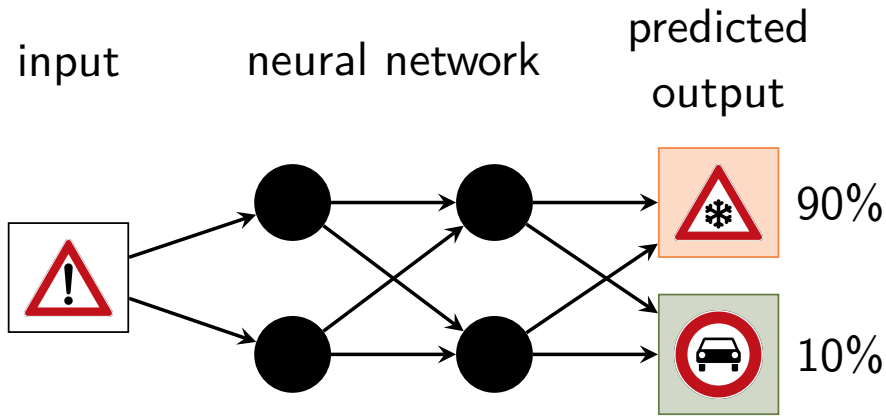
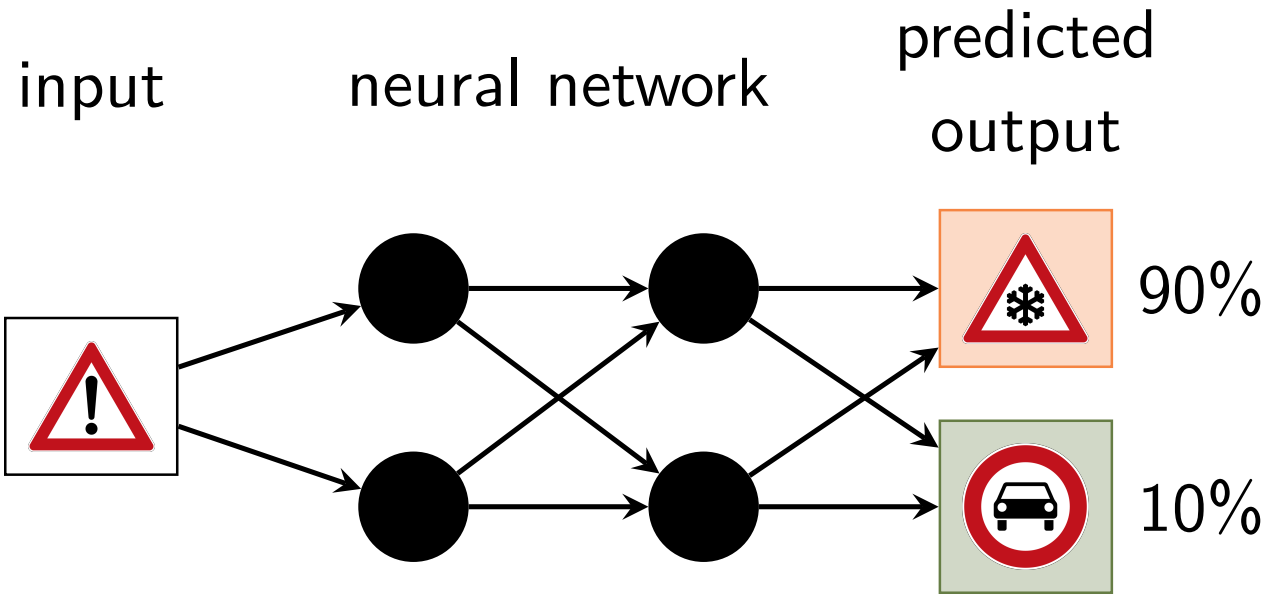
Outside the Box: Abstraction-Based Monitoring of Neural Networks

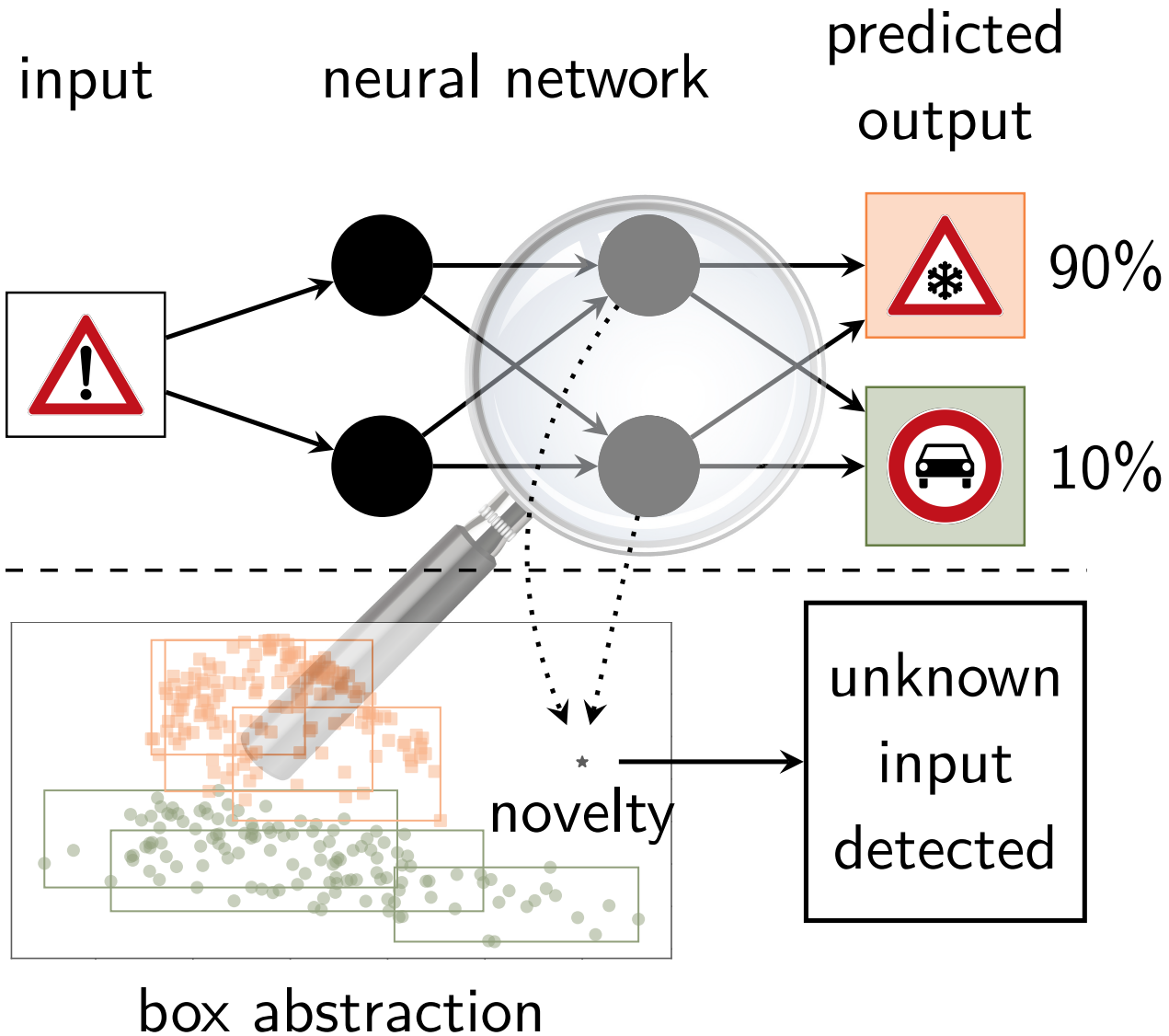
Thomas A. Henzinger, Anna Lukina, **Christian Schilling**



Novelty detection



Novelty detection by runtime monitoring



Overview

Preliminaries

Interval abstraction (basic idea)

Box abstraction (extension to clustered data)

Evaluation

Overview

Preliminaries

Interval abstraction (basic idea)

