\text{apple}, \text{red exclamation mark}, \text{red circle}\}$$

[1] Goguen "What is a concept?" International Conference on Conceptual Structures (2005).

[2] Ganter et al "Formal concept analysis" Springer (1999).



# What is a concept?

	red	one $\wedge$ $\neg$ fruit	zero	even
	1	1	0	0
	0	0	1	1
	1	0	1	1
	0	1	0	0
	1	0	0	0

Note that:

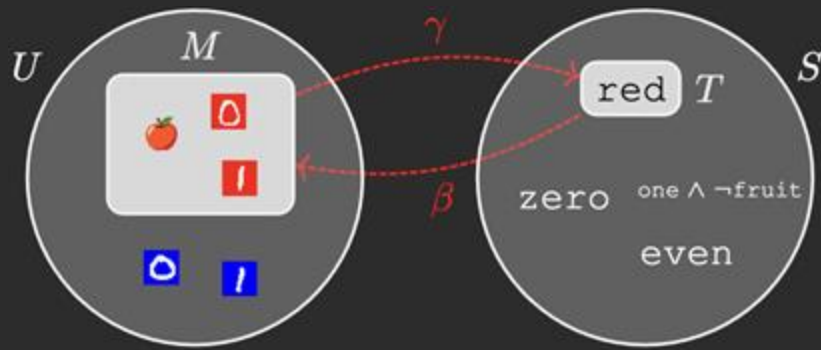
$$\beta(\{\text{red}\}) = \{\text{apple}, \text{red vertical bar}, \text{red circle}\} \quad \gamma(\{\text{apple}, \text{red vertical bar}, \text{red circle}\}) = \{\text{red}\}$$

[1] Goguen "What is a concept?" International Conference on Conceptual Structures (2005).

[2] Ganter et al "Formal concept analysis" Springer (1999).



# What is a concept?



*Concept* — Set of examples and sentences satisfying “**closure**” condition:

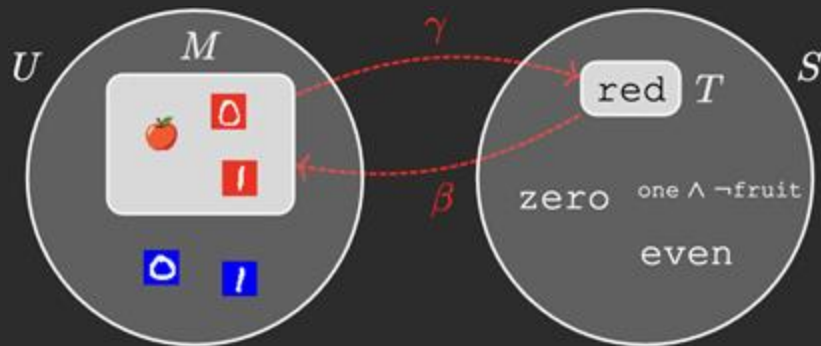
$$\beta(\{\text{red}\}) = \{\text{apple}, \text{red square}, \text{red square}\} \quad \gamma(\{\text{apple}, \text{red square}, \text{red square}\}) = \{\text{red}\}$$

[1] Goguen "What is a concept?" International Conference on Conceptual Structures (2005).

[2] Ganter et al "Formal concept analysis" Springer (1999).



*What is a concept?*



*Probabilistic interpretation:*

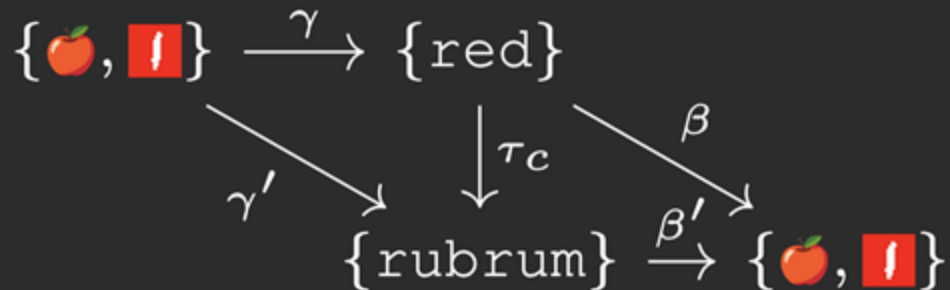
Concept membership:  $C_{\text{red}} = \mathbb{I}_{X(\omega) \in M_{\text{red}}} = \begin{cases} 1, & \text{if } X(\omega) = \text{red} \in M_{\text{red}} = \{\text{apple}, \text{red}, \text{blue}\} \\ 0, & \text{otherwise.} \end{cases}$

Concept CPD:  $P(C_{\text{red}} = 1 \mid X = \text{red}) = g_{\text{red}}(\text{red}) \quad g_i : X \rightarrow [0, 1]$



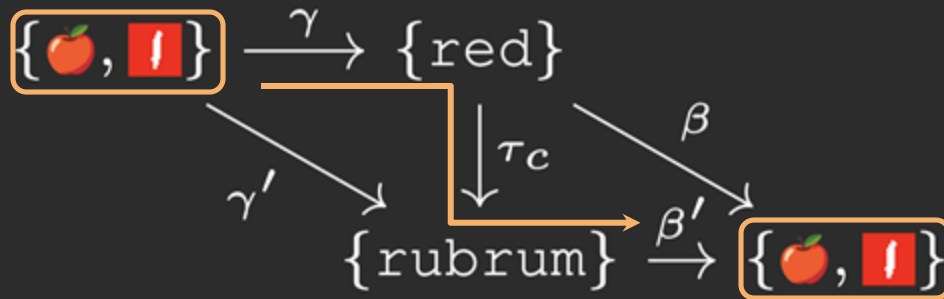
*When is a translation “sound”?*

**Sound** translations  
preserve closure:



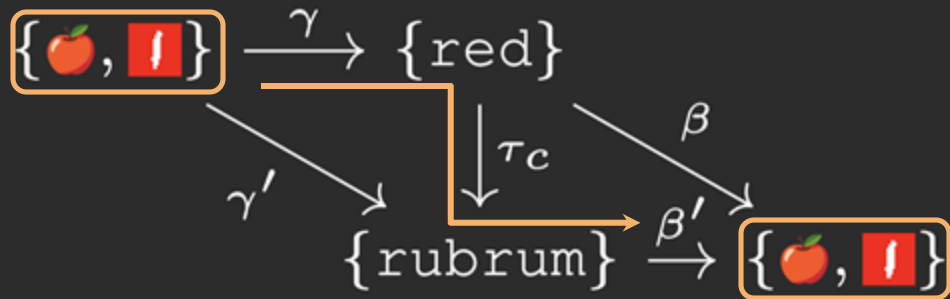
*When is a translation “sound”?*

**Sound** translations  
preserve closure:

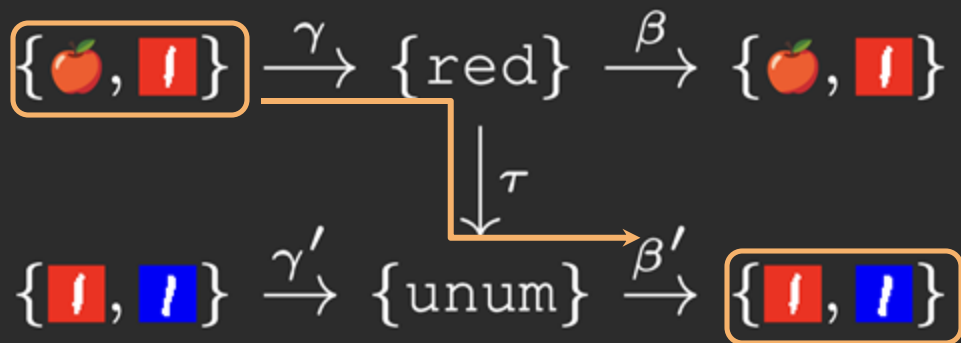


When is a translation “sound”?

**Sound** translations preserve closure:



**“Unsound”** translations do not preserve closure:



### ***Problem #4***

*Multiple translations may exist.*

### ***Research Question #4***

*How do we choose among a set of sound translations?*

*What to do when multiple **sound** translations exist?*

$$\begin{array}{lll} X_1(\text{0}) = 0 \xrightarrow{\tau} \text{unum} = \text{no} & X_1(\text{0}) = 0 \xrightarrow{\tau} \text{baz} = \text{no} & X_1(\text{0}) = 0 \xrightarrow{\tau} \text{zero} = \text{no} \\ X_1(\text{1}) = 1 \xrightarrow{\tau} \text{unum} = \text{yes} & X_1(\text{1}) = 1 \xrightarrow{\tau} \text{baz} = \text{yes} & X_1(\text{1}) = 1 \xrightarrow{\tau} \text{zero} = \text{yes} \end{array}$$

What to do when multiple **sound** translations exist?

$$\begin{array}{lll} X_1(\text{0}) = 0 \xrightarrow{\tau} \text{unum} = \text{no} & X_1(\text{0}) = 0 \xrightarrow{\tau} \text{baz} = \text{no} & X_1(\text{0}) = 0 \xrightarrow{\tau} \text{zero} = \text{no} \\ X_1(\text{1}) = 1 \xrightarrow{\tau} \text{unum} = \text{yes} & X_1(\text{1}) = 1 \xrightarrow{\tau} \text{baz} = \text{yes} & X_1(\text{1}) = 1 \xrightarrow{\tau} \text{zero} = \text{yes} \end{array}$$



Reasoning shortcut — “**Aligned**” translation not identifiable



We need an “alignment” prior  $P(\tau)$

[1] Melsa “System identification” Academic Press (1971).

[2] Geirhos et al. “Shortcut learning in deep neural networks” Nature Machine Intelligence (2020).

