

Creating Neuro-Symbolic Models for Boosting Interpretability and Reasoning Capabilities of AI

Dr. Parth Padalkar

Assistant Professor of Computer Science

Texas State University

Email: padalkar@txstate.edu

Website: <https://petrichor1998.github.io/>

Introduction

- Deep learning (Neural AI) is omnipresent in today's world; CNNs, ViTs, LLMs etc.
- Increasingly being used for sensitive applications
 - Self-Driving Cars
 - Medical Diagnosis
- However, these are “black-box” models, very difficult to get an explanation
- Symbolic AI has been present for a long time (1950s); logic programming, decision trees, automated theorem proving, SAT solvers etc.
- Rules are explainable

Neurosymbolic AI

- Neural = “Better pattern matching but poor explainability”
- Symbolic = “Better explainability and reasoning but lacks generalizability”
- Neurosymbolic (NeSy) AI aims to combine the advantages of Neural AI and Symbolic AI
- Examples:
 - Google Deepmind’s Alpha Go and Alpha Zero
 - ChatGPT5 (*loosely*)



Research Overview

Creating Interpretable NeSy models for Image Classification

Creating “Fast and Slow” models inspired by System 1 and System 2 ideology of human cognition for better reasoning and reliability



Creating Interpretable NeSy models for Image Classification

Background: Default Logic

- **Default Logic** can express rules that are true in majority of cases but not always
- e.g., “birds normally fly” OR “by default birds fly, but there are exceptional ones that do not”

Default Rules:

- $\text{flies}(X) \leftarrow \text{bird}(X) \wedge \text{not } \text{ab0}(X).$
- $\text{ab0}(X) \leftarrow \text{penguin}(X) \wedge \text{not } \text{ab1}(X).$
- $\text{ab1}(X) \leftarrow \text{super_penguin}(X).$

Background: Default Logic

- Default Rules can be realized through a logic programming language called ASP
- ASP = Answer Set Programming; Great for emulating **commonsense reasoning**
- e.g., “birds normally fly” OR “by default birds fly, but there are exceptional ones that do not”

ASP rules:

- `flies(X) :- bird(X), not ab0(X).`
- `ab0(X) :- penguin(X), not ab1(X).`
- `ab1(X) :- super_penguin(X).`

Background: FOLD-SE-M

- **FOLD-SE-M** is an algorithm that generates ASP rules from *tabular* data
- Higher accuracy than decision tree and smaller rule-set, hence more interpretable

Titanic Survival Dataset:

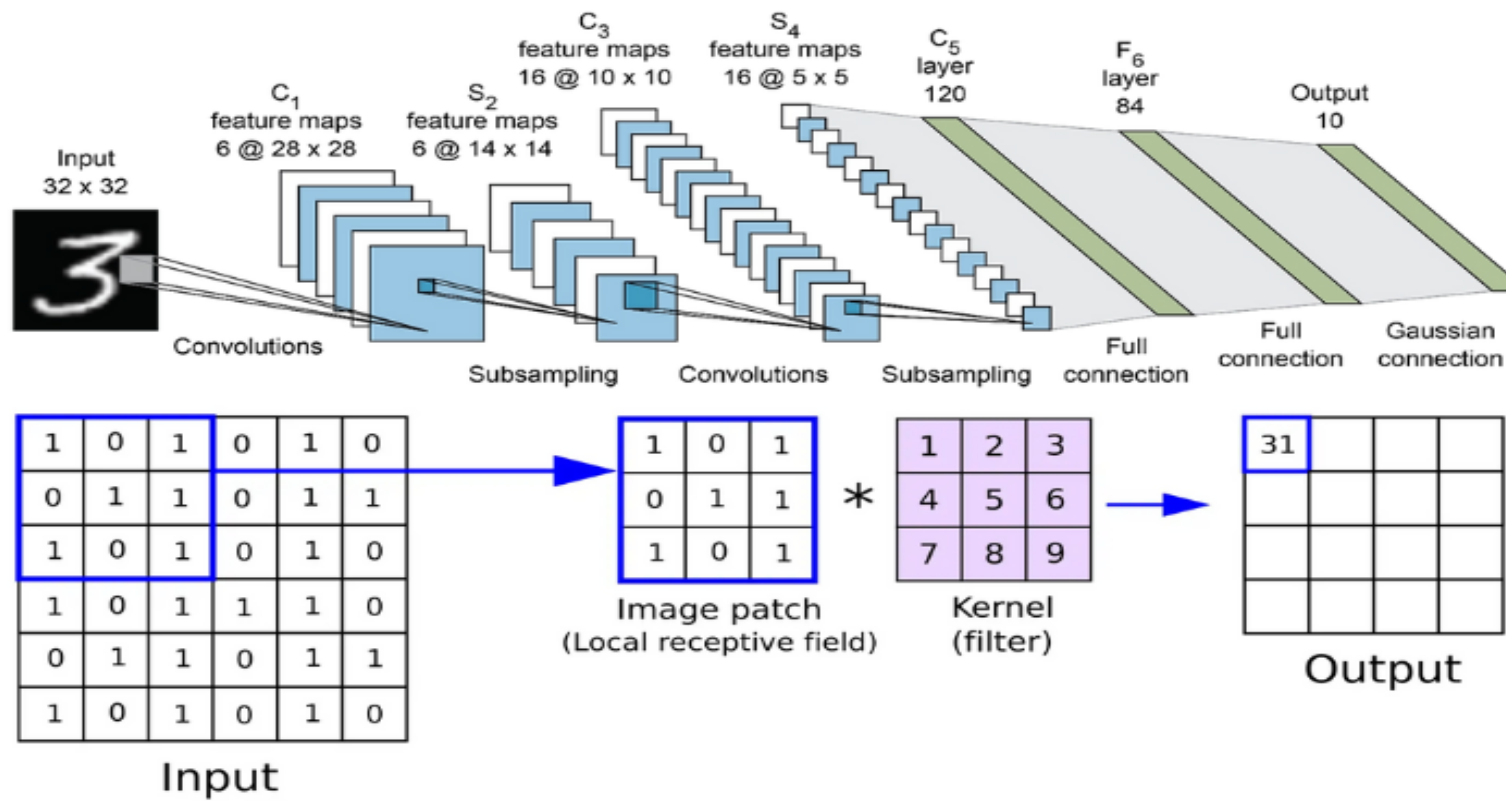
- Dimensions of Training Data: 891 x 8; Dimensions of Test Data: 418 x 8
- ASP generated:

`status(X, perished) :- sex(X, 'male').`

`status(X, perished) :- class(X, '3'), sex(X, 'female'), fare(X, N1),
not(N1=<23.25).`

- Accuracy: 0.99; Precision: 0.98; Recall: 1.0; F1-score: 0.99

Background: CNN



Source: <https://www.superannotate.com/blog/guide-to-convolutional-neural-networks>