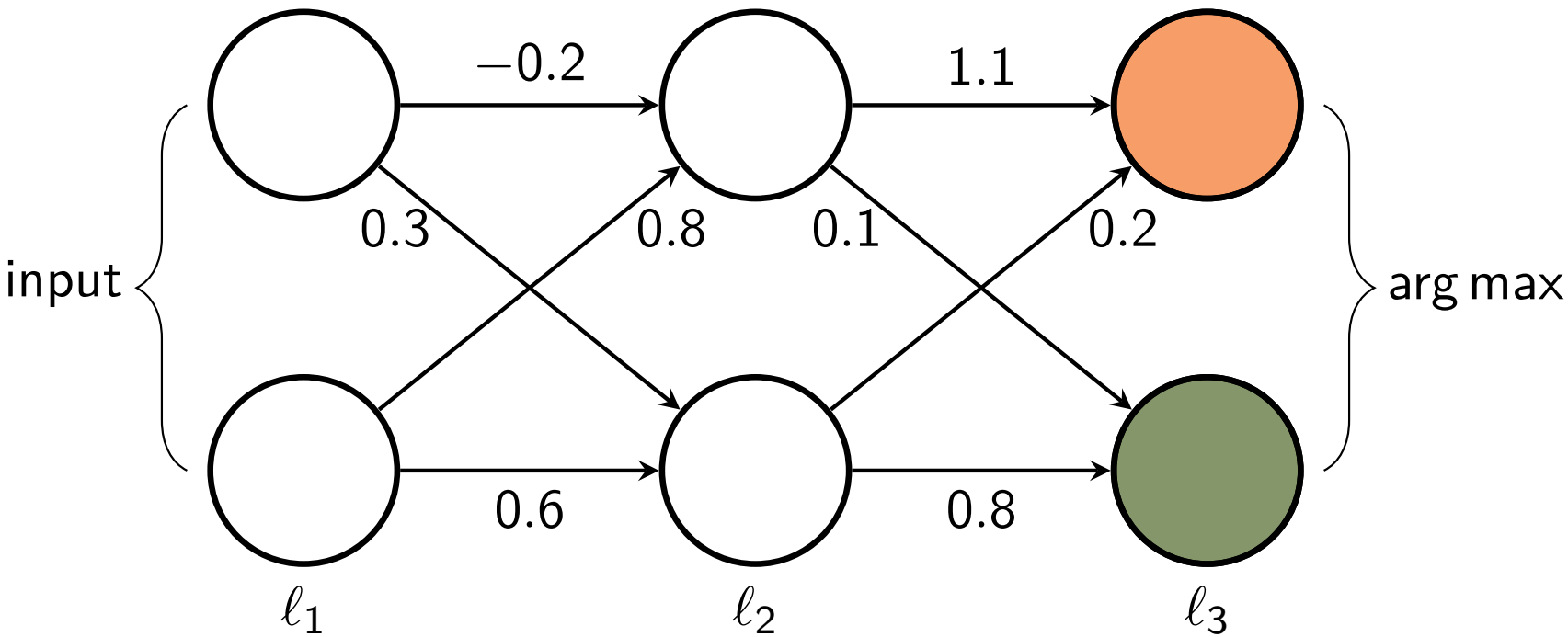
Box abstraction (extension to clustered data)

Evaluation

Example of a neural-network classifier

activation function:

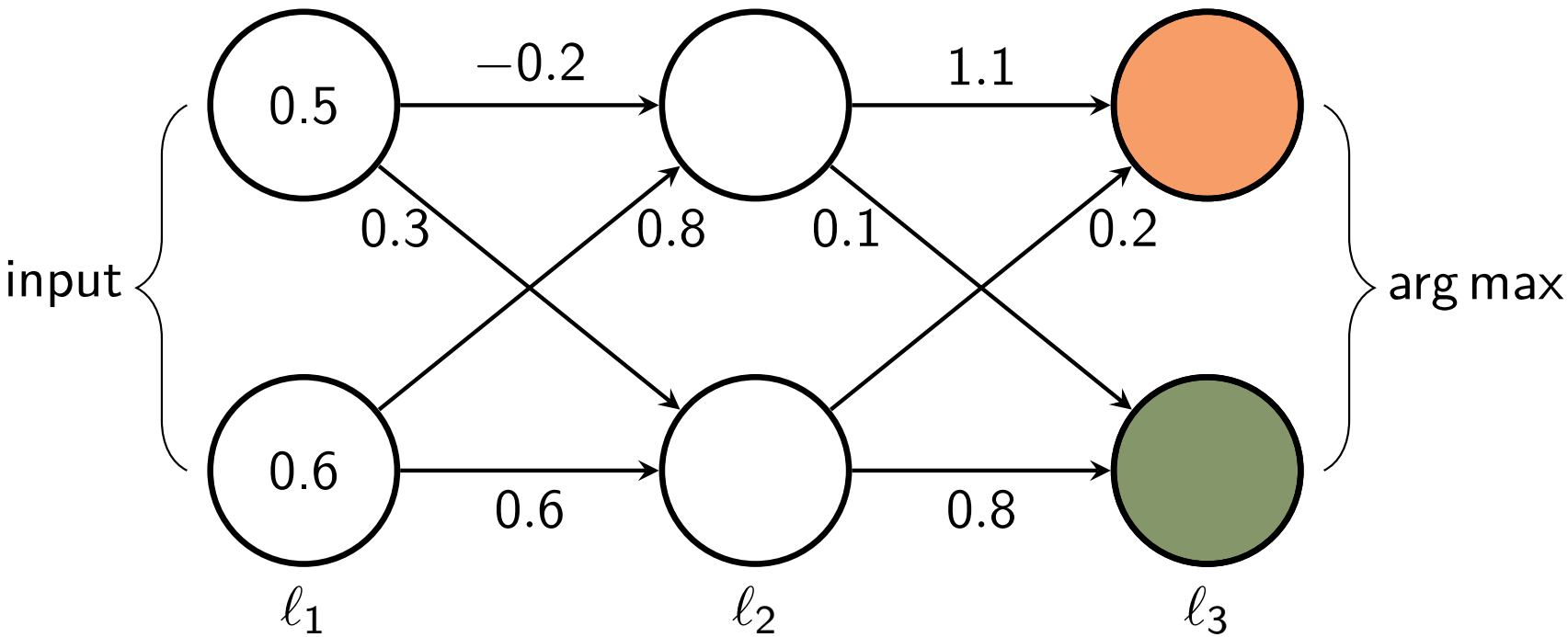
$$\sigma(x) = \max(x, 0)$$



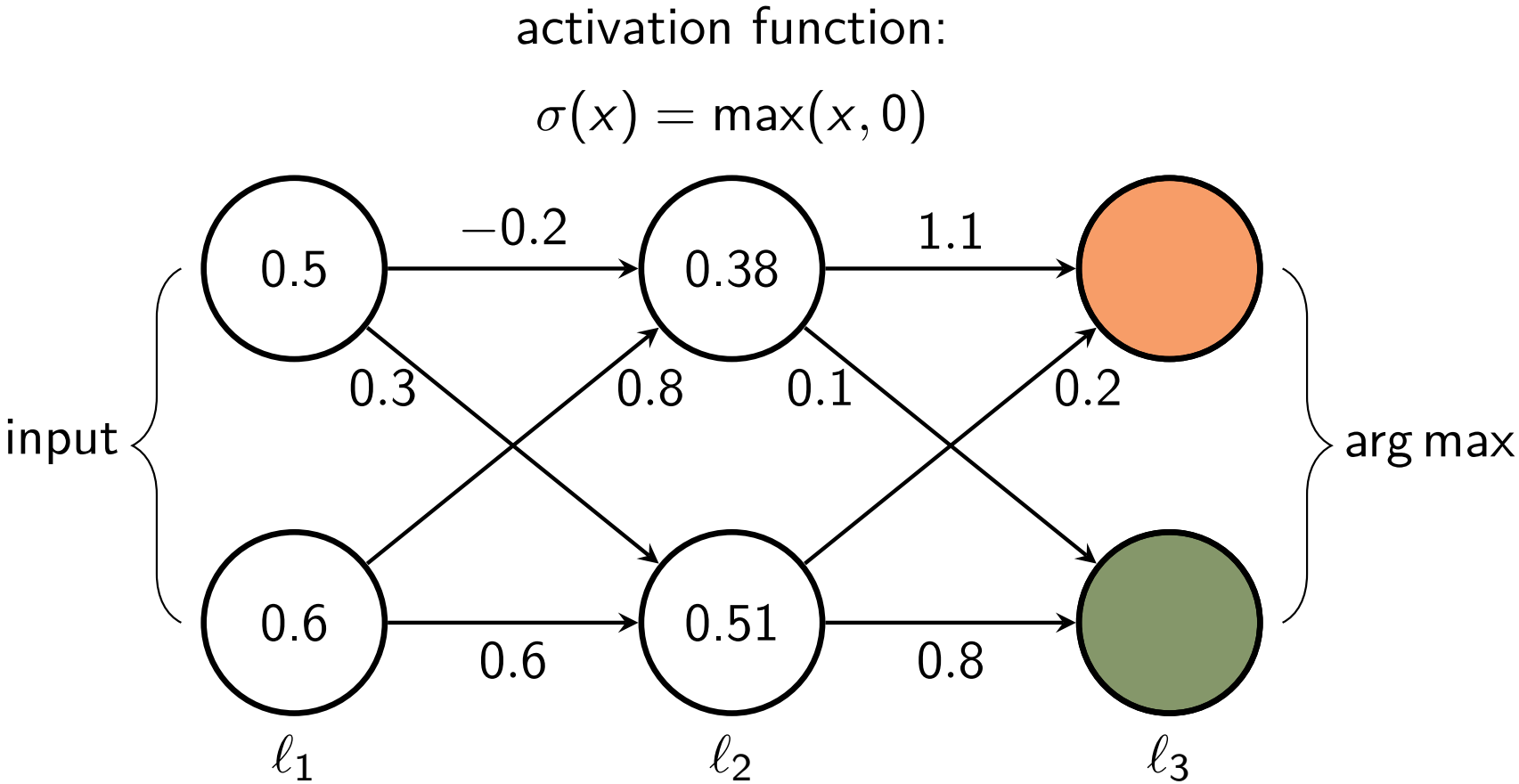
Example of a neural-network classifier

activation function:

$$\sigma(x) = \max(x, 0)$$



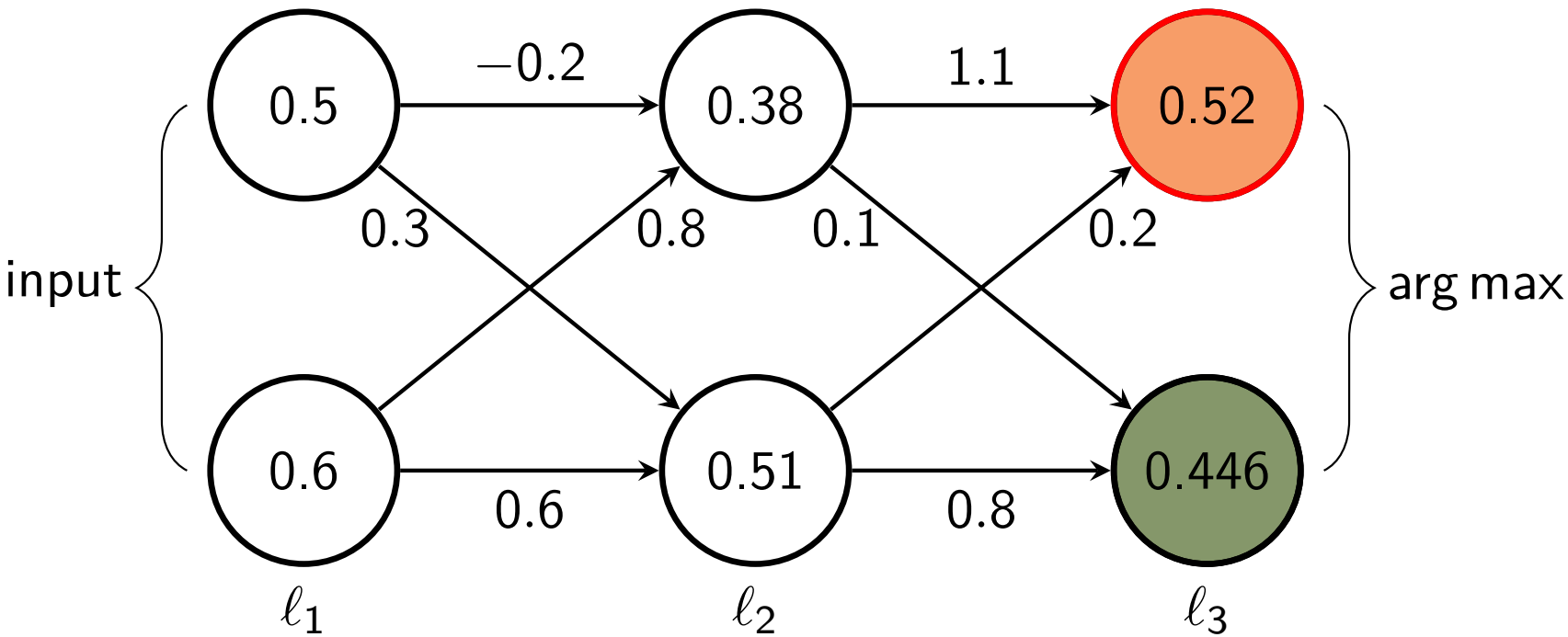
Example of a neural-network classifier



Example of a neural-network classifier

activation function:

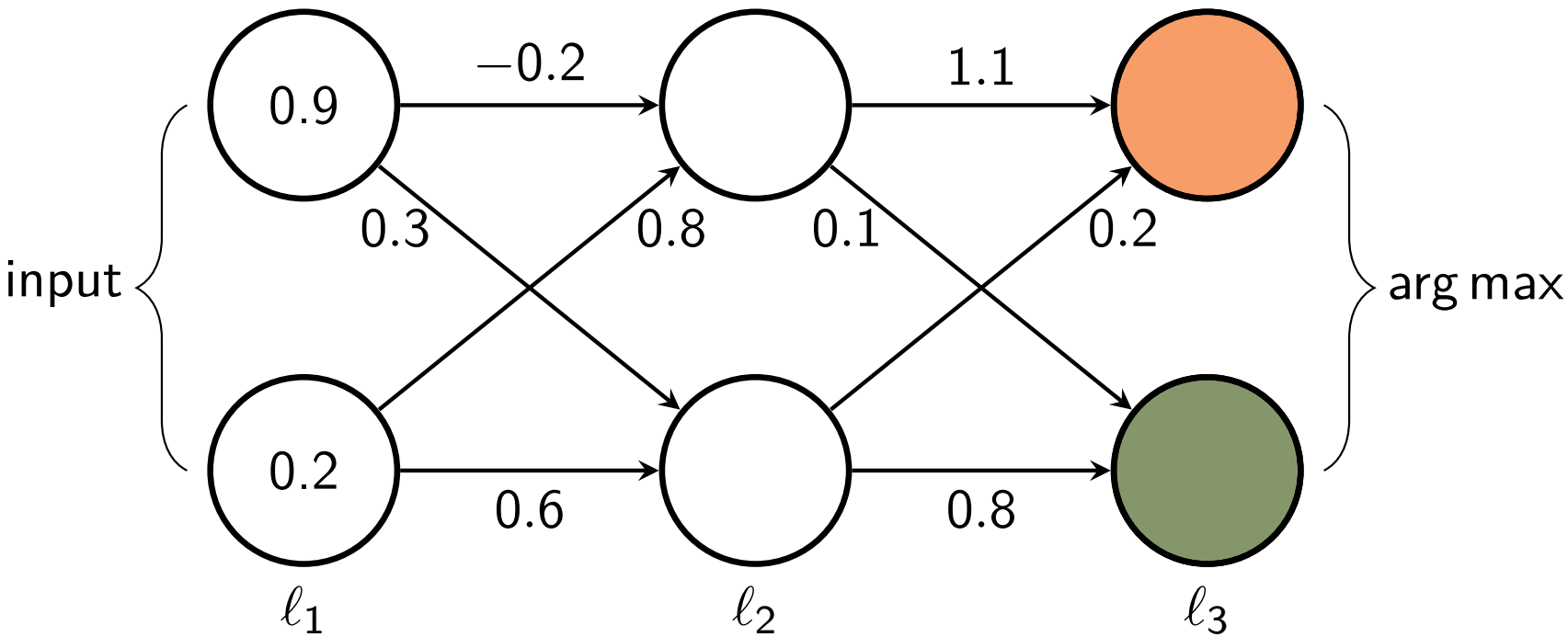
$$\sigma(x) = \max(x, 0)$$



Example of a neural-network classifier

activation function:

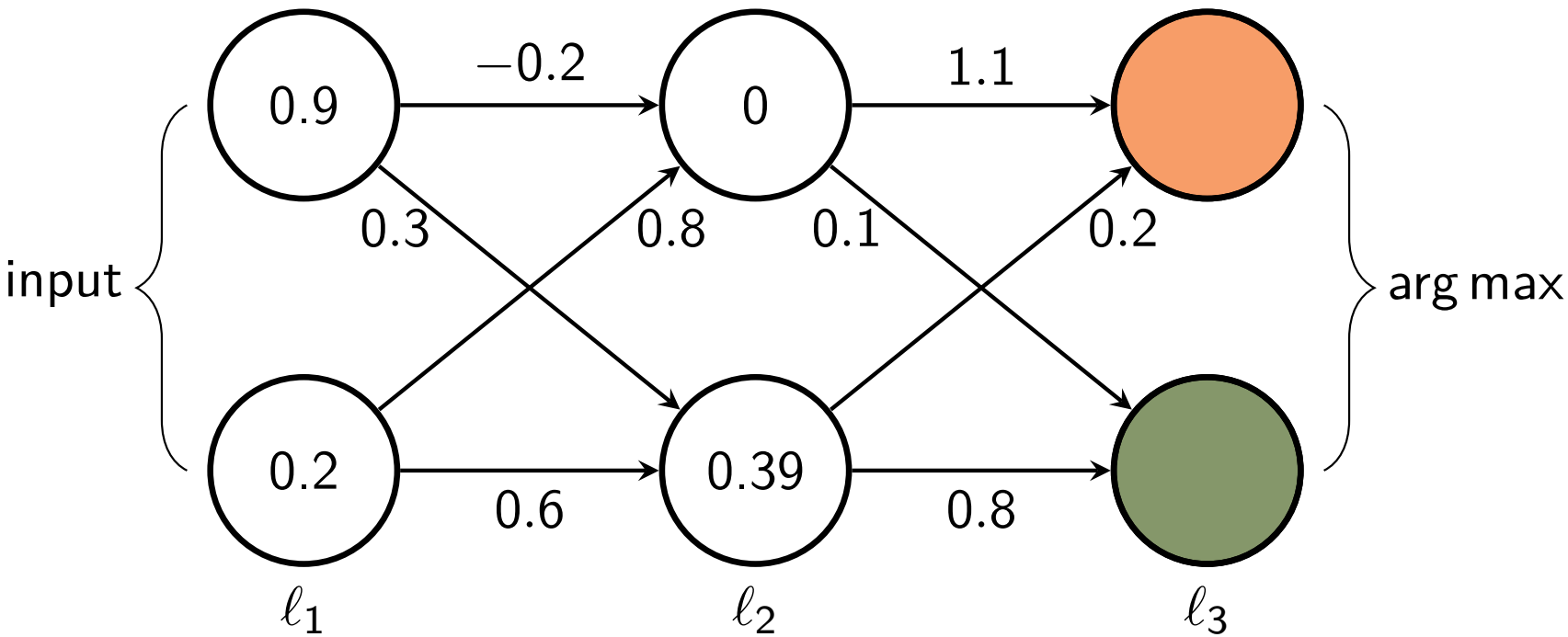
$$\sigma(x) = \max(x, 0)$$



Example of a neural-network classifier

activation function:

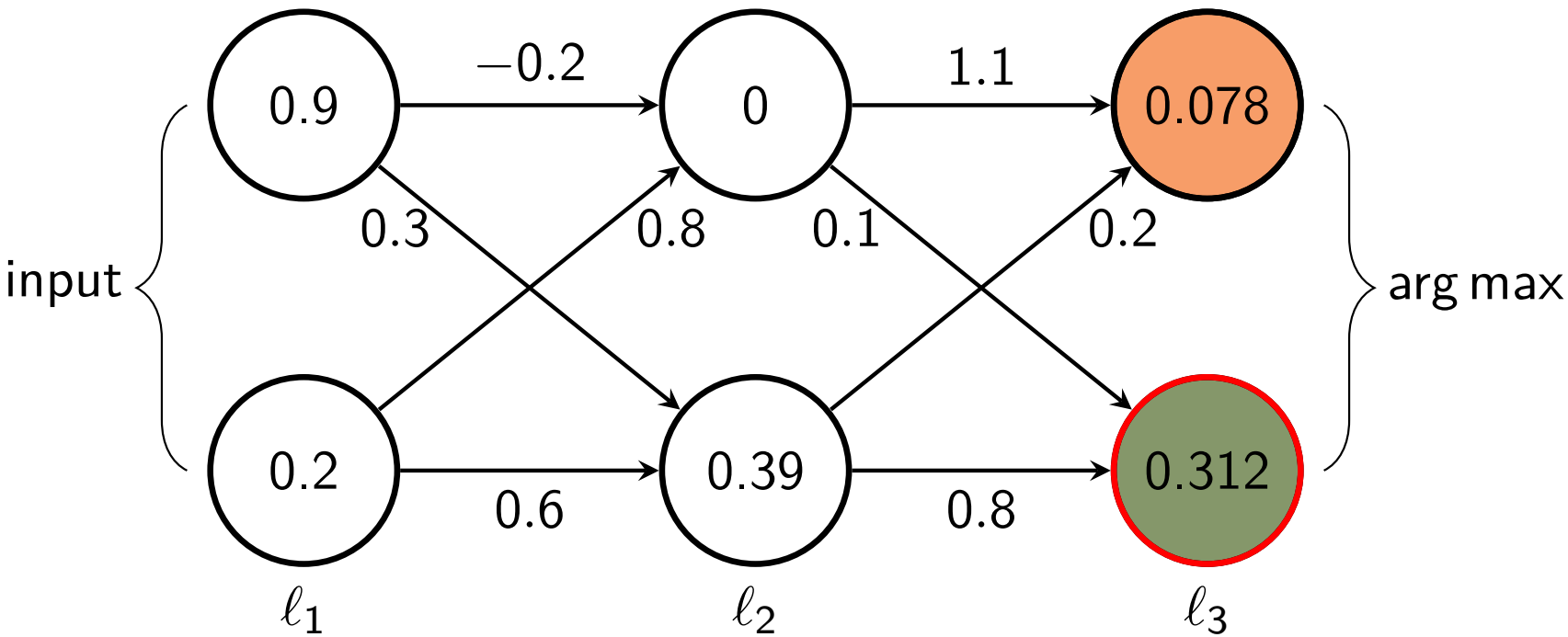
$$\sigma(x) = \max(x, 0)$$



Example of a neural-network classifier

activation function:

$$\sigma(x) = \max(x, 0)$$



Overview

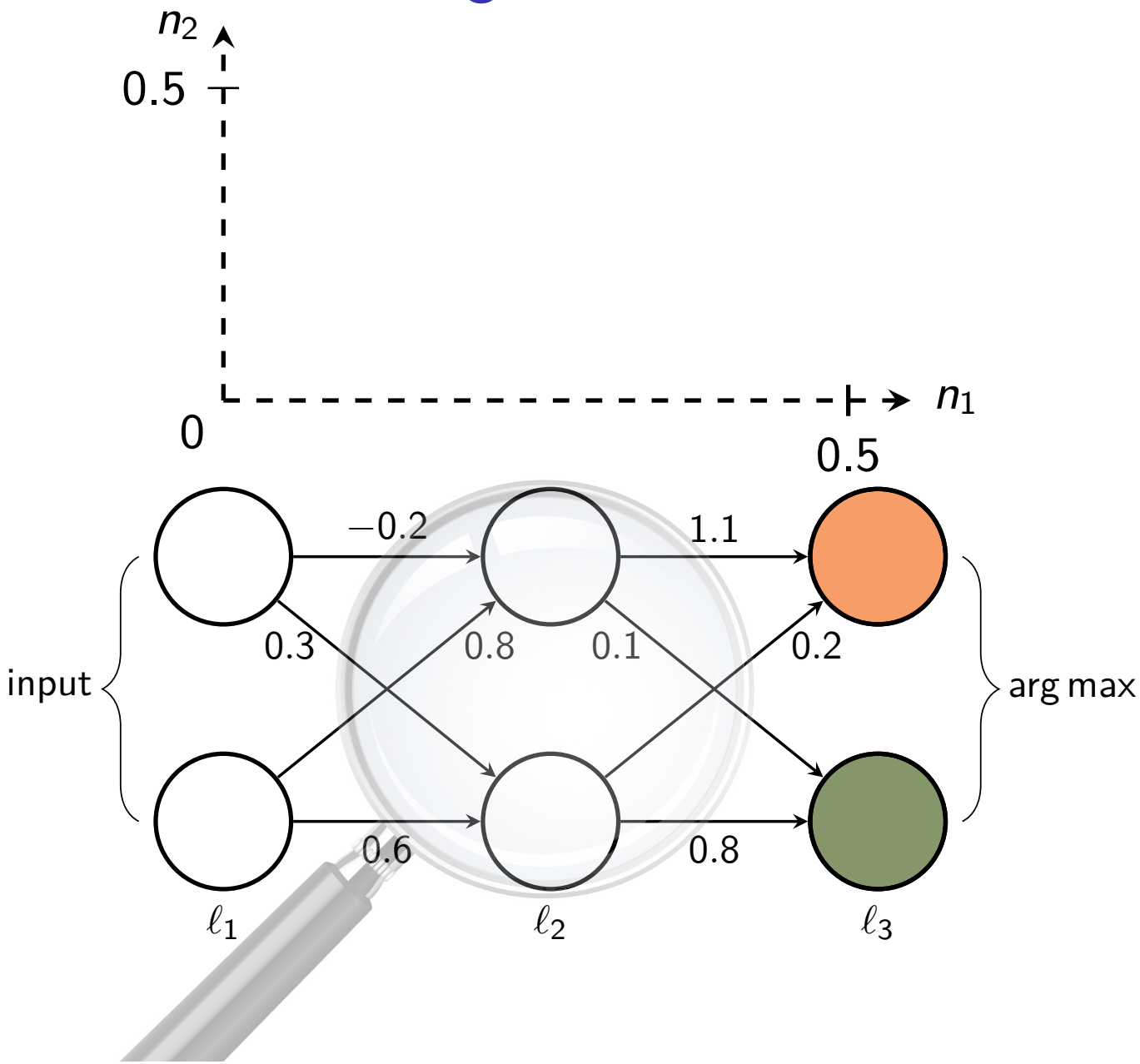
Preliminaries

Interval abstraction (basic idea)