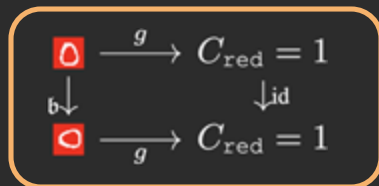
[3] Koh et al. “Concept bottleneck models” ICML (2020).

[4] Cunningham et al. “Sparse autoencoders find highly interpretable features in language models.” arXiv preprint (2023).

# Design considerations

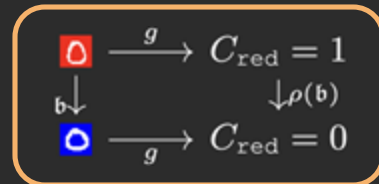
$$P(C \mid X)$$

Concept invariances



$$K \ll D$$

Concept equivariances



$$I(Y; C) \approx I(Y; X)$$

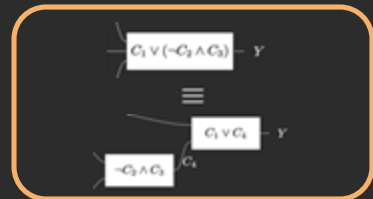
$$P(Y \mid C)$$



Compositionality

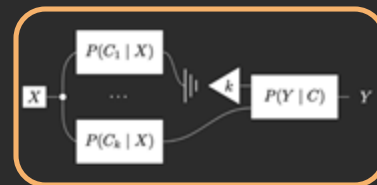


Sparsity



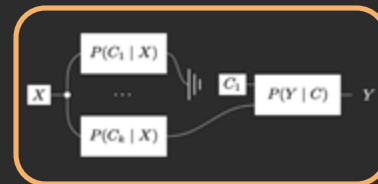
## Human-model interaction

Do-interventions



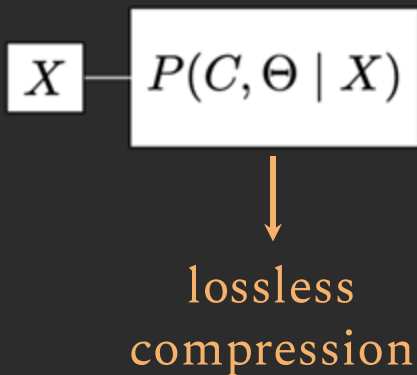
Estimate causal effects

Ground-truth interventions



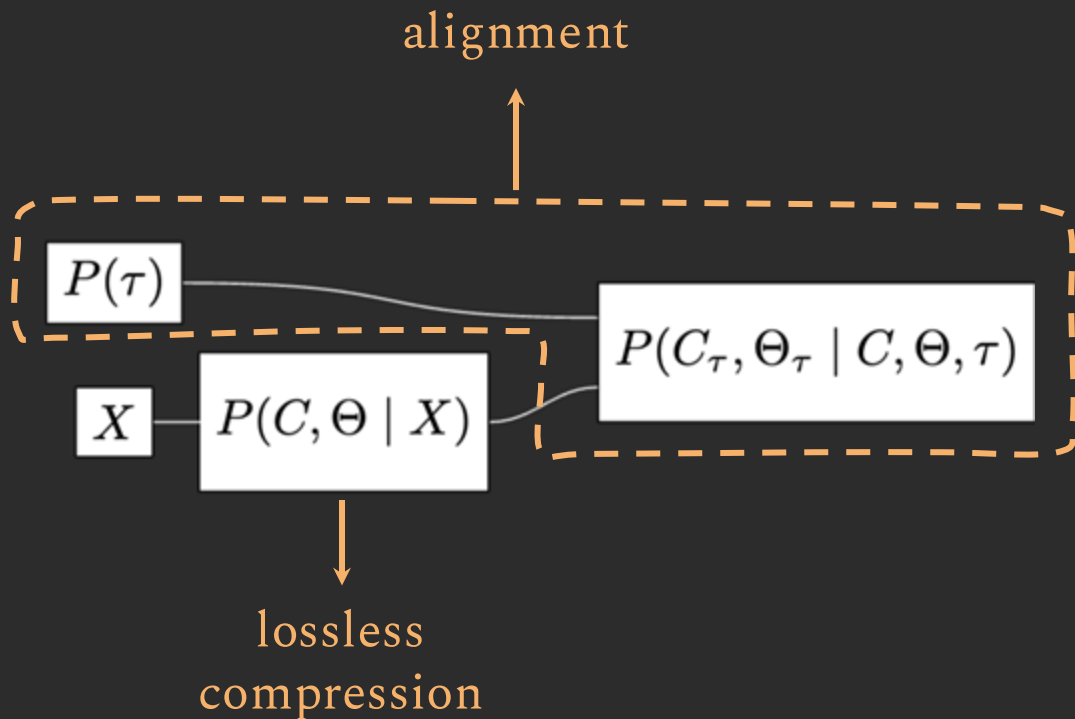
Improve model predictions

# Blueprint for interpretable models

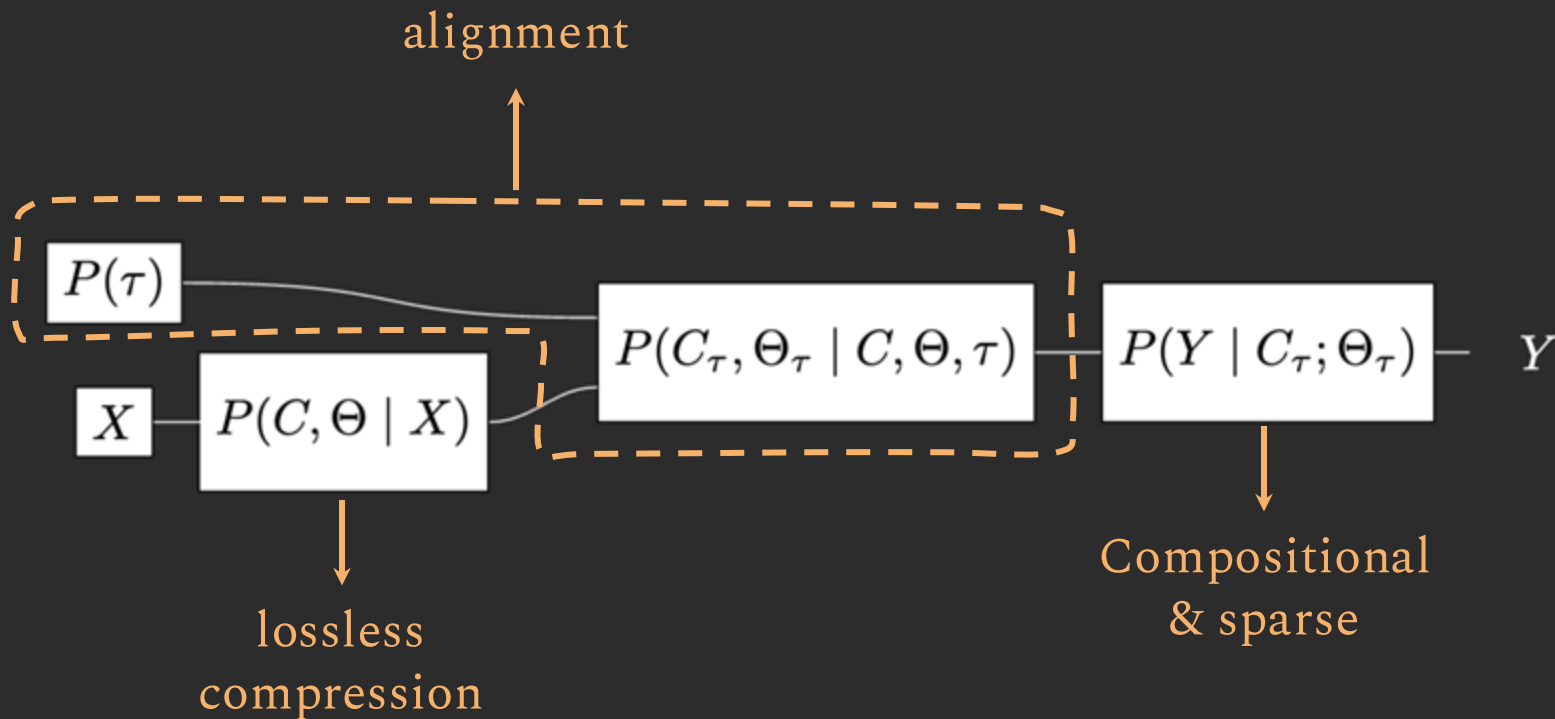




# Blueprint for interpretable models



# Blueprint for interpretable models



# Outline

I. What is interpretability?

II. Foundations

1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models

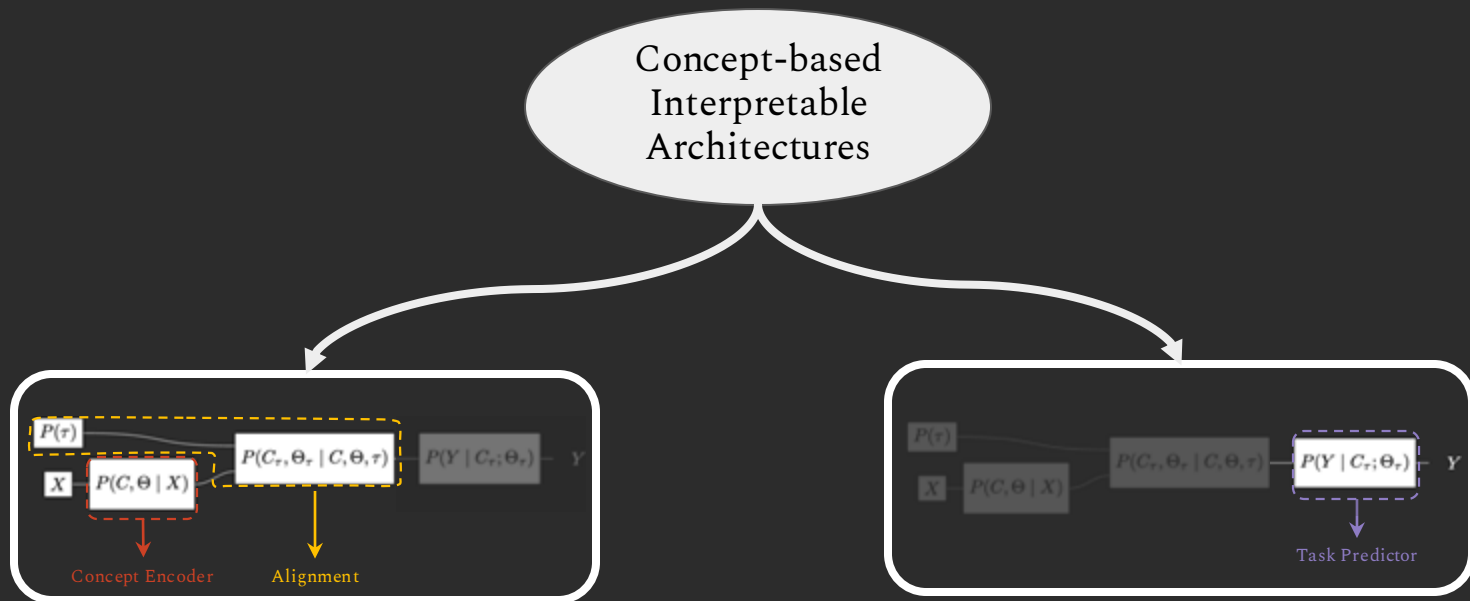
III. Instantiations

1. Instantiating the Concept Encoder  $P(C \mid X)$
2. Instantiating the Task Predictor  $P(Y \mid C)$

IV. Open questions

# A high-level taxonomy of concept-based interpretability

We look at methods based on the **main components in the blueprint they modify**:



How are concepts **represented** and **aligned**?

How are tasks **predicted**?

# Outline

I. What is interpretability?

II. Foundations

1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models

III. Instantiations

**1. Instantiating the concept encoder  $P(C | X)$**

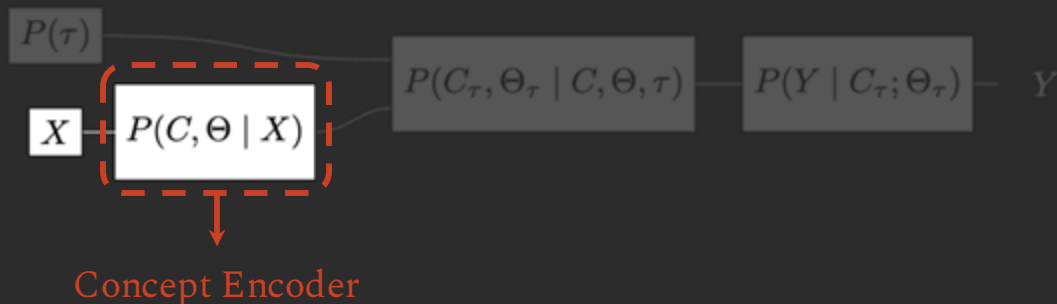
2. Instantiating the task predictor  $P(Y | C)$

IV. Open questions

# Instantiating the concept encoder $P(C | X)$

For now, we will assume:

1. The task predictor  $P(Y | C; \Theta)$  will be parameterised as a **(sparse) linear layer**.
2. The **parameters  $\Theta$  are deterministic and learnable** variables
  1. That is: we assume  $P(C, \Theta | X) = P(C | X) P(\Theta | X)$  with  $P(\Theta | X)$  being a delta distribution.
3. We will parameterise  $P(C | X)$  as a **deep neural network  $g(x; \theta_g)$**



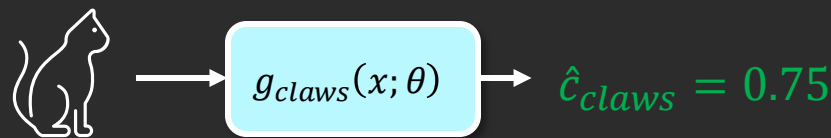
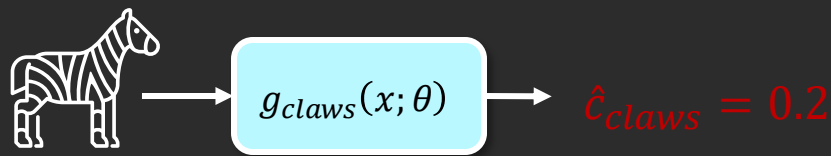
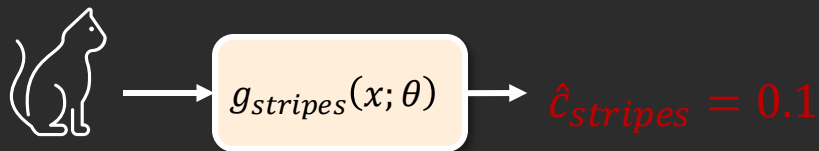
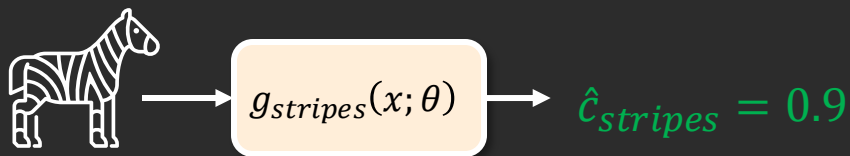