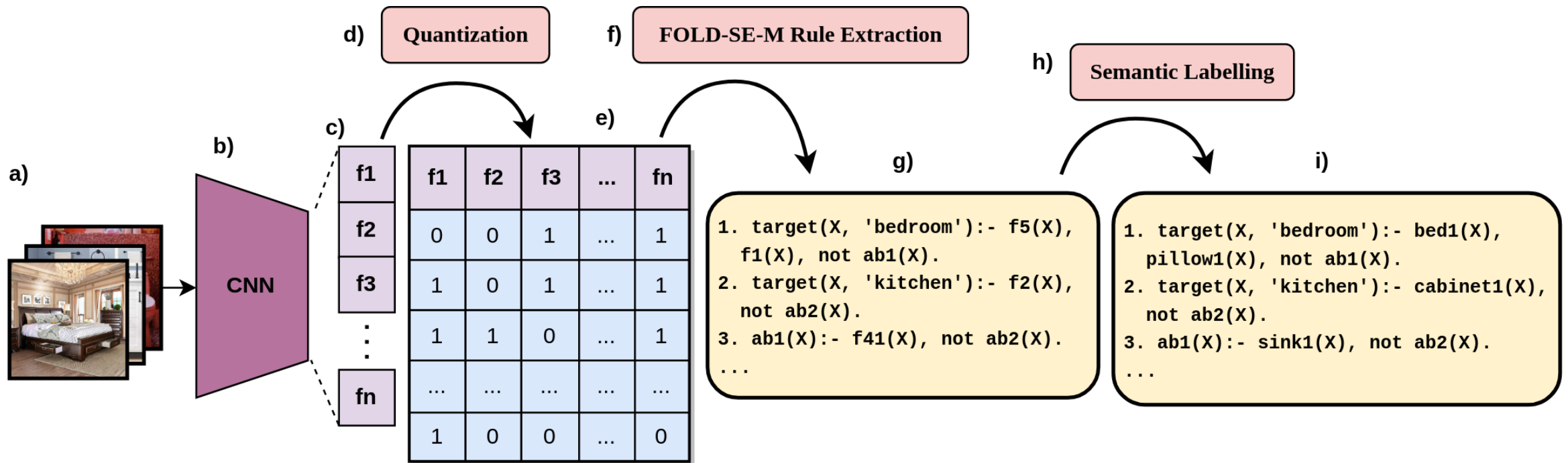
NeSyFOLD, An interpretable model from CNN

Idea:

- The last layer filters implicitly learn to represent high level concepts
- What if we could learn rules for every class based on the concepts learnt by the filters?

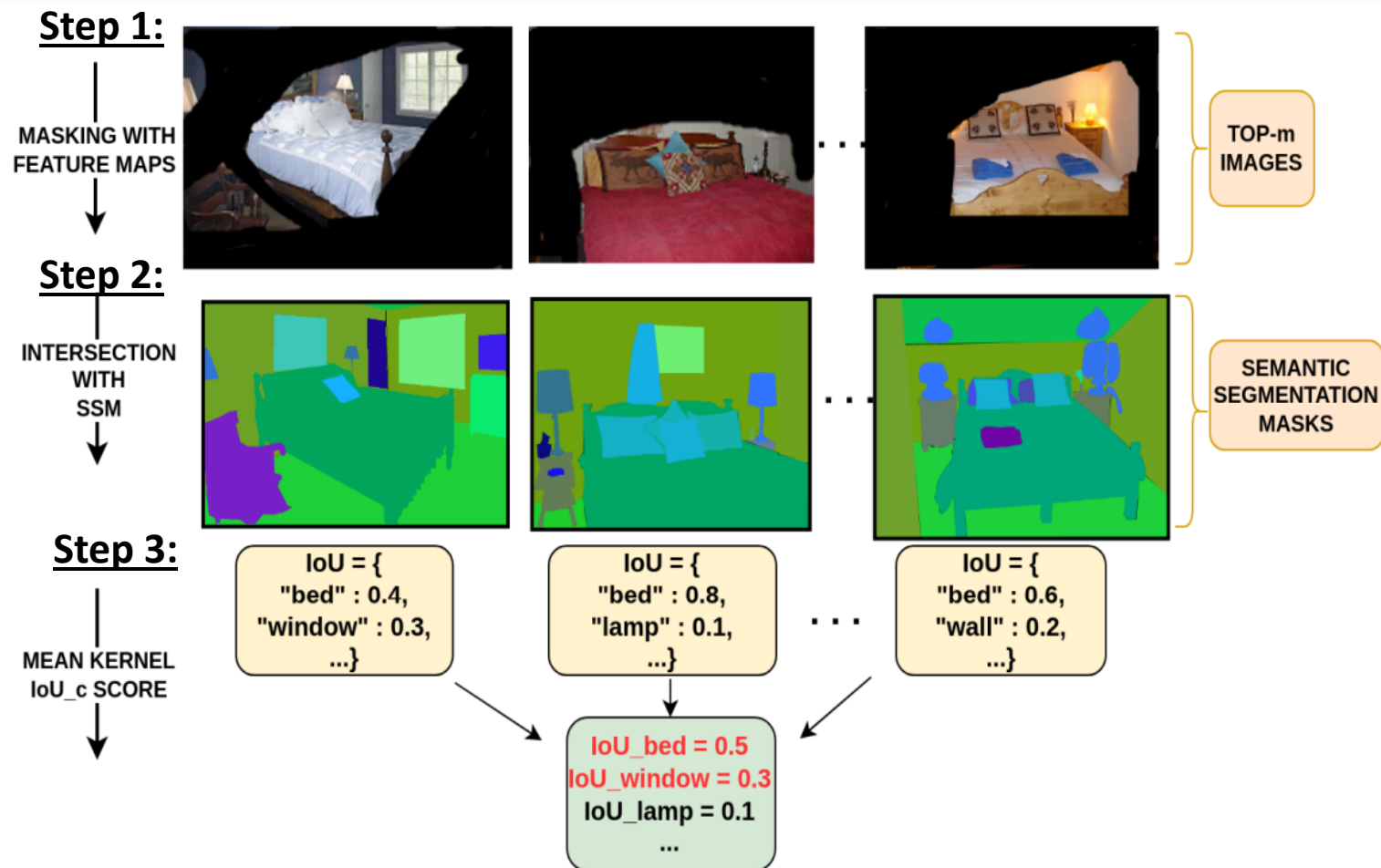
NeSyFOLD is a framework for generating rules from a CNN and creating an interpretable NeSy model

NeSyFOLD, An interpretable model from CNN



- Use the CNN + the labelled rule-set for prediction
- Use s(CASP) ASP engine to get justifications as well

Semantic Labelling of Predicates



Experiments

RULE-SET:

1. `target(X, 'kitchen') :- cabinet2(X), not ab1(X).`
2. `target(X, 'bedroom') :- bed1(X).`
3. `target(X, 'bathroom') :- sink1_toilet1(X).`
4. `target(X, 'bathroom') :- wall2(X).`
5. `target(X, 'kitchen') :- cabinet1(X).`
6. `target(X, 'bedroom') :- bed2(X).`
7. `target(X, 'bathroom') :- wall1(X).`
8. `target(X, 'kitchen') :-
 work_surfacel_kitchen_island1(X).`
9. `ab1(X) :- not cabinet1(X), bed1(X).`

s(CASP) justification

JUSTIFICATION:

'target' holds (for `img`, and `kitchen`),
because '`cabinet2`' holds (for `img`), and
there is no evidence that
'`ab1`' holds (for `img`),
because there is no evidence that
'`cabinet1`' holds (for `img`) and
'`bed1`' holds (for `img`).