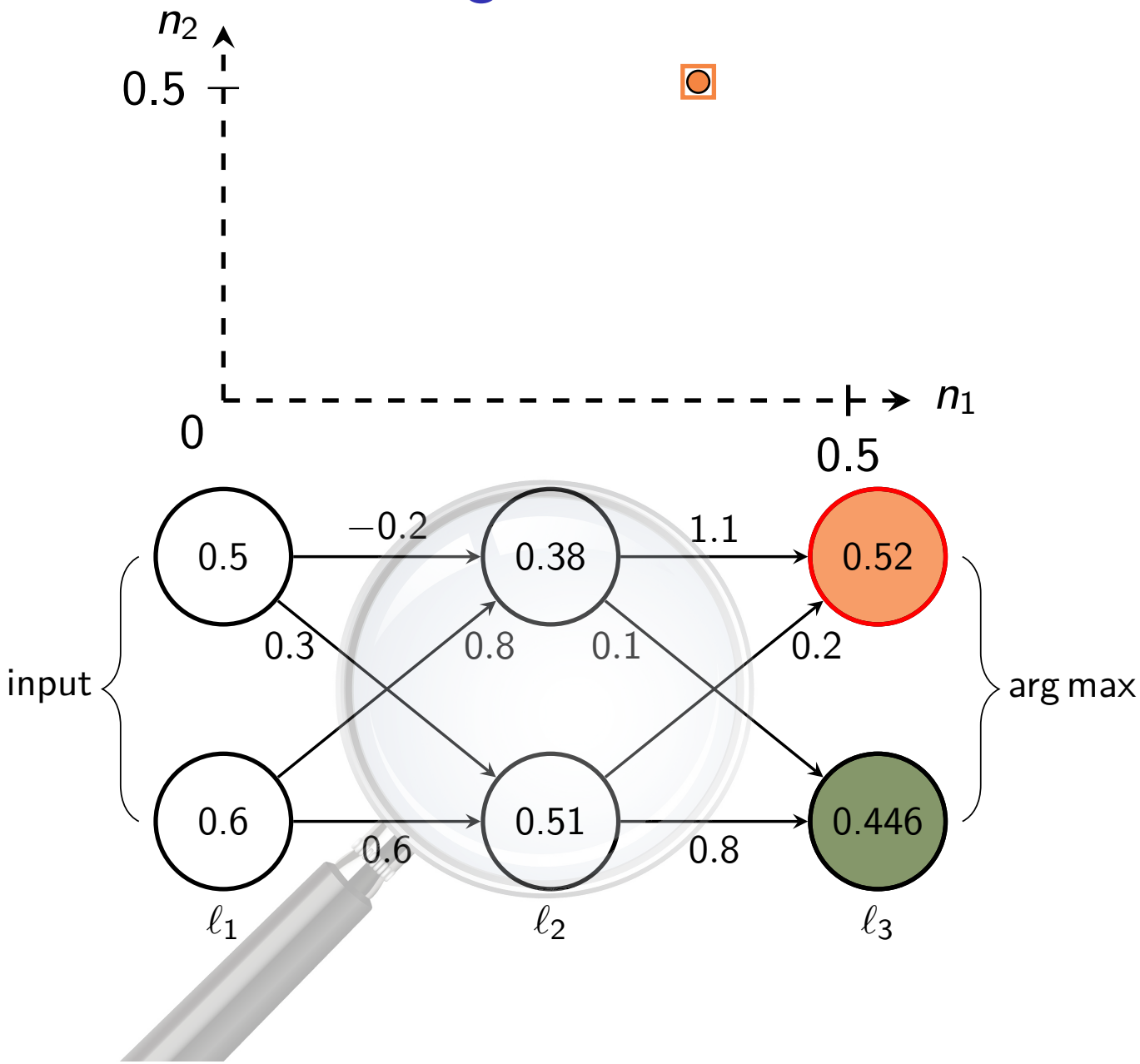
Box abstraction (extension to clustered data)