**This allows us to focus on how  $P(C | X)$  represents concepts**

# First steps: bounded scalar concept representations

The simplest concept encoder  $g(x)$  represents concepts as **bounded scalars**:

$$x \longrightarrow g_{\text{stripes}}(x; \theta_i) \longrightarrow \hat{c}_i \propto P(C_i = 1 \mid x)$$

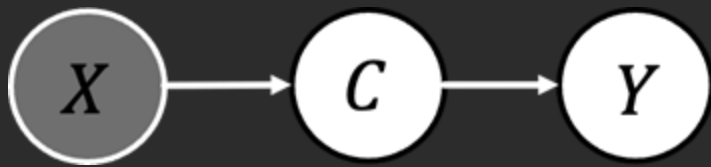
For example, we may use sigmoidal “probes” for each concept of interest:



# First steps: bounded scalar concept representations

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

1. We have a set of **binary concepts labels**  $C$  for each training sample.
2. **Each concept** in  $C$  can be **independently predicted** from  $X$ .
3. The **downstream task**  $Y$  can be **perfectly predicted from**  $C$  alone.



$$P(C, Y | X) = P(C | X)P(Y | C)$$

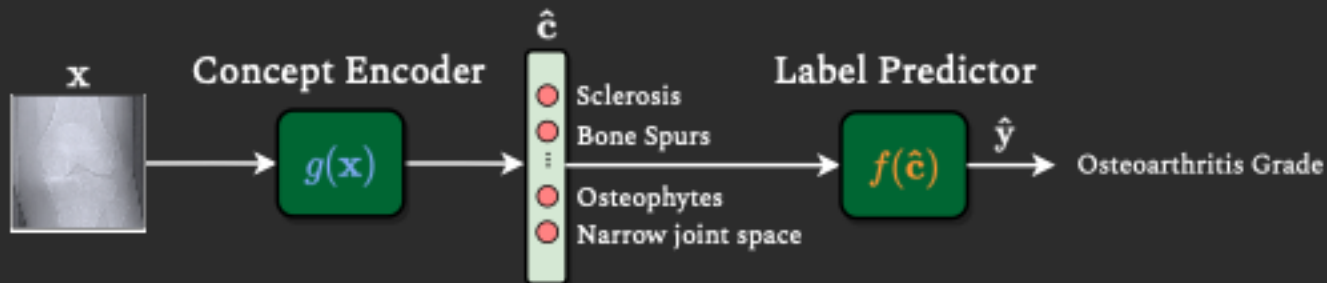




# First steps: bounded scalar concept representations

CBMs exploit scalar bounded concept representations through two components:

1. A **concept encoder**  $g(\mathbf{x})$  that predicts binary concepts  $C$  from the features  $X$
2. A linear **label predictor**  $f(\mathbf{c})$  that predicts the task  $Y$  from the concepts  $C$



# How to train your CBM: vanilla alignment

Given a dataset  $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(j)})\}_{j=1}^N$  we can **train a CBM** in three different ways:



# How to train your CBM: vanilla alignment

Given a dataset  $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(j)})\}_{j=1}^N$  we can **train a CBM** in three different ways:

1. **Independently**: train  $g(\mathbf{x})$  and  $f(\mathbf{c})$  separately.

**Align** each component's output to its corresponding ground-truth labels using a **cross-entropy loss** (CE)

$$\left\{ \begin{array}{l} \mathbb{E}_{(\mathbf{x}, \mathbf{c}, y) \sim \mathcal{D}} [\text{BCE}(g(\mathbf{x}), \mathbf{c})] \\ \mathbb{E}_{(\mathbf{x}, \mathbf{c}, y) \sim \mathcal{D}} [\text{CE}(f(\mathbf{c}), y)] \end{array} \right.$$



# How to train your CBM: vanilla alignment

Given a dataset  $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(j)})\}_{j=1}^N$  we can **train a CBM** in three different ways:

1. **Independently**: train  $g(\mathbf{x})$  and  $f(\mathbf{c})$  separately.
2. **Sequentially**: train  $g(\mathbf{x})$  and then train  $f(\mathbf{c})$  from the concept encoder's outputs.