MODEL:

{`target(img, kitchen)`, `cabinet2(img)`,
`not ab1(img)`, `not cabinet1(img)`, `bed1(img)`}

Query:

`?- target(img, X).`

Experiments

- **6%** better accuracy than the previous SOTA
- **83%** smaller rule-set size

Limitations:

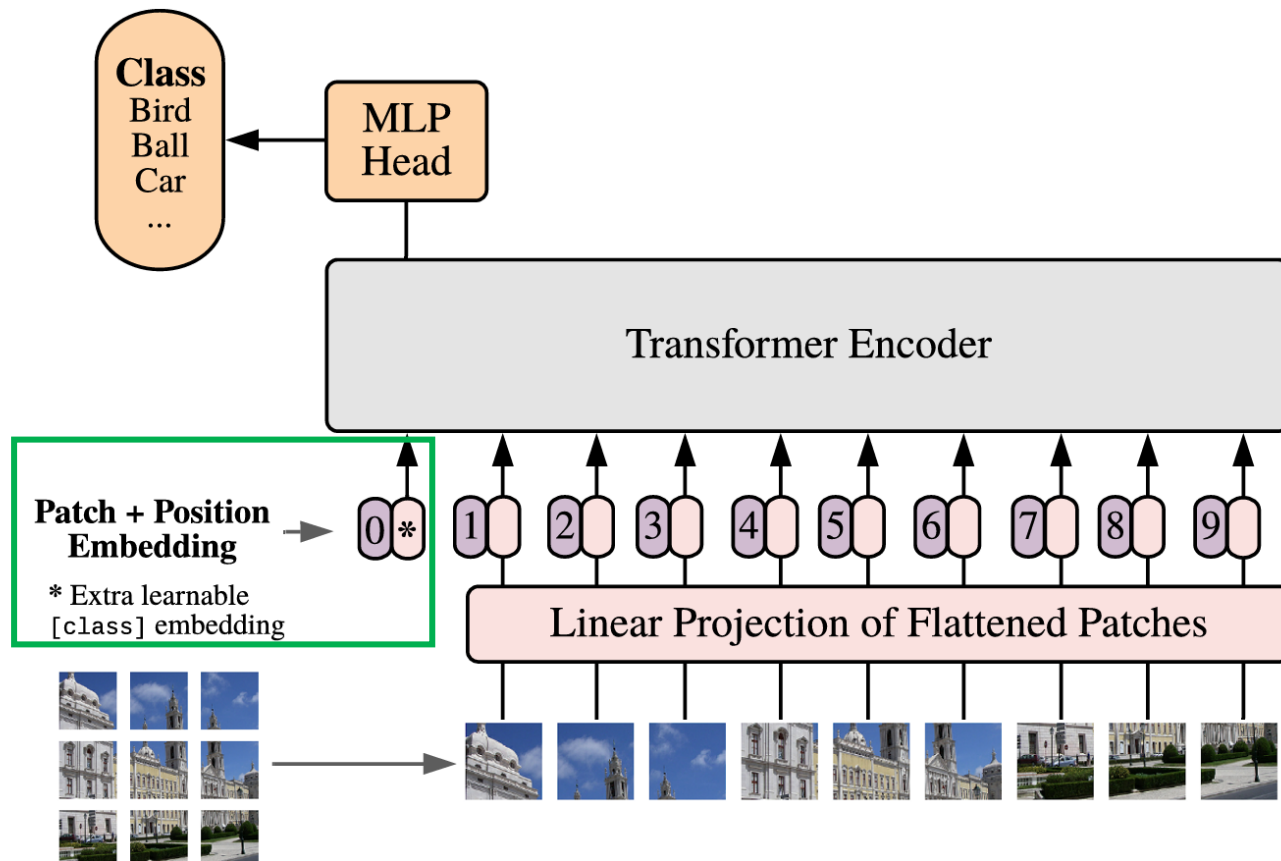
- 12% acc. drop compared to vanilla CNN
- The binarization cause information loss

Idea:

- Binarize the filters while training
- Accuracy drop reduces to **3%**
- Paper accepted at NeSy 2026

Data	Algo	Fid.	Acc.	Pred.	Size
P2	ERIC	0.89 ± 0.01	0.89 ± 0.01	11 ± 1	52 ± 8
	NF	0.93 ± 0.01	0.92 ± 0.01	16 ± 2	28 ± 5
	NF-E	0.93 ± 0.01	0.92 ± 0.01	8 ± 2	12 ± 5
P3.1	ERIC	0.82 ± 0.01	0.81 ± 0.01	33 ± 4	118 ± 13
	NF	0.85 ± 0.03	0.84 ± 0.03	28 ± 6	49 ± 9
	NF-E	0.87 ± 0.02	0.86 ± 0.02	10 ± 3	16 ± 5
P3.2	ERIC	-	0.90	33	127
	NF	0.94 ± 0.0	0.92 ± 0.0	16 ± 4	26 ± 7
	NF-E	0.92 ± 0.01	0.91 ± 0.01	6 ± 1	7 ± 1
P3.3	ERIC	-	0.75	44	176
	NF	0.83 ± 0.01	0.79 ± 0.01	32 ± 5	60 ± 11
	NF-E	0.82 ± 0.03	0.78 ± 0.03	9 ± 0	23 ± 4
P5	ERIC	0.65 ± 0.01	0.63 ± 0.01	57 ± 4	171 ± 10
	NF	0.67 ± 0.03	0.64 ± 0.03	56 ± 3	131 ± 10
	NF-E	0.70 ± 0.02	0.67 ± 0.02	12 ± 4	30 ± 8
P10	ERIC	0.39 ± 0.01	0.36 ± 0.01	85 ± 6	208 ± 16
	NF	0.46 ± 0.02	0.42 ± 0.02	84 ± 6	131 ± 11
	NF-E	0.49 ± 0.02	0.44 ± 0.01	36 ± 4	65 ± 9
GT43	ERIC	0.73 ± 0.01	0.73 ± 0.01	232 ± 3	626 ± 28
	NF	0.75 ± 0.04	0.75 ± 0.04	206 ± 28	418 ± 79
	NF-E	0.85 ± 0.13	0.85 ± 0.13	58 ± 14	99 ± 28
MS	ERIC	0.70 ± 0.01	0.72 ± 0.01	71 ± 4	211 ± 15
	NF	0.78 ± 0.02	0.75 ± 0.02	63 ± 8	120 ± 19
	NF-E	0.80 ± 0.03	0.78 ± 0.03	20 ± 4	36 ± 9