Evaluation

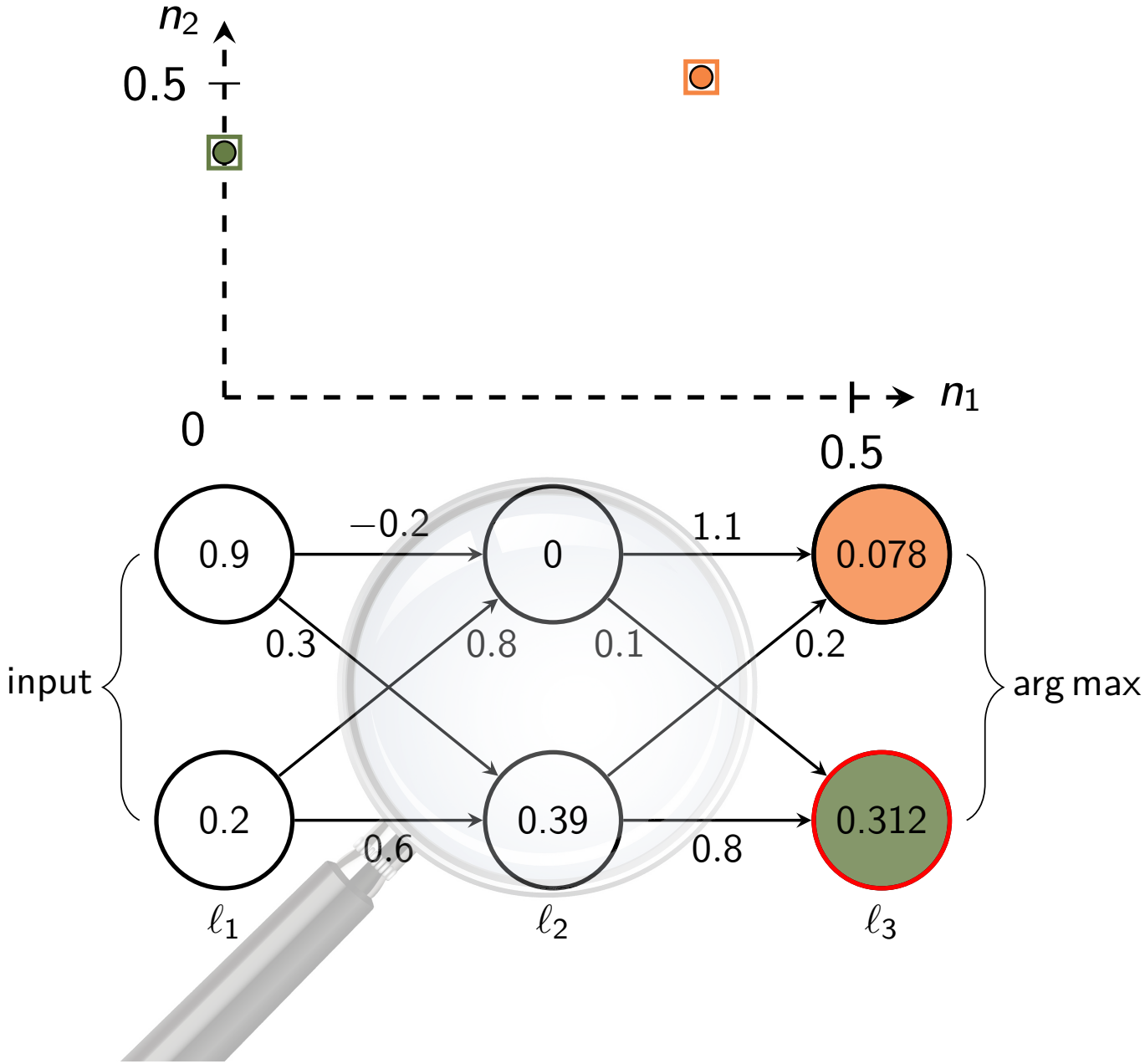
Constructing an interval abstraction



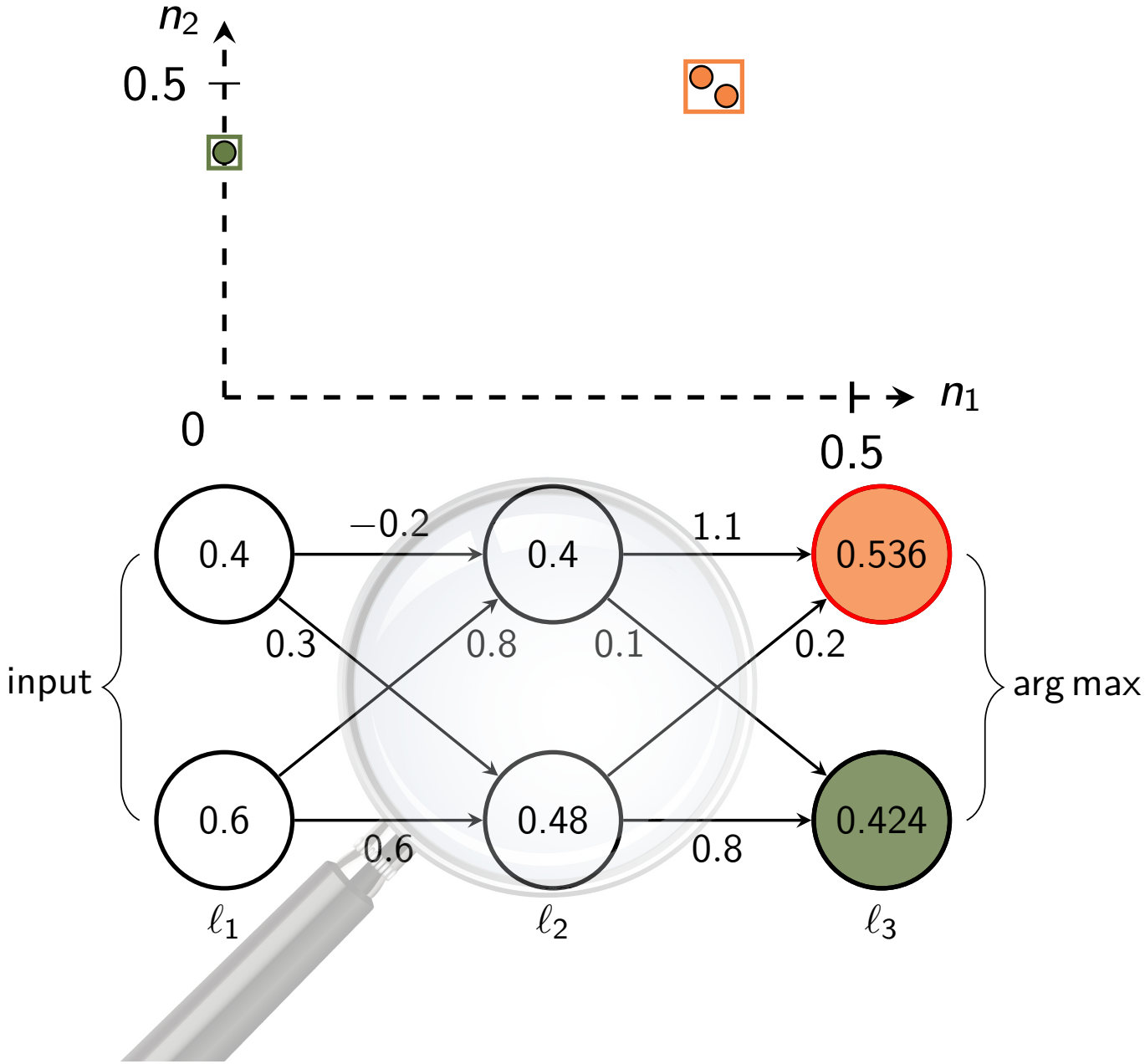
Constructing an interval abstraction



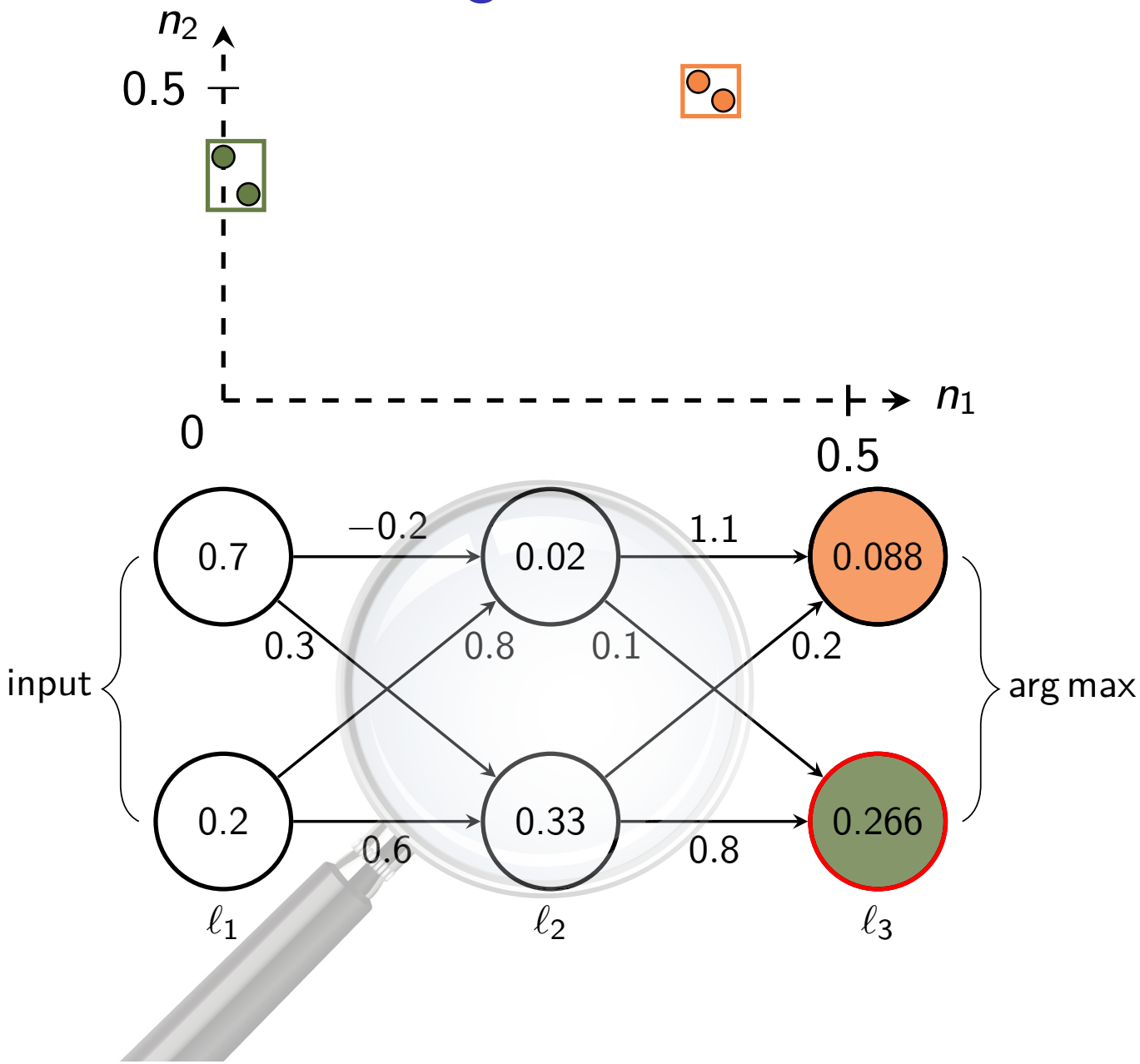