**Train the concept encoder**

$$\mathbb{E}_{(\mathbf{x}, \mathbf{c}, y) \sim \mathcal{D}} [\text{BCE}(g(\mathbf{x}), \mathbf{c})] \xrightarrow{\text{Freeze } g} \mathbb{E}_{(\mathbf{x}, \mathbf{c}, y) \sim \mathcal{D}} [\text{CE}(f(g(\mathbf{x})), y)]$$

**Train the task predictor**



# How to train your CBM: vanilla alignment

Given a dataset  $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{c}^{(j)}, y^{(j)})\}_{j=1}^N$  we can **train a CBM** in three different ways:

1. **Independently**: train  $g(\mathbf{x})$  and  $f(\mathbf{c})$  separately.
2. **Sequentially**: train  $g(\mathbf{x})$  and then train  $f(\mathbf{c})$  from the concept encoder's outputs.
3. **Jointly**: train  $g(\mathbf{x})$  and  $f(\mathbf{c})$  at the same time through a weighted loss.

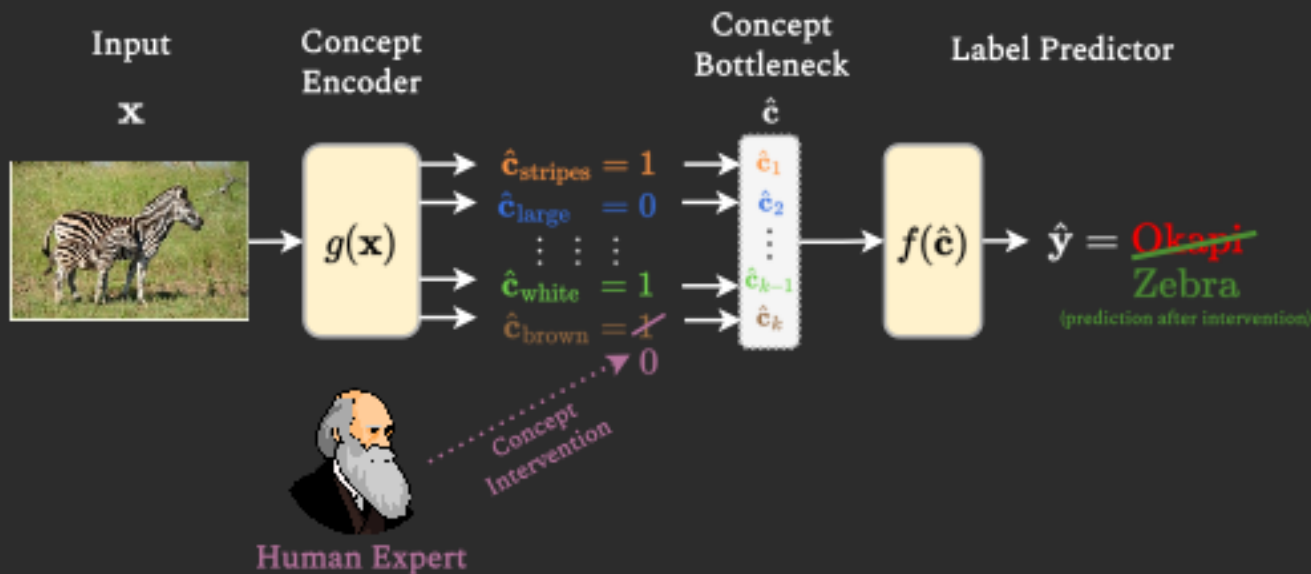
$$\mathbb{E}_{(\mathbf{x}, \mathbf{c}, y) \sim \mathcal{D}} [\text{CE}(f(g(\mathbf{x})), y) + \lambda \cdot \text{BCE}(g(\mathbf{x}), \mathbf{c})]$$

Controls how much we value **concept alignment** vs **task alignment**



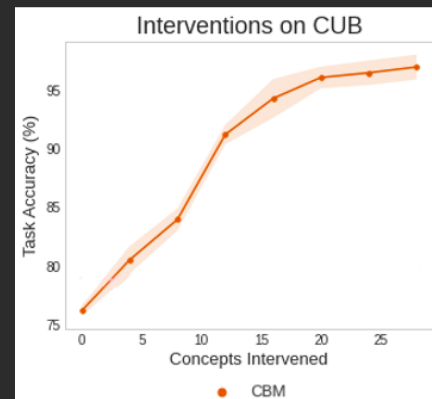
# Test-time concept interventions

Notice that this simple architecture already supports some of the powerful human-AI interactions we previously mentioned



# Test-time concept interventions

This leads to models that can improve their performance when deployed in conjunction with a domain expert



[1] [Shin et al. "A Closer Look at the Intervention Procedure of Concept Bottleneck Models." ICML 2023.](#)

[2] [Chauhan et al. "Interactive concept bottleneck models." AAAI \(2023\).](#)



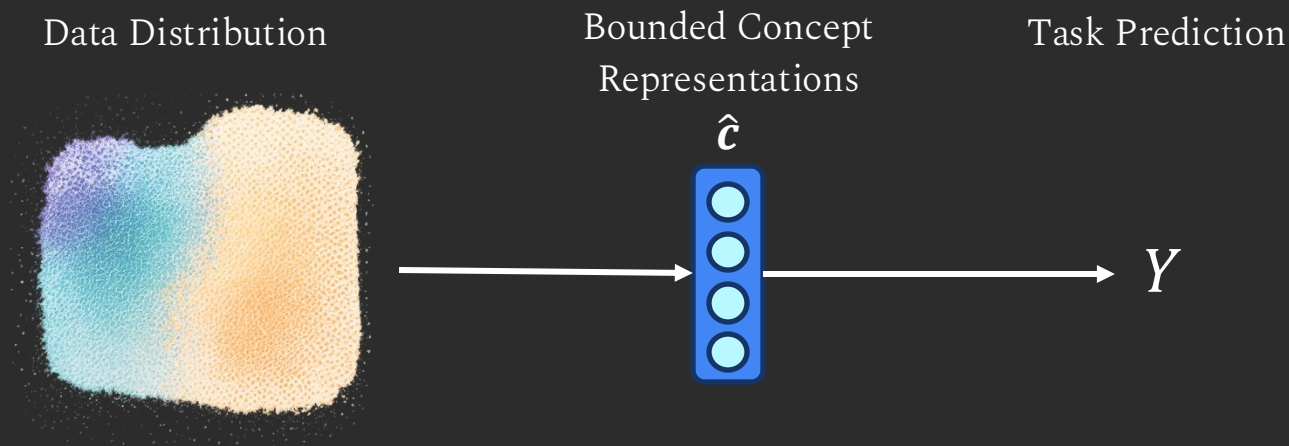
ICML 23



AAAI 23

# Diversity in concept representations

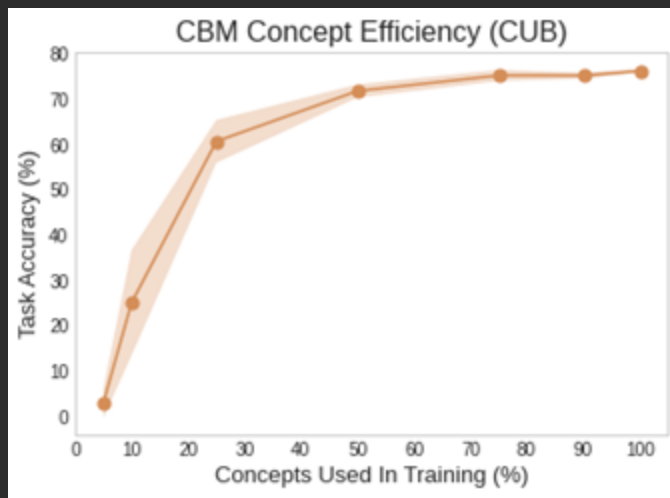
CBMs are a theoretically useful framework, but they are highly constrained in practice because of the **information bottleneck** their bounded scalar encodings impose





# Diversity in concept representations

CBMs are a theoretically useful framework, but they are highly constrained in practice because of the **information bottleneck** their bounded scalar encodings impose

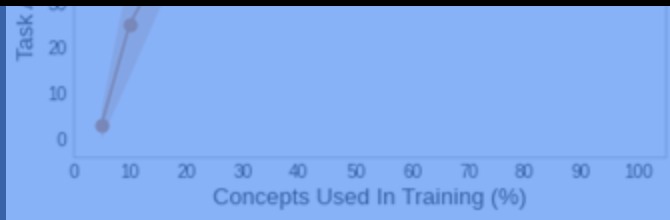


A CBM's task accuracy significantly decreases when its concept set becomes **incomplete**

# Diversity in concept representations

CBMs are a theoretically useful framework, but they are highly constrained in practice because of the **information bottleneck** their bounded scalar encodings impose

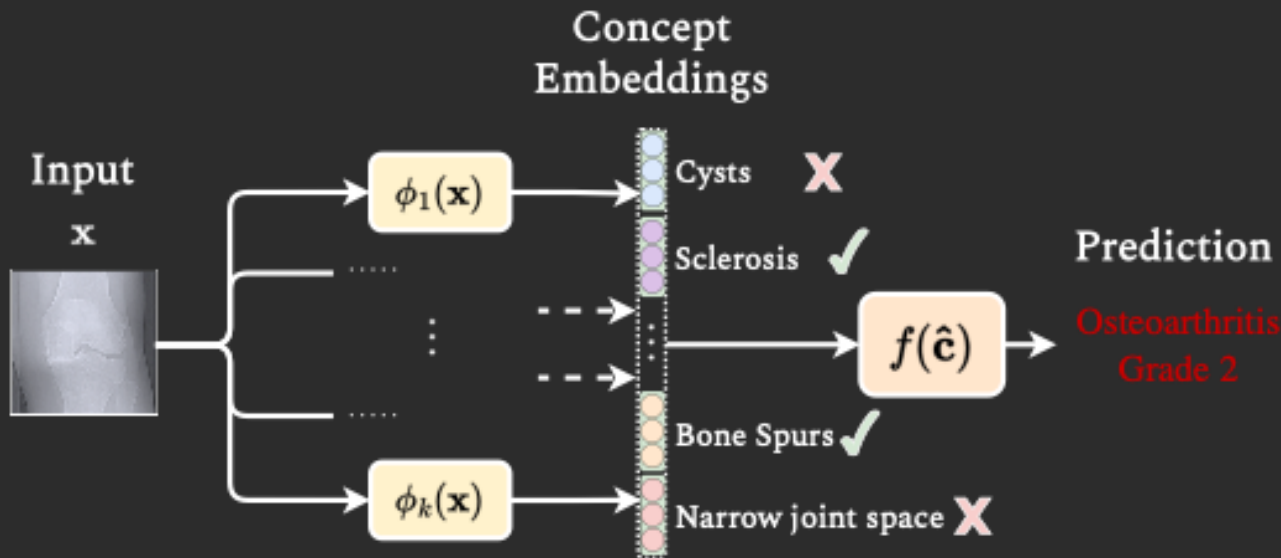
How can we design concept representations that can operate in incompleteness?