The Vision Transformer



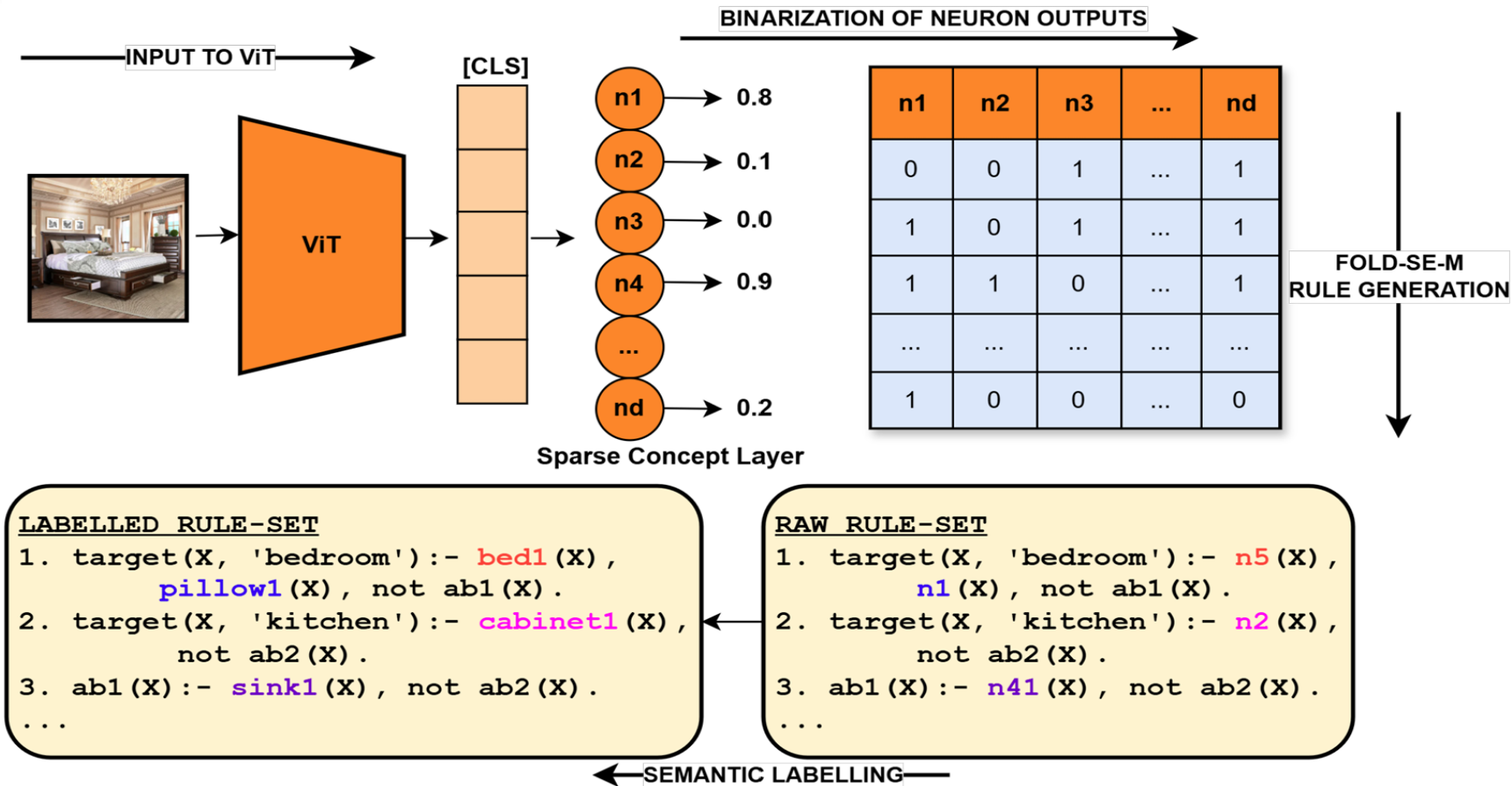
- No Filters = Nothing to binarize
- No feature localization

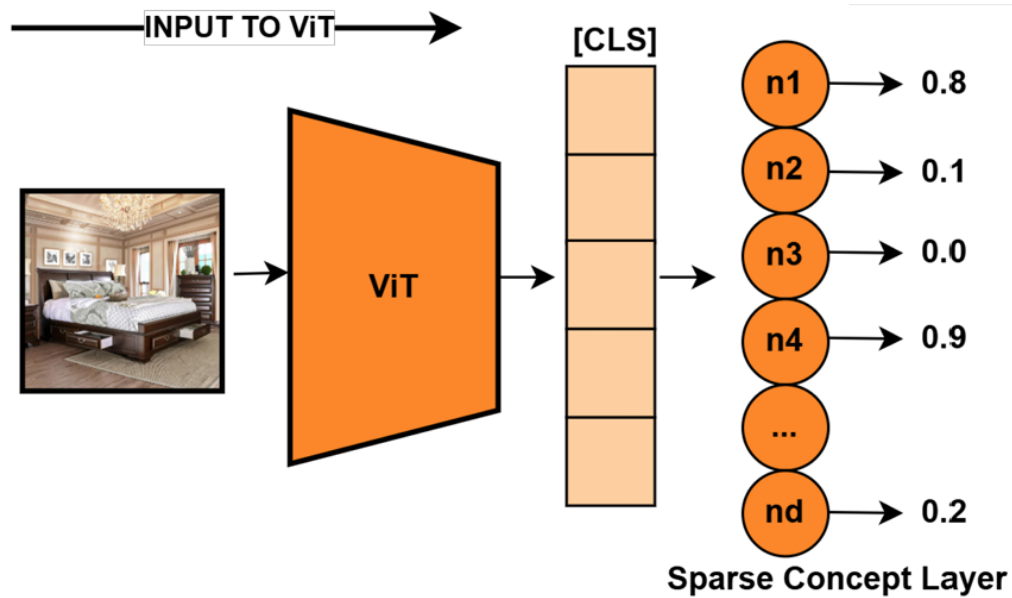
However,

- Class embedding is available containing dense info about class

Idea: Extract the semantic info. from the class embedding via a “sparse concept layer” that has neurons which could be binarized

The NeSyViT Framework

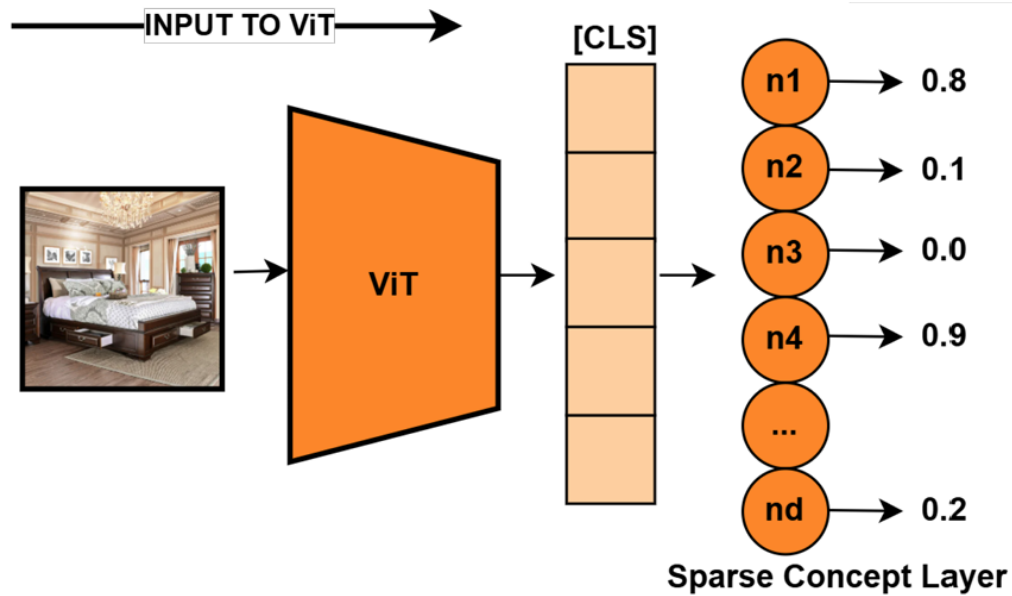




Three training objectives:

1. Similar vectors for images of the same class for better classification
2. The output values should be as close to 0 or 1 as possible, binarization causes minimal loss
3. Only a few neurons should activate leading to sparse vectors referencing only a few concepts

$$L = \alpha L_{supcon} + \beta L_{entropy} + \gamma L_{sparsity}$$



$$\mathcal{L}_{supcon}^i = -\frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau)}{\sum_{a \in \mathcal{A}(i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_a / \tau)}$$

$$\mathcal{L}_{supcon} = \frac{1}{N} \sum_{i \in \mathcal{B}} \mathcal{L}_{supcon}^i$$

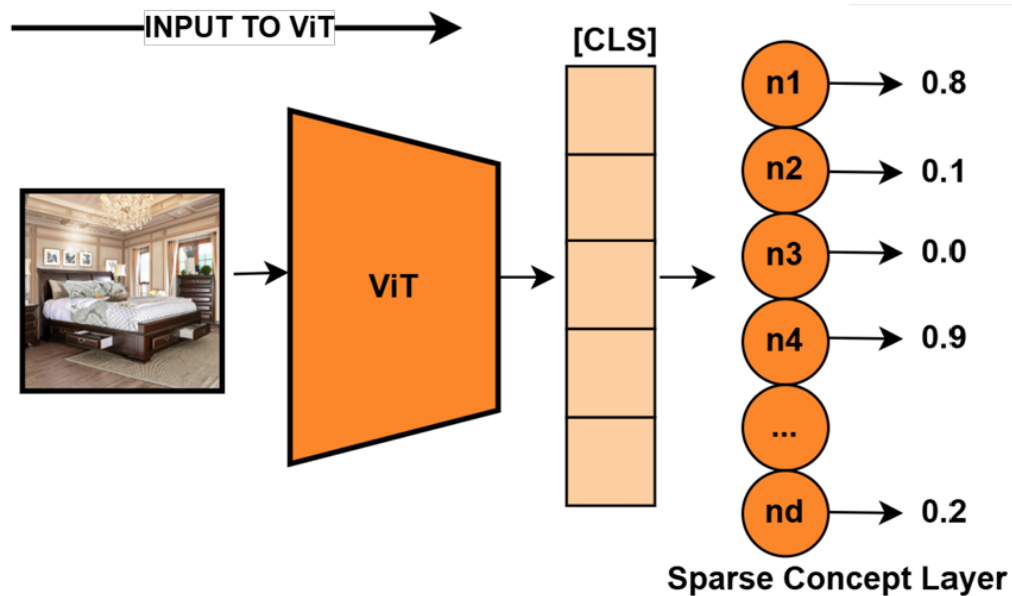
$$\mathcal{L}_{entropy} = -\frac{1}{ND} \sum_{i=1}^N \sum_{j=1}^D [z_{i,j} \log(z_{i,j} + \epsilon) + (1 - z_{i,j}) \log(1 - z_{i,j} + \epsilon)]$$

$$\mathcal{L}_{sparsity} = \frac{1}{ND} \sum_{i=1}^N \sum_{j=1}^D |z_{i,j}|$$

Three training objectives:

1. Supervised Contrastive Loss : Pushes similar vectors together and dissimilar away
2. Entropy loss: penalizes values that have high entropy i.e. are not close to 0 or 1
3. Sparsity loss: Of all the values in the vector pushes only a few to be close to 1

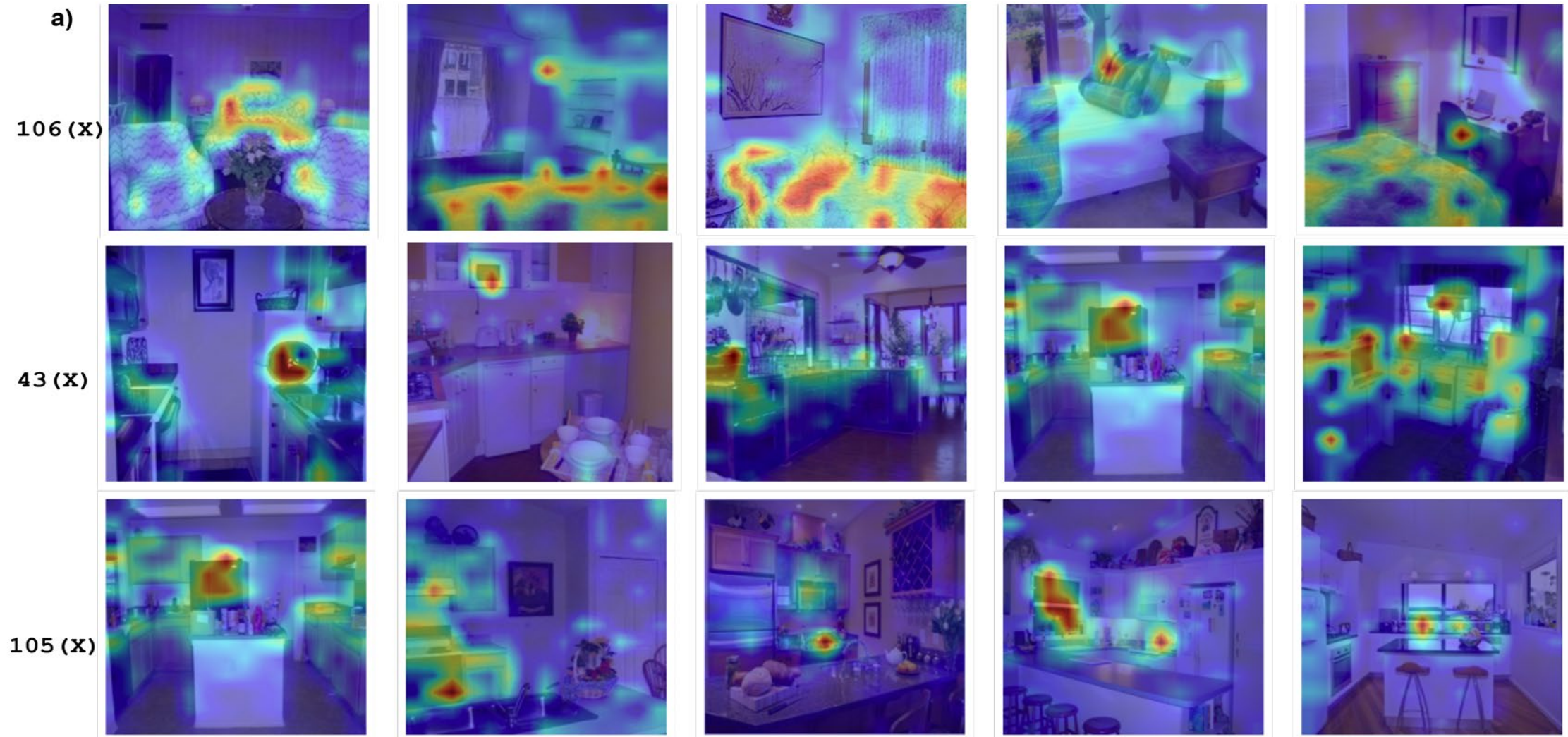
$$L = \alpha L_{supcon} + \beta L_{entropy} + \gamma L_{sparsity}$$



**6% increase in accuracy over vanilla ViT
in addition to an interpretable rule-set**

Three training objectives:

1. **Supervised Contrastive Loss** : Pushes similar vectors together and dissimilar away
2. **Entropy loss**: penalizes values that have high entropy i.e. are not close to 0 or 1
3. **Sparsity loss**: Of all the values in the vector pushes only a few to be close to 1



b) RAW RULE-SET P3.1:

```
1. target(X,'bedroom') :- 106(X) .
2. target(X,'bathroom') :-
    not 43(X) .
3. target(X,'kitchen') :- 105(X) .
```

LABELLED RULE-SET P3.1:

```
1. target(X,'bedroom') :- bed1(X) .
2. target(X,'bathroom') :-
    not water_cooler1_tray1_refrigerator1_range1(X) .
3. target(X,'kitchen') :- refrigerator2(X) .
```

Why not use the image patch features as well in ViT?