Constructing an interval abstraction



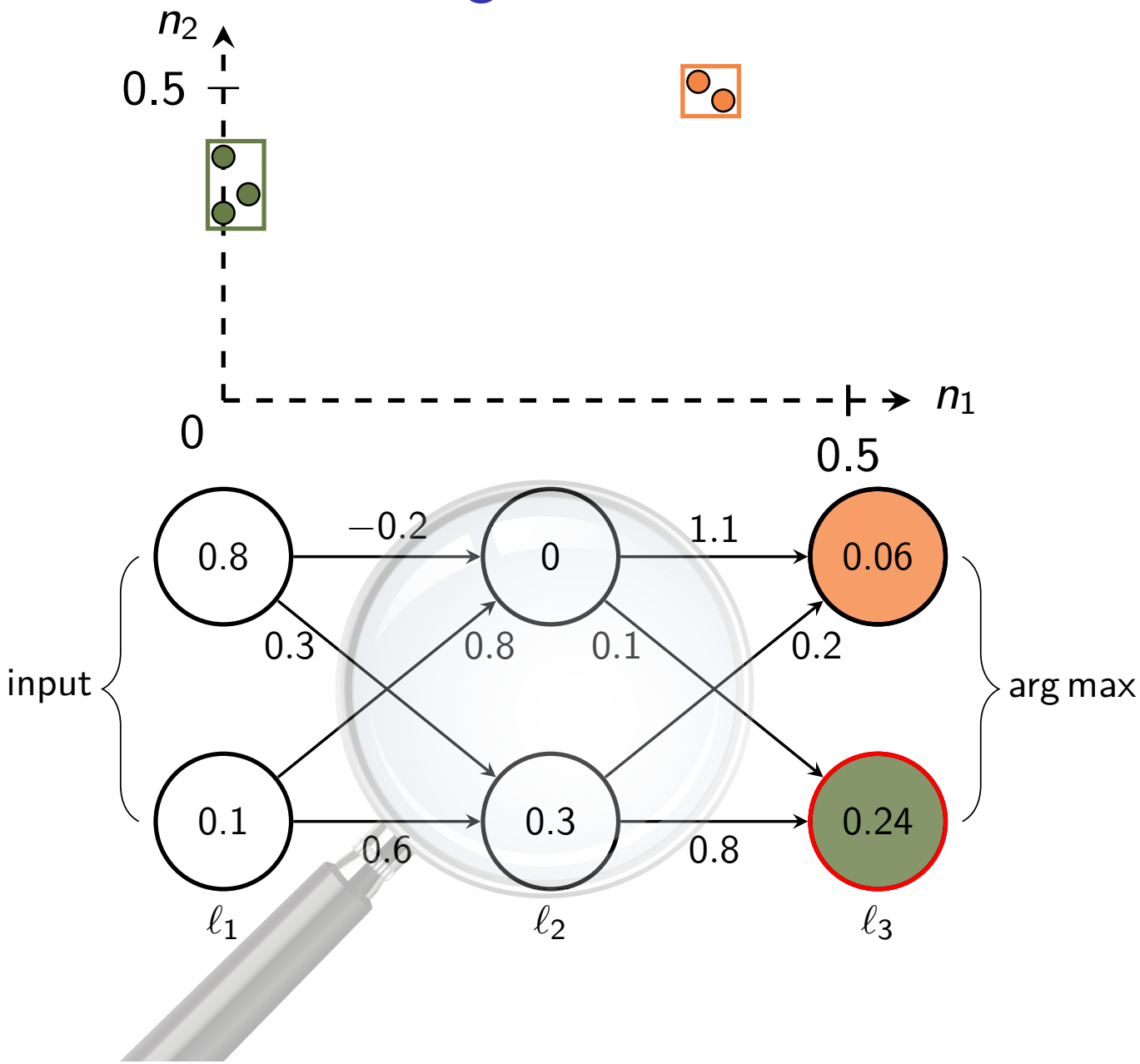
Constructing an interval abstraction



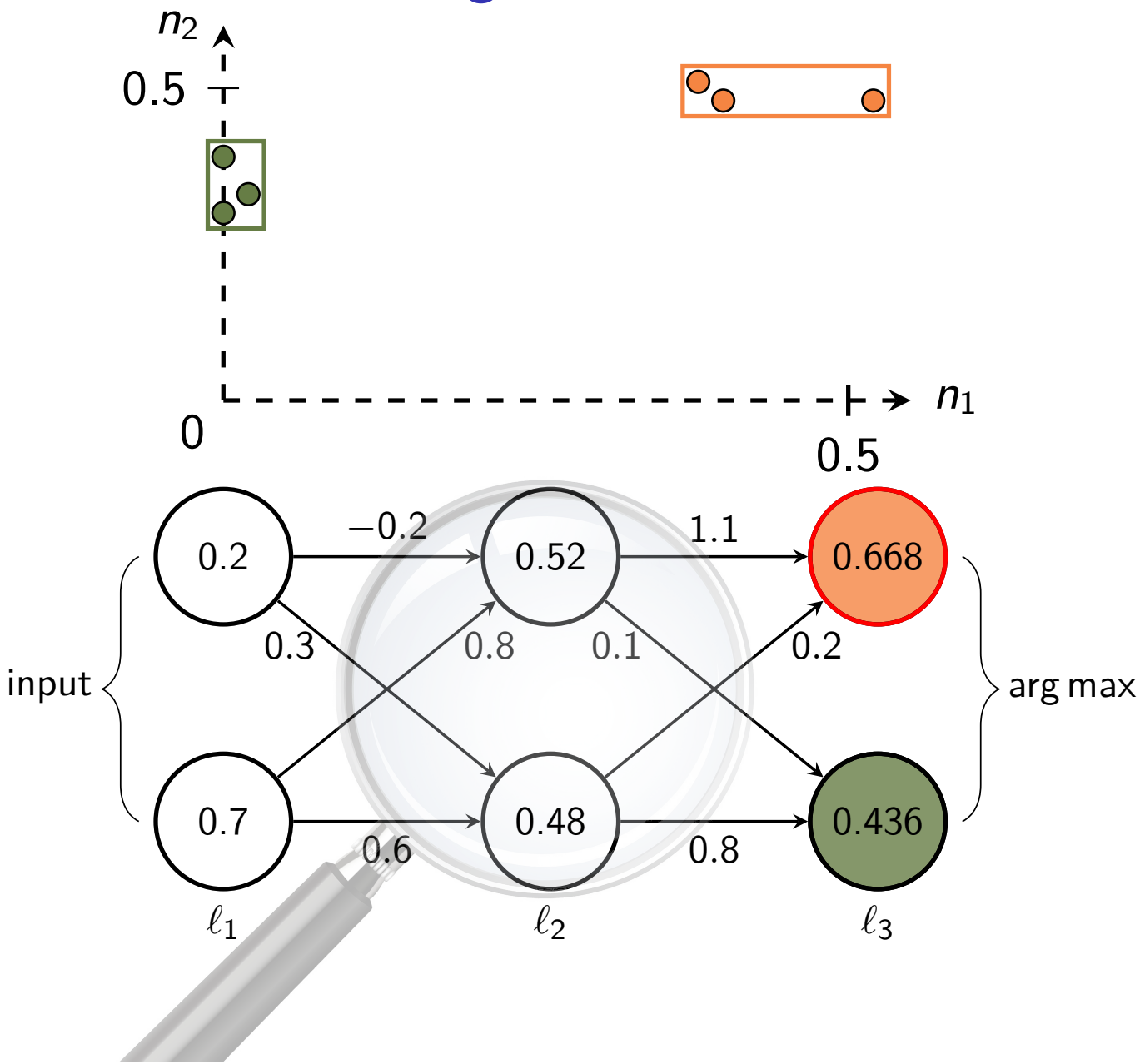
Constructing an interval abstraction



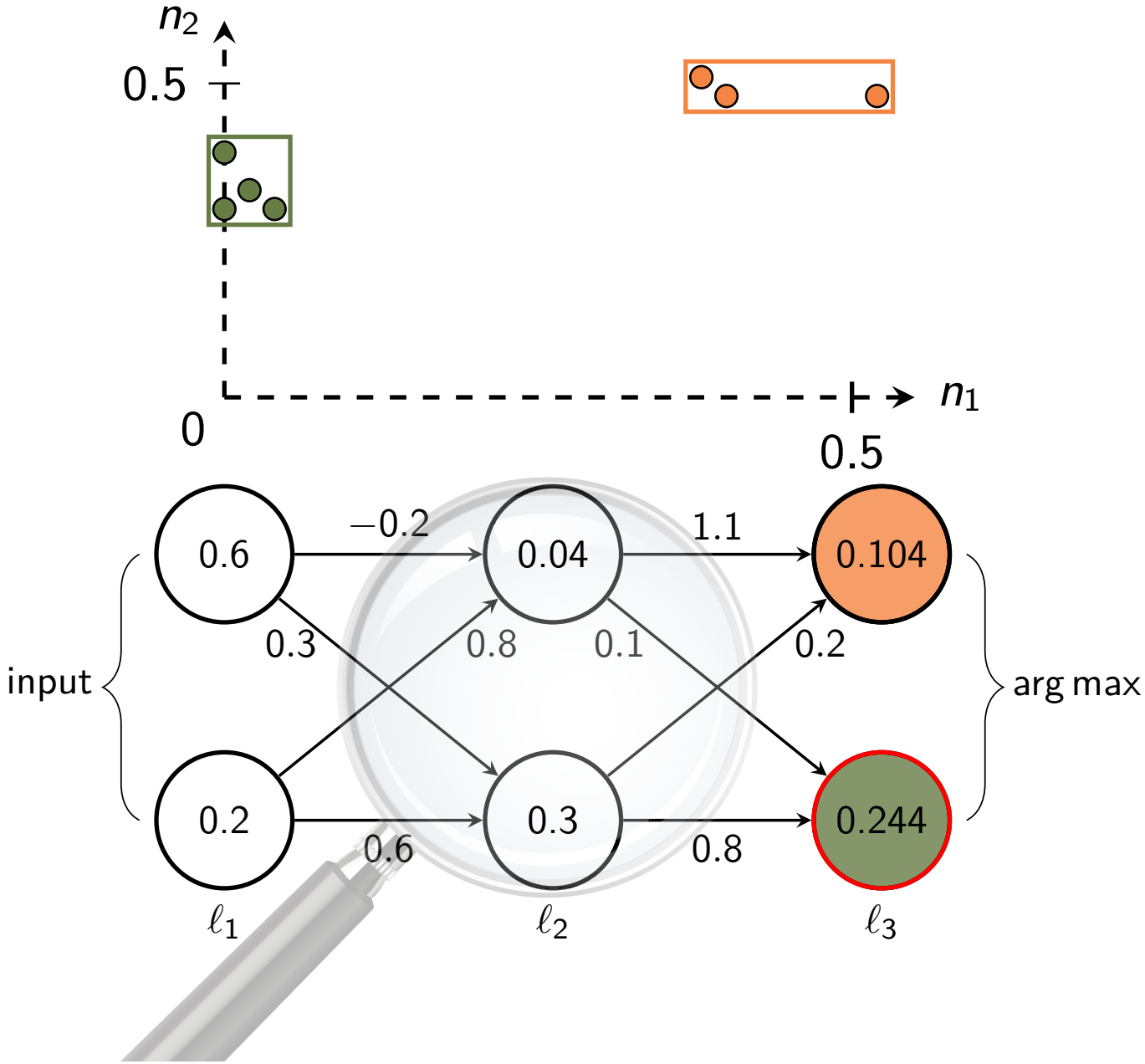