A CBM's task accuracy significantly decreases when its concept set becomes **incomplete**

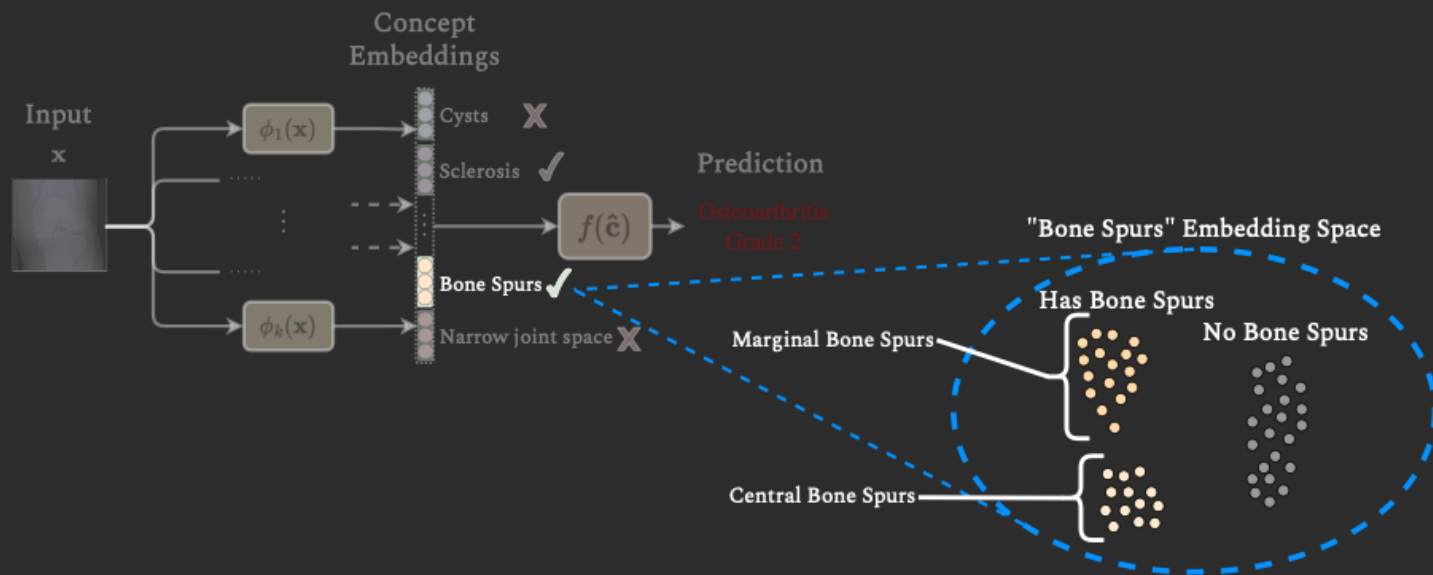
# Dynamic concept embeddings

We can circumvent this by learning **dynamic embedding representations** that represent a concept's activation using a high-dimensional vector



# Dynamic concept embeddings

We can circumvent this by learning **dynamic embedding representations** that represent a concept's activation using a high-dimensional vector



# Dynamic concept embeddings

For example, **Concept Embedding Models (CEMs)** decompose the concept representation  $\hat{\mathbf{c}}_i$  as the mixture of two embedding representations  $\{\hat{\mathbf{c}}_i^+(\mathbf{x}), \hat{\mathbf{c}}_i^-(\mathbf{x})\}$ :

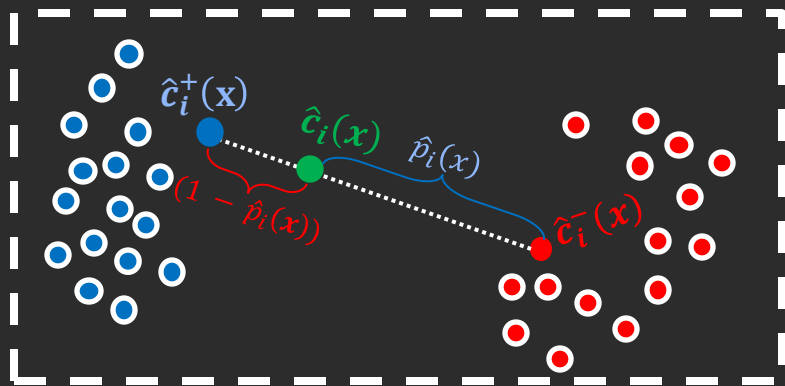
$$\hat{\mathbf{c}}_i(\mathbf{x}) = \hat{p}_i(\mathbf{x}) \hat{\mathbf{c}}_i^+(\mathbf{x}) + (1 - \hat{p}_i(\mathbf{x})) \hat{\mathbf{c}}_i^-(\mathbf{x})$$



“Positive” concept embeddings



“Negative” concept embeddings



Where the probability  $\hat{p}_i(\mathbf{x}) = P(c_i = 1 | \mathbf{x})$  of concept  $c_i$  being present in  $\mathbf{x}$  is given by a simple (linear) function  $s(\hat{\mathbf{c}}_i^+(\mathbf{x}), \hat{\mathbf{c}}_i^-(\mathbf{x}))$  of the positive and negative embeddings aligned via cross-entropy.



# Dynamic concept embeddings

More recent works have adapted concept embeddings to:

1. Enable better-calibrated **uncertainty** estimates (**Probabilistic CBMs** [1])
2. Compute arbitrary concept **conditional distributions** (**Energy-based CBMs** [2])
3. Learn “ask” for **effective concept interventions** (**Intervention-aware CEMs** [3])
4. Better **generalize** within out-of-distribution shifts (**MixCEMs** [4])



[1] [Kim et al. "Probabilistic Concept Bottleneck Models." ICML \(2023\).](#)

[2] [Xu et al. "Energy-based concept bottleneck models: Unifying prediction, concept intervention, and probabilistic interpretations." ICLR \(2024\).](#)

[3] [Espinosa Zarlenga et al. "Learning to receive help: Intervention-aware concept embedding models." NeurIPS \(2023\).](#)

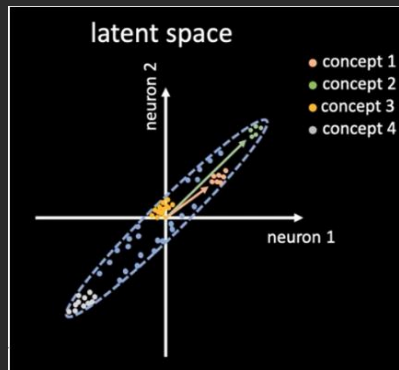
[4] [Espinosa Zarlenga et al. "Avoiding Leakage Poisoning: Concept Interventions Under Distribution Shifts." ICML \(2025\).](#)

# (Large) unbounded scalar concept representations

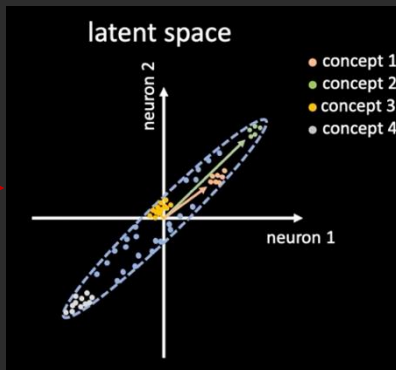
Recent approaches exploit unbounded scalar representations of large concept sets:

1. **Concept Whitening** rotates the output of a Batch Normalization layer so that the resulting activation scores are **axis-aligned with known concepts**

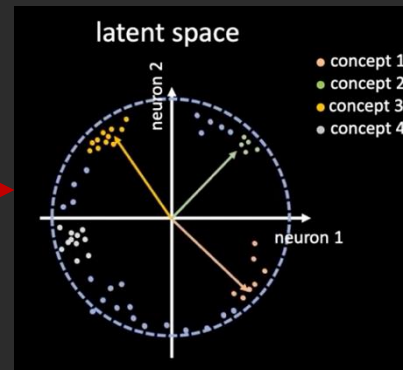
Normalize



Whiten



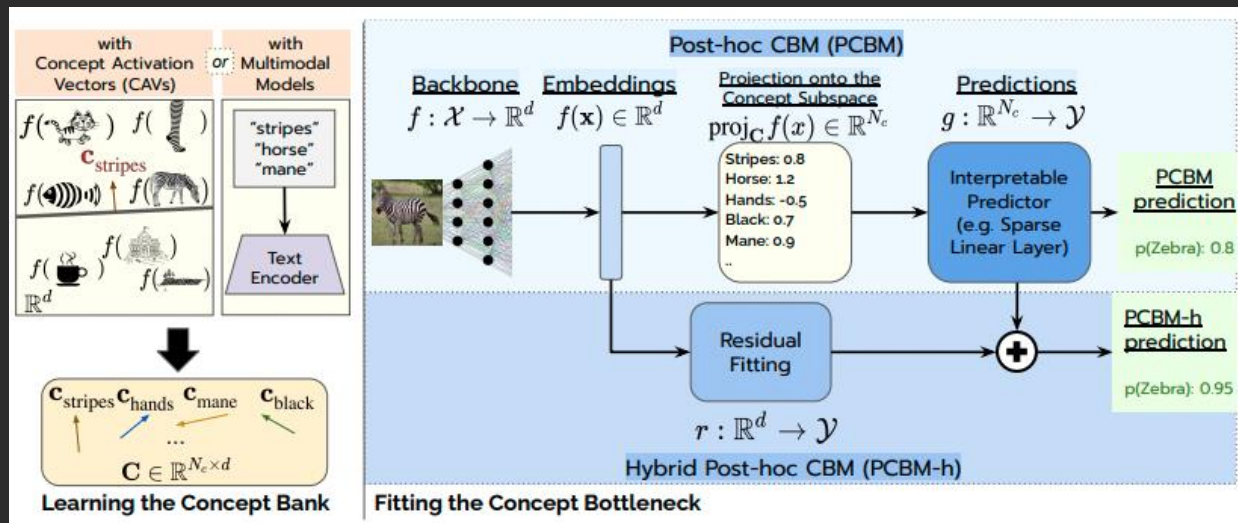
Rotate



# (Large) unbounded scalar concept representations

Recent approaches exploit unbounded scalar representations of large concept sets:

2. **Post-hoc CBMs (PCBMs)** project a pre-trained DNN's latent space into a concept score space using **linear concept probes**





# (Large) unbounded scalar concept representations

This can be turned into a fully-unsupervised pipeline if we exploit large language models as **knowledge bases** and multimodal embeddings (e.g., CLIP) as **annotators**:



*"List the most important features for recognizing something as a {class}:"*

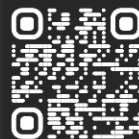
These are called Label-free CBMs [1] or Language-guided Bottlenecks (LaBos) [2]

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

[2] [Yang et al. "Language in a bottle: Language model guided concept bottlenecks for interpretable image classification." CVPR \(2023\).](#)



ICLR 23

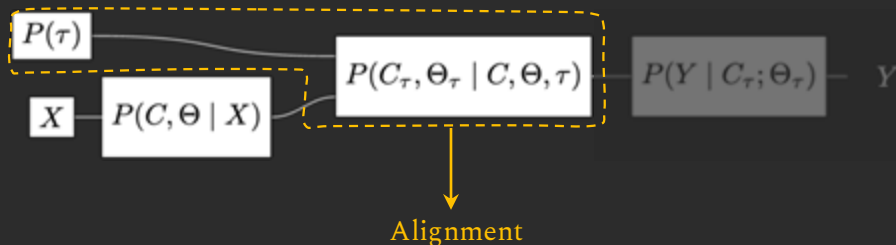


CVPR 23

# What about other forms of alignment?

So far, we have assumed that we align concept representations and human-understandable concepts by:

1. Minimizing a **cross-entropy** term, or
2. Projecting samples into a **direction** correlated with each concept (e.g., via CLIP)



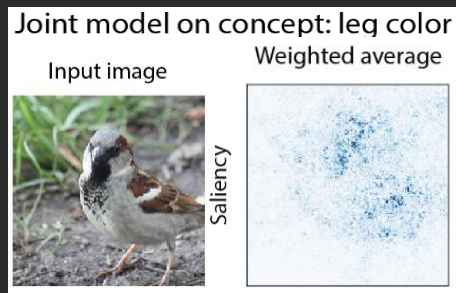
[1] Margelou et al. "Do concept bottleneck models learn as intended?" ICLR Workshop on Responsible AI (2021).

[2] Mahinpei et al. "Promises and pitfalls of black-box concept learning models." ICML Workshop on Theoretic Foundation, Criticism, and Application of XAI (2021).

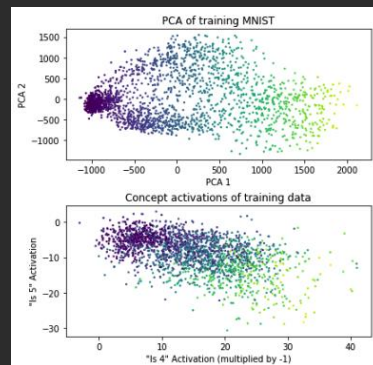
[3] Raman et al. "Do Concept Bottleneck Models Respect Localities?" TMLR (2025).

# What about other forms of alignment?

These alignment mechanisms can lead to unwanted representational **leakage**



Saliency maps seem to suggest  
concepts do not properly attend  
the right features  
(Margeloiu et al., 2021) [1]



CBMs may have incentives to encode  
the entire data representation in the  
concepts' soft predictions  
(Mahinpei et al., 2021) [2]

[1] Margeloiu et al. "Do concept bottleneck models learn as intended?" ICLR Workshop on Responsible AI (2021).

[2] Mahinpei et al. "Promises and pitfalls of black-box concept learning models." ICML Workshop on Theoretic Foundation, Criticism, and Application of XAI (2021).

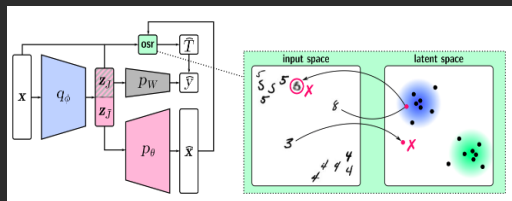
[3] Raman et al. "Do Concept Bottleneck Models Respect Localities?" TMLR (2025).

# Re-thinking traditional concept alignment

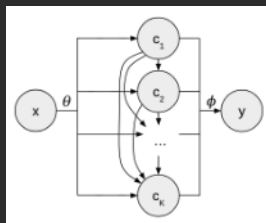
Nevertheless, several works have been proposed to **mitigate leakage** by:

1. Framing alignment as a disentanglement learning **task** (e.g., **GlanceNets** [1])
2. Modeling concept relationships (e.g., **Autoregressive CBMs** [2] or **Stochastic CBMs** [3])

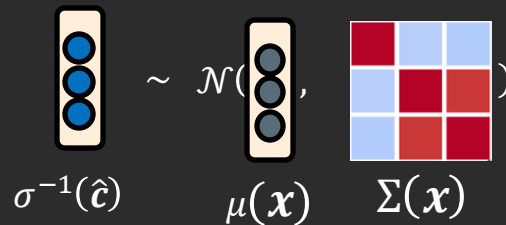
GlanceNets



Autoregressive CBMs



Stochastic CBMs



[1] [Marconato et al. "Glancenets: Interpretable, leak-proof concept-based models." NeurIPS \(2022\).](#)

[2] [Havasi et al. "Addressing leakage in concept bottleneck models." NeurIPS \(2022\).](#)

[3] [Vandenhirtz, Laguna et al. "Stochastic Concept Bottleneck Models." NeurIPS \(2024\).](#)



GlanceNets



Autoregressive



Stochastic

# Outline

I. What is interpretability?

II. Foundations

1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models

III. Instantiations

1. Instantiating the concept encoder  $P(C | X)$

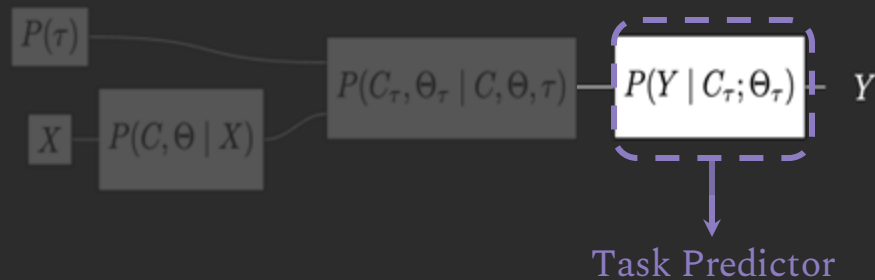
**2. Instantiating the task predictor  $P(Y | C)$**

IV. Open questions

# Instantiating the task predictor $P(Y | C; \Theta)$

So far, we have assumed that the label predictor  $P(Y | C; \Theta)$  is a linear layer.

Linear models, however, are not very expressive and **may struggle to learn complex non-linear interactions** between concepts and labels.



We will now consider different ways of parameterising  $P(\Theta | X)$  and  $P(Y | C; \Theta)$

# Powerful interpretable task predictors

A simple way to extend the power of linear task predictors is to make them act linearly at least in the **concept neighbourhood** of each sample  $\mathbf{x}$

Traditional Linear Task Predictor:  $f(\hat{\mathbf{c}}; \boldsymbol{\theta}) = \boldsymbol{\theta}^T \hat{\mathbf{c}}$

“Linear-ish” Task Predictor:  $f(\hat{\mathbf{c}}; \boldsymbol{\theta}(\mathbf{x})) = \boldsymbol{\theta}(\mathbf{x})^T \hat{\mathbf{c}}$

Where:

1. The distribution  $P(\boldsymbol{\theta} | X) = \boldsymbol{\theta}(\mathbf{x})$  is parameterised as a learnable DNN, and
2. The function  $\boldsymbol{\theta}(\mathbf{x})$  is encouraged to be “stable” for samples with similar concepts



# Neuro-Symbolic task predictors

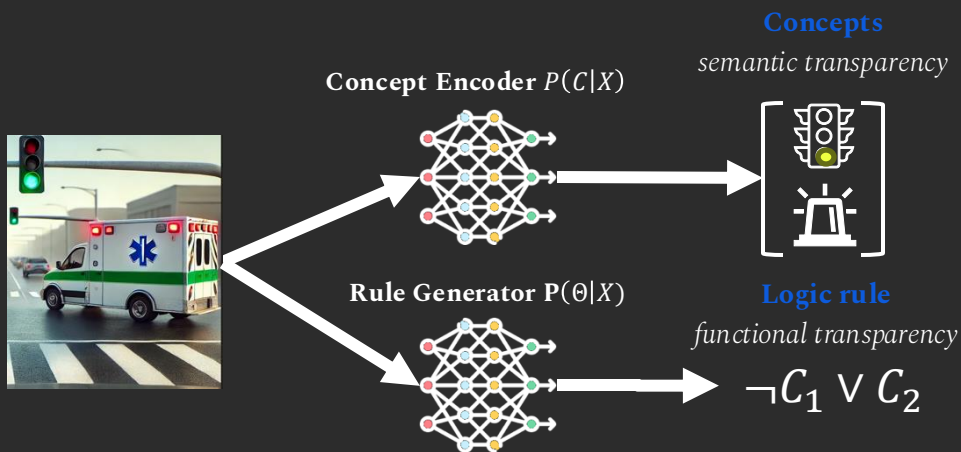
Another way is to exploit different parameter spaces  $P(\Theta | X)$  such as symbolic rules



# Neuro-Symbolic task predictors

Another way is to exploit different parameter spaces  $P(\Theta | X)$  such as symbolic rules

**Step 1 (neural generation):** Two DNNs generate both concept activations & rule parameters



[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.