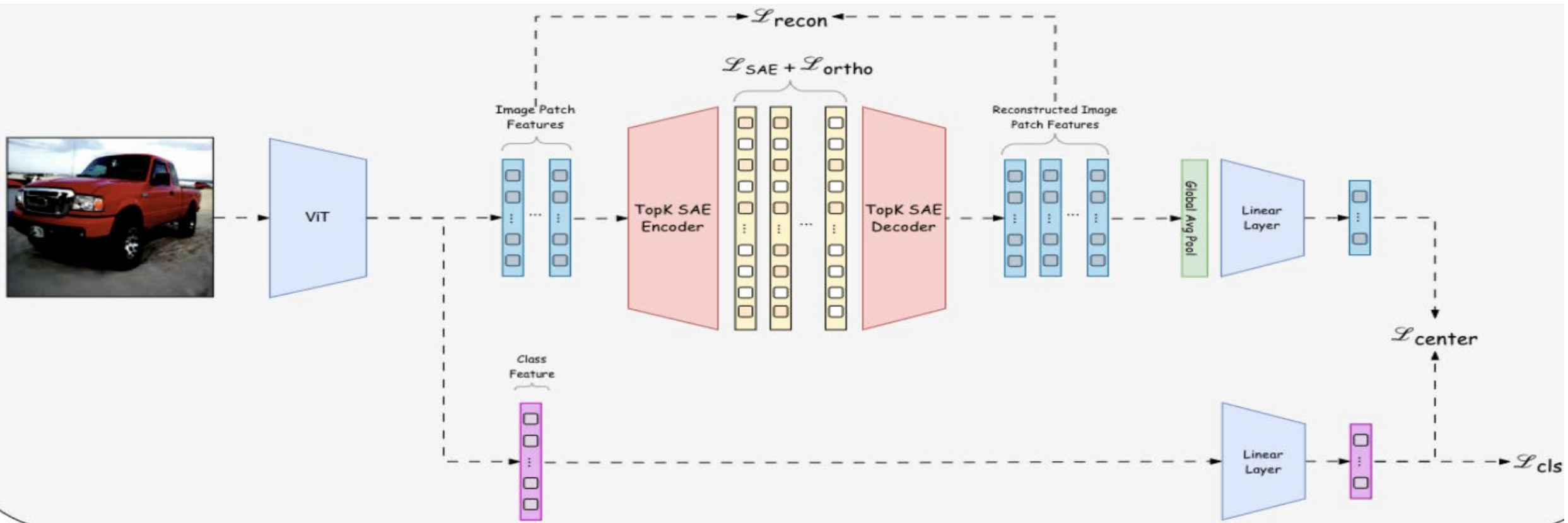
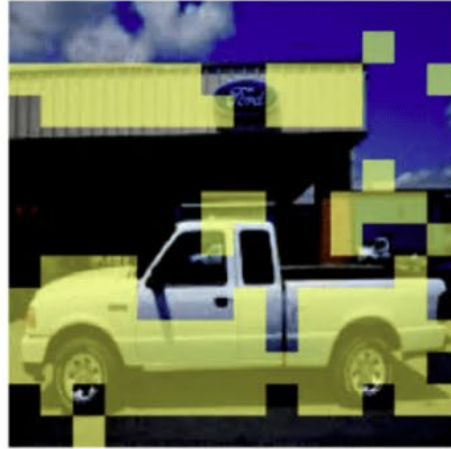


Table 1: Classification accuracy (%) across nine benchmark datasets (mean $\pm$ std). Best results per dataset are highlighted in **bold**.

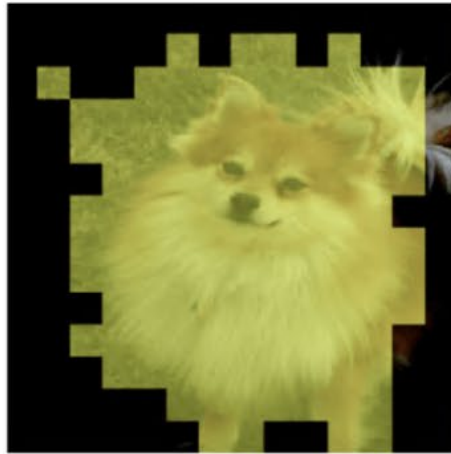
Dataset	PatchSAE	Vanilla ViT	SpIn-ViT (Separate)	SpIn-ViT (Frozen)	SpIn-ViT (Vanilla)	SpIn-ViT (Ours)
Flowers102	94.99 $\pm$ 0.42	98.54 $\pm$ 0.21	97.94 $\pm$ 0.38	98.71 $\pm$ 0.29	98.90 $\pm$ 0.19	99.88 $\pm$ 0.28
Caltech101	95.10 $\pm$ 0.31	96.80 $\pm$ 0.58	95.70 $\pm$ 0.24	93.47 $\pm$ 0.62	96.81 $\pm$ 0.23	97.02 $\pm$ 0.18
Stanford Cars	75.71 $\pm$ 1.53	82.37 $\pm$ 1.24	82.57 $\pm$ 1.39	66.46 $\pm$ 1.64	82.14 $\pm$ 1.21	84.61 $\pm$ 1.02
FGVC	55.43 $\pm$ 2.18	71.11 $\pm$ 1.67	68.32 $\pm$ 2.01	70.12 $\pm$ 1.75	67.63 $\pm$ 1.82	68.43 $\pm$ 1.79
EuroSAT	78.58 $\pm$ 1.08	97.41 $\pm$ 0.22	97.31 $\pm$ 0.21	62.42 $\pm$ 1.88	98.62 $\pm$ 0.18	98.58 $\pm$ 0.21
DTD	76.42 $\pm$ 1.32	80.90 $\pm$ 0.93	80.78 $\pm$ 0.98	53.44 $\pm$ 2.31	82.59 $\pm$ 1.11	84.42 $\pm$ 1.28
Sun397	77.14 $\pm$ 1.27	76.54 $\pm$ 1.12	76.43 $\pm$ 1.22	72.27 $\pm$ 1.89	78.01 $\pm$ 1.04	80.00 $\pm$ 1.38
Food101	84.81 $\pm$ 1.14	85.08 $\pm$ 1.25	85.13 $\pm$ 0.82	86.65 $\pm$ 1.12	85.55 $\pm$ 1.22	86.81 $\pm$ 1.11
OxfordPet	91.96 $\pm$ 0.65	90.32 $\pm$ 0.96	90.22 $\pm$ 0.85	92.48 $\pm$ 1.02	92.09 $\pm$ 0.30	94.96 $\pm$ 0.85
Average	81.13 $\pm$ 1.10	86.56 $\pm$ 0.91	86.04 $\pm$ 0.90	77.34 $\pm$ 1.39	86.93 $\pm$ 0.81	88.30 $\pm$ 0.90

a)

neuron_63(X)



neuron_123(X)



b)

Rules for Cars Dataset

1. $\text{label}(X, \text{'Truck'}) \text{ :- neuron_63}(X), \text{ not ab1}(X).$
2. $\text{label}(X, \text{'Truck'}) \text{ :- not neuron_111}(X), \text{ not neuron_1}(X).$
3. $\text{ab1}(X) \text{ :- neuron_50}, \text{ neuron}(61).$

Rules for OxfordPet Dataset

1. $\text{label}(X, \text{'Dog'}) \text{ :- neuron_123}(X).$
2. $\text{label}(X, \text{'Dog'}) \text{ :- not neuron_50}, \text{ not neuron}(61).$

What to do with Interpretability?

Idea:

- The rule-set can be scrutinized to identify biases
- Specific filter in the CNN can be targeted to unlearn/learn the undesired/desired concepts

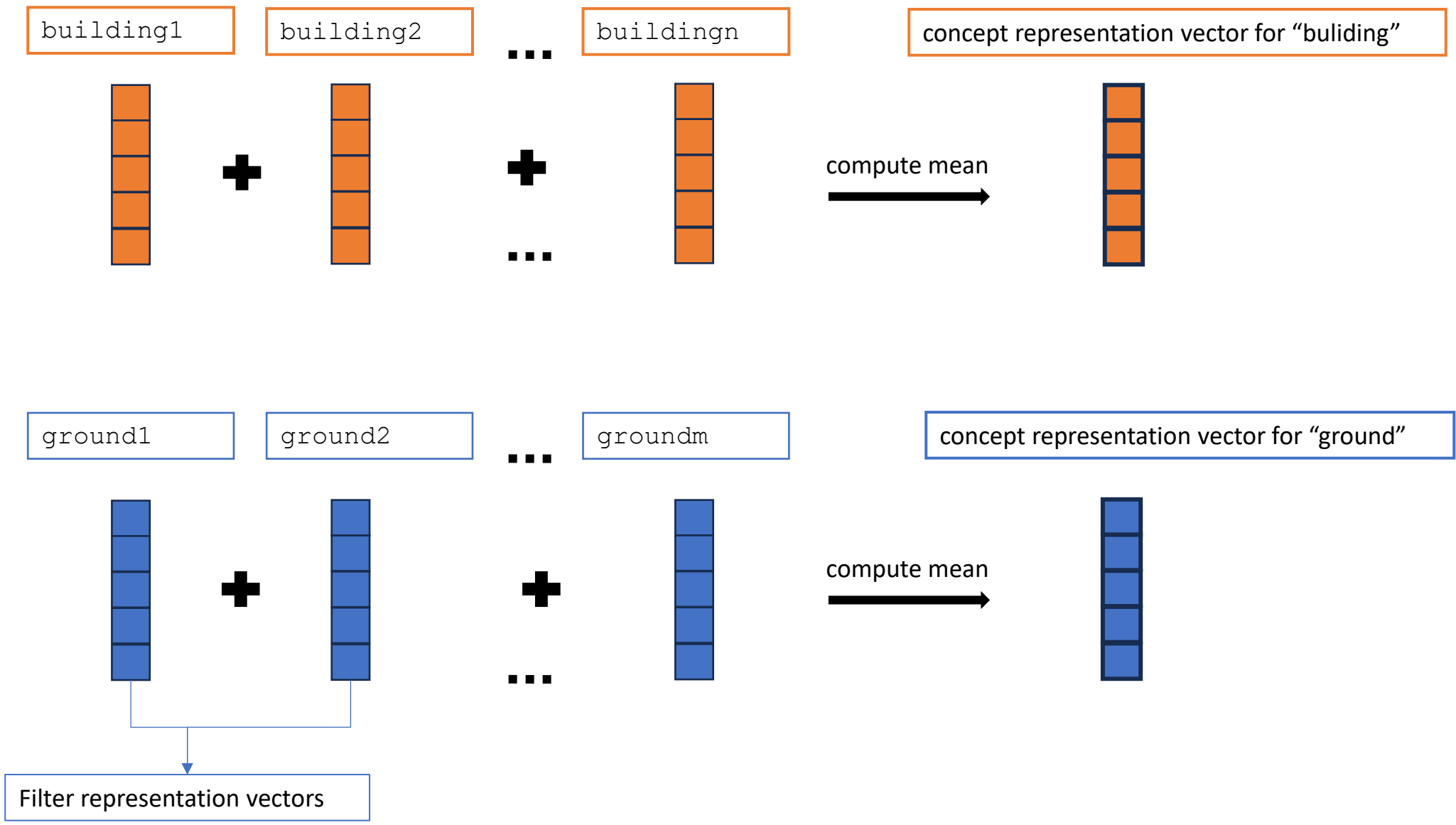
RULE-SET 1:

```
1. target(X,'street') :- building1(X).
2. target(X,'desert_road') :- sky1(X).
3. target(X,'desert_road') :- not building5(X),
    not building2(X), not ab1(X).
4. target(X,'street') :- building6(X).
5. target(X,'street') :- not road1(X).
6. ab1(X) :- not sky2(X), not building4(X),
    building3(X).
```

Step 1: Identify Desired and Undesired Concepts

	<i>'street'</i>	<i>'desert road'</i>
<i>Undesired</i>	1. sky 2. tree	1. sky 2. building
<i>Desired</i>	1. building	1. sand 2. ground 3. desert

Step 2: Compute the concept representation vectors of desired and undesired concepts



Step 3: Use the semantic similarity loss during retraining

$$\mathcal{L}_{SS} = \sum_{i=1}^N \left[\sum_{j=1}^K \left[\lambda_B \sum_{b \in \mathbf{B}} \cos_sim(\mathbf{r}_j^i, \mathbf{r}_b) - \lambda_G \sum_{g \in \mathbf{G}} \cos_sim(\mathbf{r}_j^i, \mathbf{r}_g) \right] \right]$$

- $\cos_sim()$: function for computing cosine similarity between 2 vectors
- $\mathbf{r}_j^i$: The **filter** representation vector of the j^{th} filter for the i^{th} training example
- $\mathbf{r}_b$: The **concept** representation vector of a concept “b” in the **undesirable** concepts $\mathbf{B}$
- $\mathbf{r}_g$: The **concept** representation vector of a concept “g” in the **desirable** concepts $\mathbf{G}$
- N, K : No. of training images and no. of filters in the last layer of the CNN respectively
- λ_B, λ_G : Hyperparameters

Intuition of the loss function:

- The similarity score between the filter representation vectors of the CNN and the concept representation vectors in the desirable/undesirable concepts is calculated
- The loss value is high if the filter representation vectors are similar to the undesirable concept representation vectors
- Hence, minimizing this loss means driving the filters in the CNN to learn representation that are closer to the desirable representations and away from the undesirable representations.