ICML 23



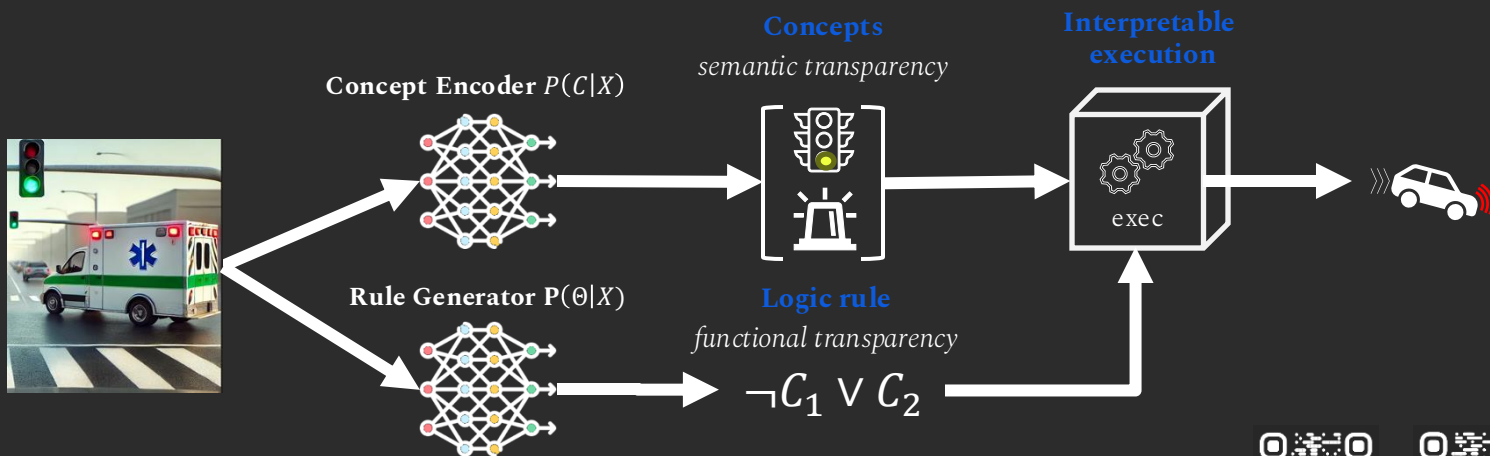
NeurIPS 24

# Neuro-Symbolic task predictors

Another way is to exploit different parameter spaces  $P(\Theta | X)$  such as symbolic rules

**Step 1 (neural generation):** Two DNNs generate both concept activations & rule parameters

**Step 2 (interpretable execution):** 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.



ICML 23

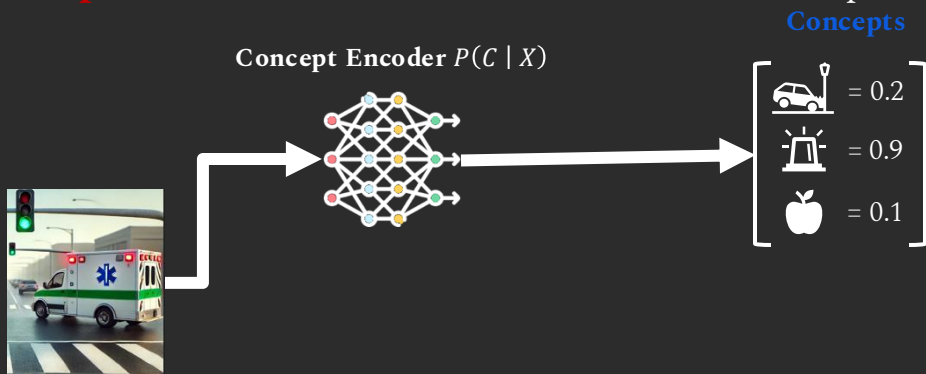


NeurIPS 24

# Neuro-Symbolic task predictors

An example is a **Concept-based Memory Reasoning** (CMR) where we:

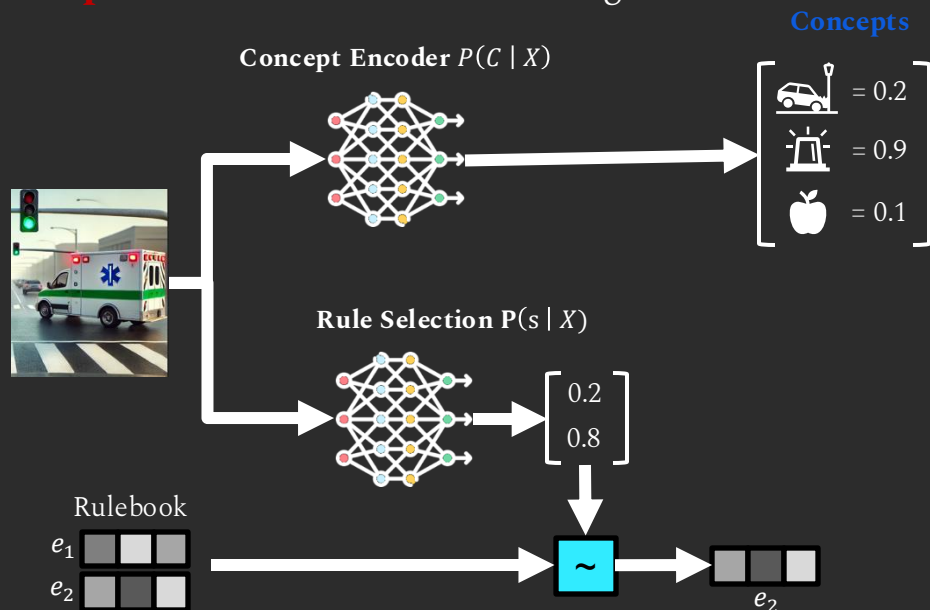
**Step 1:** Predict a set of bounded scalar concept representations  $P(C | X)$  with a DNN



# Neuro-Symbolic task predictors

An example is a **Concept-based Memory Reasoning** (CMR) where we:

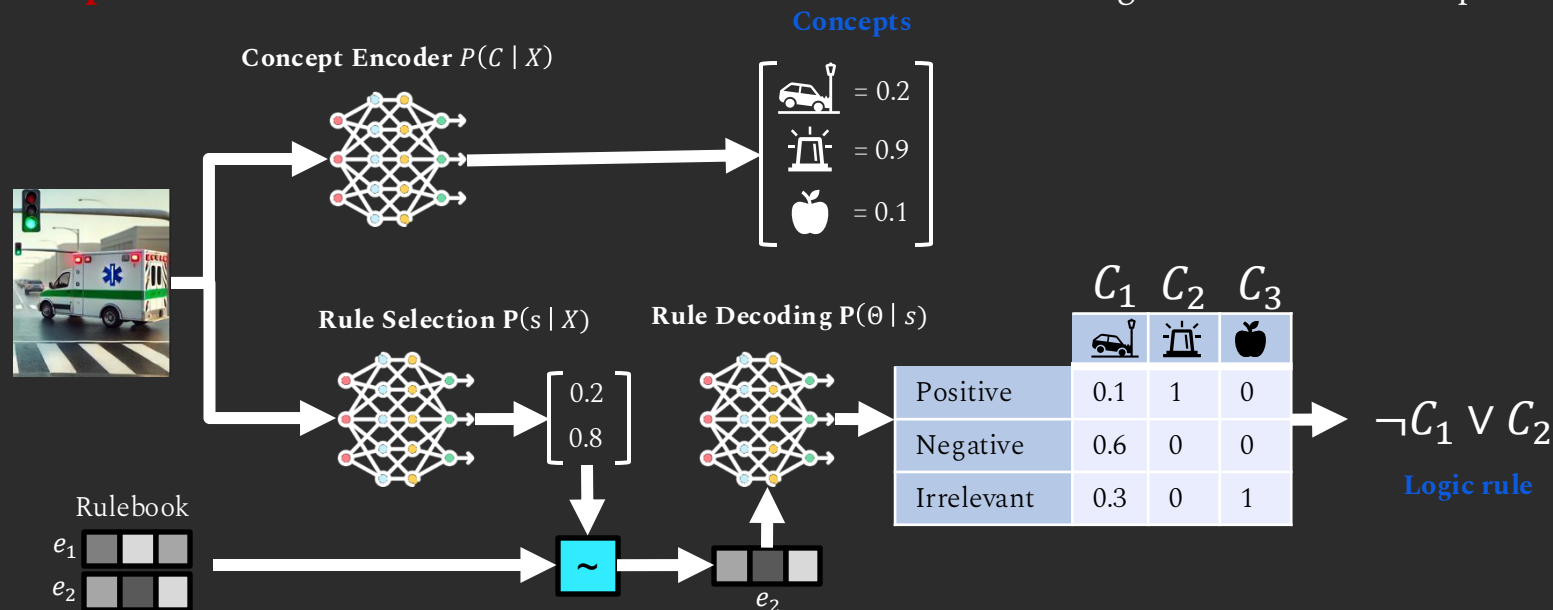
**Step 2:** Select a rule “embedding” from a learnable “rulebook” using a DNN



# Neuro-Symbolic task predictors

An example is a **Concept-based Memory Reasoning** (CMR) where we:

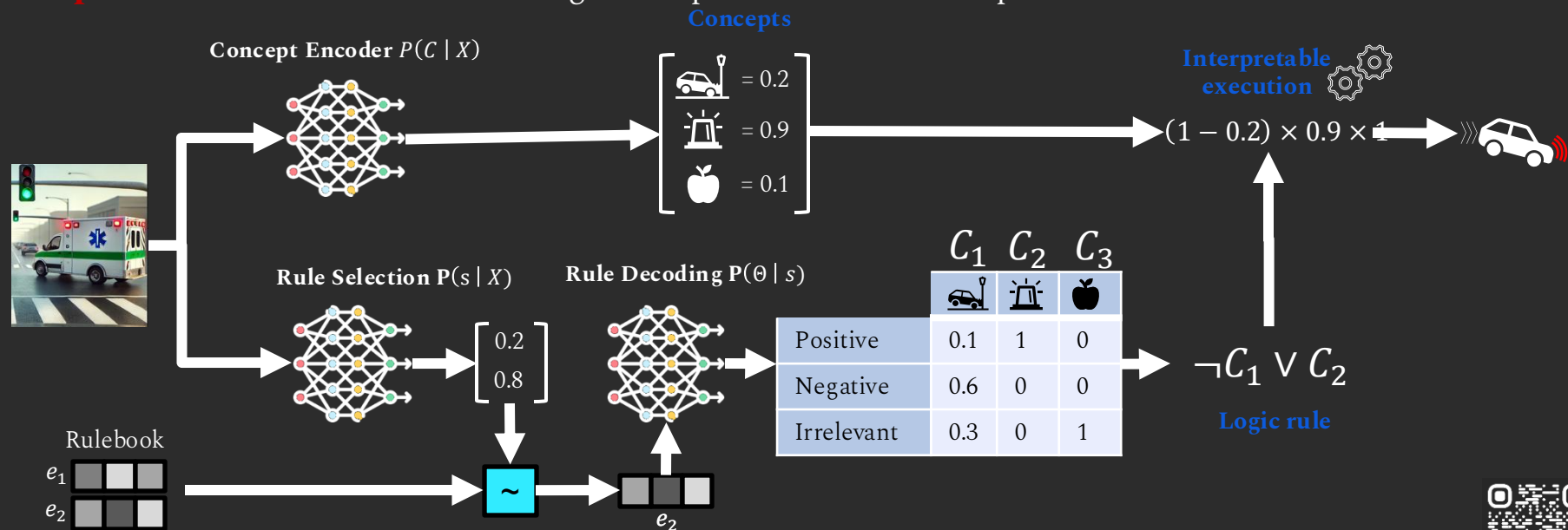
**Step 3:** Use a learnable model to decode the rule embedding into three states per concept



# Neuro-Symbolic task predictors

An example is a **Concept-based Memory Reasoning** (CMR) where we:

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



# Neuro-Symbolic task predictors

These steps enable CMR to:

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



# Neuro-Symbolic task predictors

Other neuro-symbolic task predictors include:

Logic Explained Networks (LENs)

---



DeepProbLog

---



Deep Concept Reasoner (DCR)

---



Neural Algorithmic Reasoning (NAR)

---



- [1] [Ciravegna et al. "Logic explained networks." Artificial Intelligence \(2023\)](#)
- [2] [Manhaeve et al. "Deepproblog: Neural probabilistic logic programming." NeurIPS \(2018\).](#)
- [3] [Barbiero et al. "Interpretable neural-symbolic concept reasoning." ICML \(2023\).](#)
- [4] [Veličković et al. "Neural algorithmic reasoning." Patterns \(2021\).](#)



# Outline

I. What is interpretability?

II. Foundations

1. Assumptions, data structures, and design principles for interpretability
2. Blueprint for interpretable models

III. Instantiations

1. Instantiating the concept encoder  $P(C; X)$
2. Instantiating the task predictor  $P(Y; C)$

IV. Open questions

# Challenges and open questions

There are still a lot of significant challenges and open questions in concept-based interpretability, particularly on making these models **more “real-world-friendly”**.



# Challenges and open questions: Label-free approaches

**Label-free models** open the door for **practical interpretable models**, yet they are currently not as reliable as supervised concept-based models

1. How can label-free models be used **without domain-specific foundation models**?



# Challenges and open questions: Label-free approaches

**Label-free models** open the door for **practical interpretable models**, yet they are currently not as reliable as supervised concept-based models

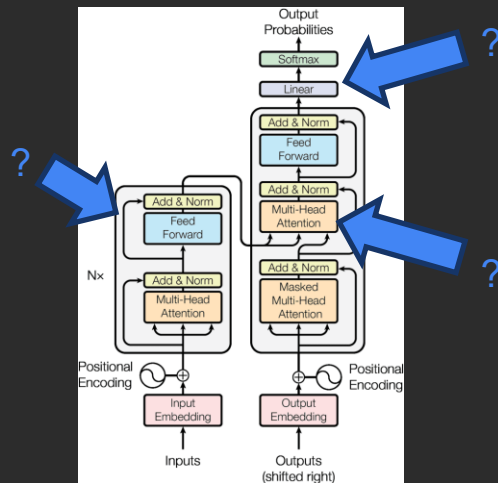
2. How can we effectively **intervene** in label-free settings?



# Challenges and open questions: Scalability

Concept-based interpretable models have yet to be properly integrated to **large scale and multimodal models**. This raises a lot of important questions:

1. **Where should we look for/place** concepts in **LLMs**? Should this be done at a token, sentence, paragraph, or text-level?

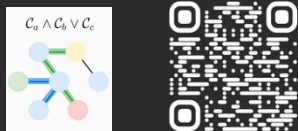


# Challenges and open questions: Scalability

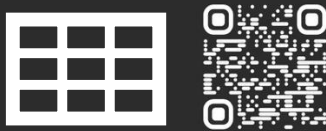
Concept-based interpretable models have yet to be properly integrated to **large scale and multimodal models**. This raises a lot of important questions:

2. What does a **"concept" really mean** in textual inputs? What about in other modalities such as genomics, finance, and law?

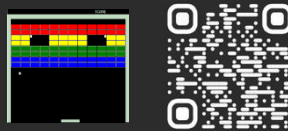
Graph Data  
(e.g., Xuanyuan et al.)



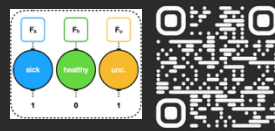
Tabular Data  
(e.g., Espinosa Zarlenga et al.)



RL Tasks  
(e.g., Ye et al.)



Time Series Data  
(e.g., Kazhdan et al.)



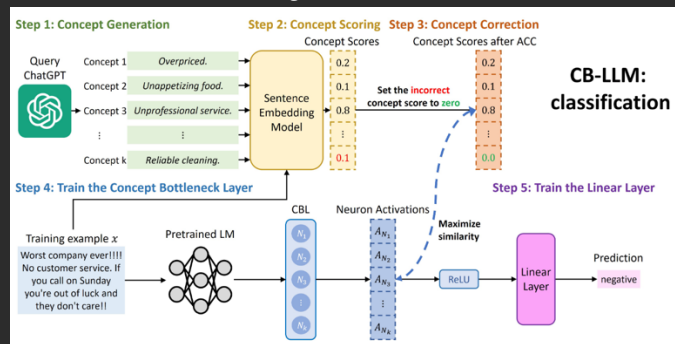
- [1] Xuanyuan et al. "Global concept-based interpretability for graph neural networks via neuron analysis." AAAI (2023).
- [2] Espinosa Zarlenga et al. "Tabcbm: Concept-based interpretable neural networks for tabular data." TMLR (2024).
- [3] Ye et al. "Concept-based interpretable reinforcement learning with limited to no human labels." ICML (2024).
- [4] Kazhdan et al. "MEME: generating RNN model explanations via model extraction." arXiv (2020).

# Challenges and open questions: Scalability

Concept-based interpretable models have yet to be properly integrated to **large scale and multimodal models**. This raises a lot of important questions:

3. Can we even **scale** concept-based architectures to current large model standards?

Concept Bottleneck LLMs  
(e.g., Sun et al.)



# Conclusion: Concepts and their role in interpretability

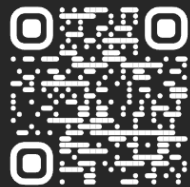
Today we:

1. Argued that interpretability can be thought as a form of “**inference equivariance**”
2. Showed that **verifying this equivariance is intractable** unless we rely on compressing our input space into a set of “**concepts**”
3. Proposed a **blueprint** for designing interpretable models formed by (1) a **concept encoder**, (2) an **alignment mechanism**, and (3) a **label predictor**
4. Discussed several **instantiations** of this blueprint across the literature

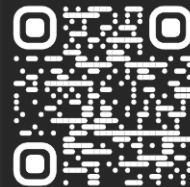


## Further resources: Website and library

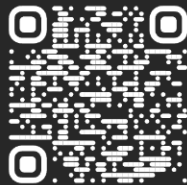
1. Website of a longer iteration of this tutorial (extended bibliography and slides):



[conceptlearning.github.io](https://conceptlearning.github.io)



2. PyTorch Concepts (PyC), tutorials and library for concept-based interpretability:



[pyc-team.github.io/pyc-book/intro.html](https://pyc-team.github.io/pyc-book/intro.html)



**Thank you, grazie, gracias for your time!**

**Contact:** [pietro.barbiero@ibm.com](mailto:pietro.barbiero@ibm.com) and [me466@cam.ac.uk](mailto:me466@cam.ac.uk)