NeSyBiCor Framework

The 3 steps constitute the NeSyBiCor (NeSy Bias Correction) framework

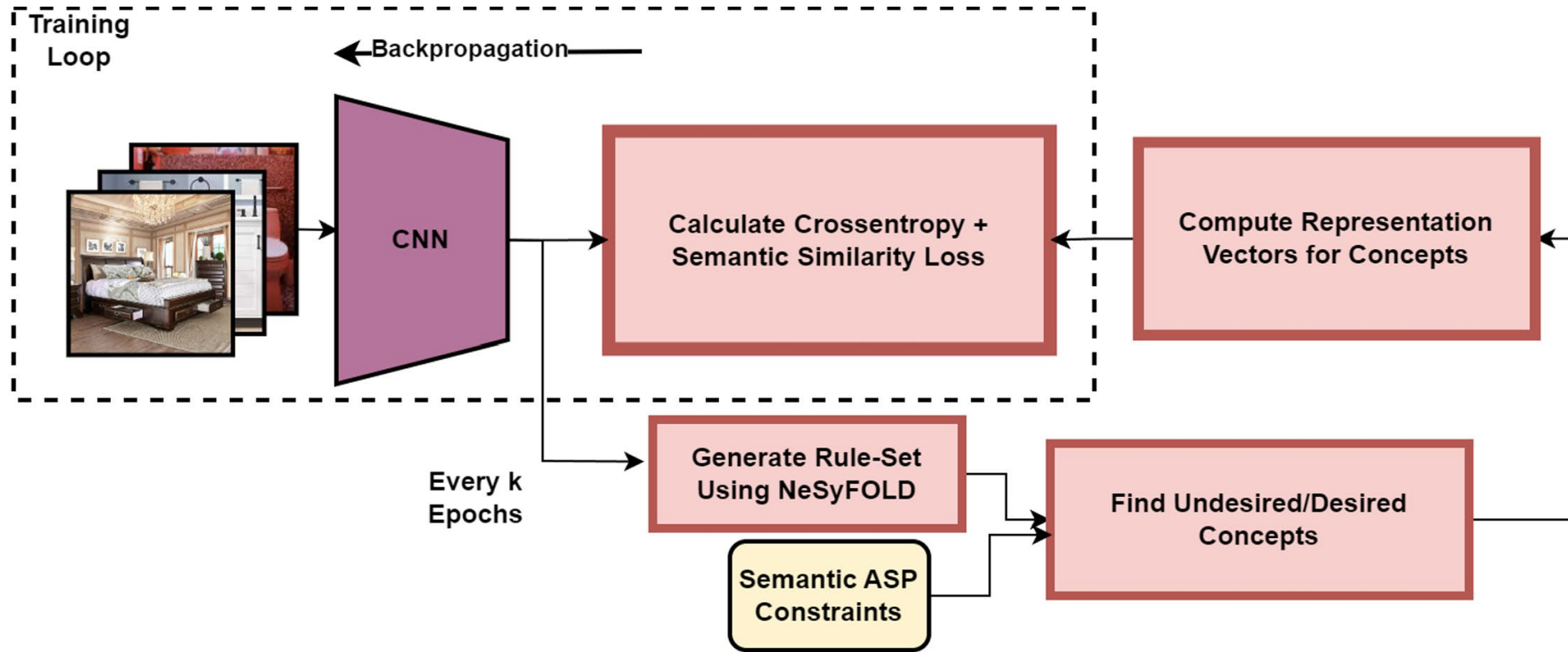
RULE-SET 1:

```
1. target(X,'street') :- building1(X).
2. target(X,'desert_road') :- sky1(X).
3. target(X,'desert_road') :- not building5(X),
    not building2(X), not ab1(X).
4. target(X,'street') :- building6(X).
5. target(X,'street') :- not road1(X).
6. ab1(X) :- not sky2(X), not building4(X),
    building3(X).
```

RULE-SET 1*:

```
1. target(X,'street') :- building1(X).
2. target(X,'desert_road') :- ground1_road1(X).
3. target(X,'street') :- building3(X).
4. target(X,'desert_road') :- not building2(X).
```

NeSyBiCor Framework

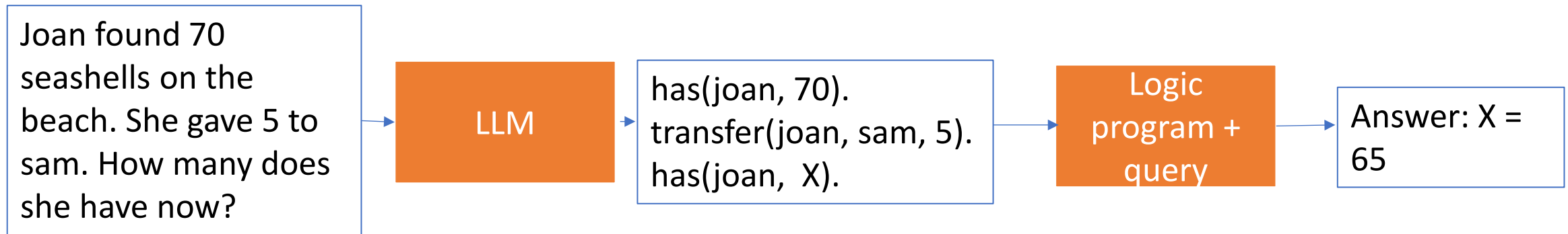




Creating “Fast and Slow” models for better reasoning and reliability

LLM + Logic Programming

Idea: Use the LLM for extracting information, use logic programs for reasoning



STAR Framework

Reliable chatbots can be built using the STAR framework

100% accuracy in a narrow domain

Ongoing Projects

- 1. Can we isolate particular heads in the transformer layers and preserve them?**
 - i. Primarily to uncover how symbolic manipulation works across the LLM
 - ii. [Prev. works](#) (Brinkmann et al. @ACL findings 2024) show how search algorithms are learnt in narrow LLMs
 - iii. Finding ways to ground concepts to groups of neurons in the LLMs to see if they work on OOD graphs
- 2. Using ASP to verify reasoning traces of LLMs**
 - i. LLM generated reasoning traces often are hallucinatory
 - ii. [Feng et al. @ICLR 2026](#) showed verification by generating FOL statements from reasoning steps
 - iii. With ASP systematic verification could be possible even if there are uncertainties (via defaults and exceptions)
- 3. Fast and Slow architecture for planning and intelligent systems**
 - i. Planning in open world domain requires commonsense knowledge + reasoning
 - ii. The STAR framework can be modified to dynamically generate logic rules in memory; execute them via ASP solver for better reasoning; application to robotics
- 4. Extending the interpretability techniques to object detection and domain applications**
 - i. Extend to object detection, should be straightforward.
 - ii. Medical, Agriculture, etc.

Conclusion

Vision: “Building AI that is capable of explaining its decisions and employing systematic reasoning at the same time”

- **NeSyFOLD, NeSyViT and NeSyBiCor** are steps towards making AI models explainable and correctable
- **STAR** framework step towards complementing neural AI with reasoning abilities
- Lot of interesting work to do; Excited to explore collaborations!



Thank You.