Constructing an interval abstraction



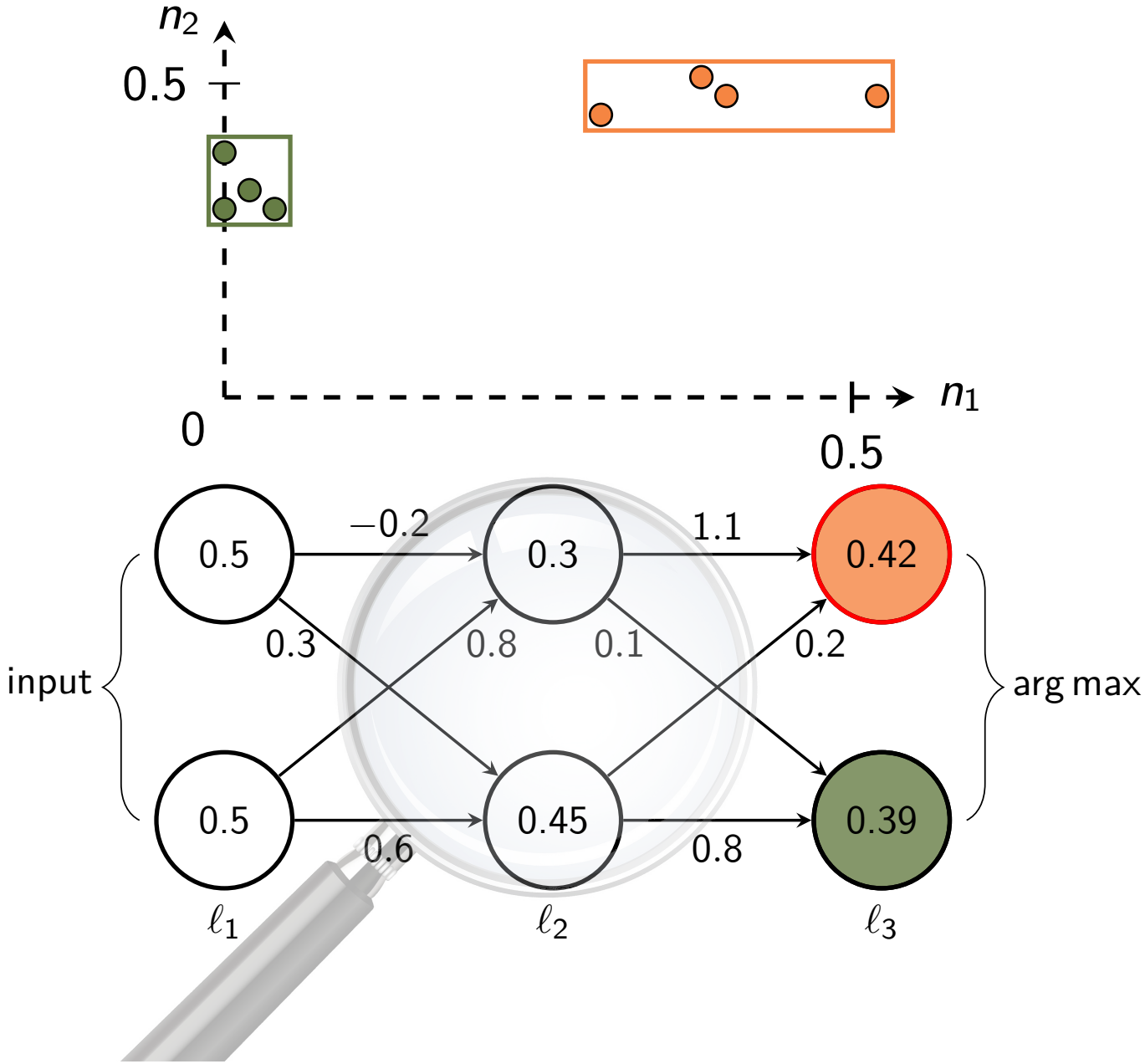
Constructing an interval abstraction



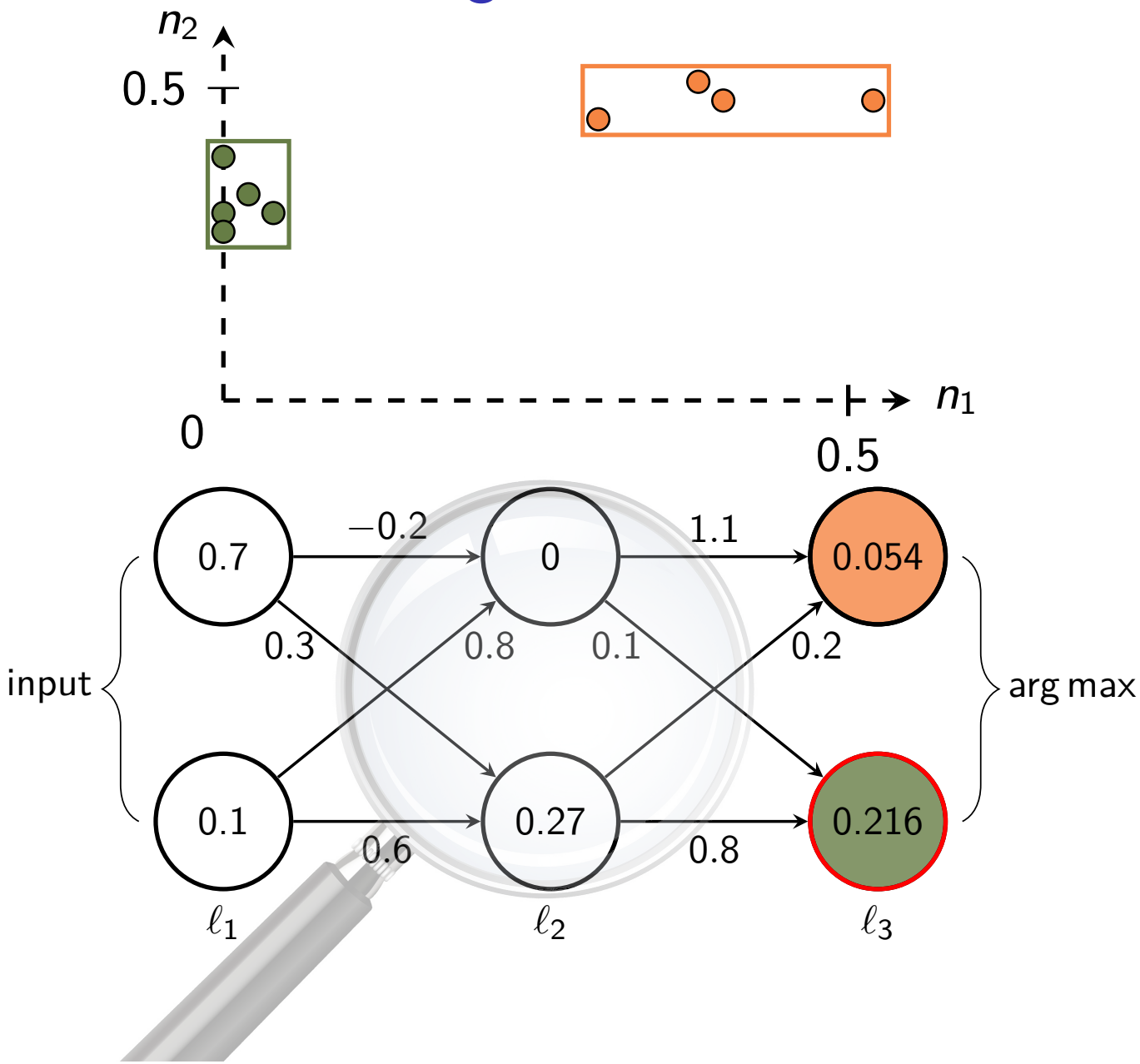
Constructing an interval abstraction



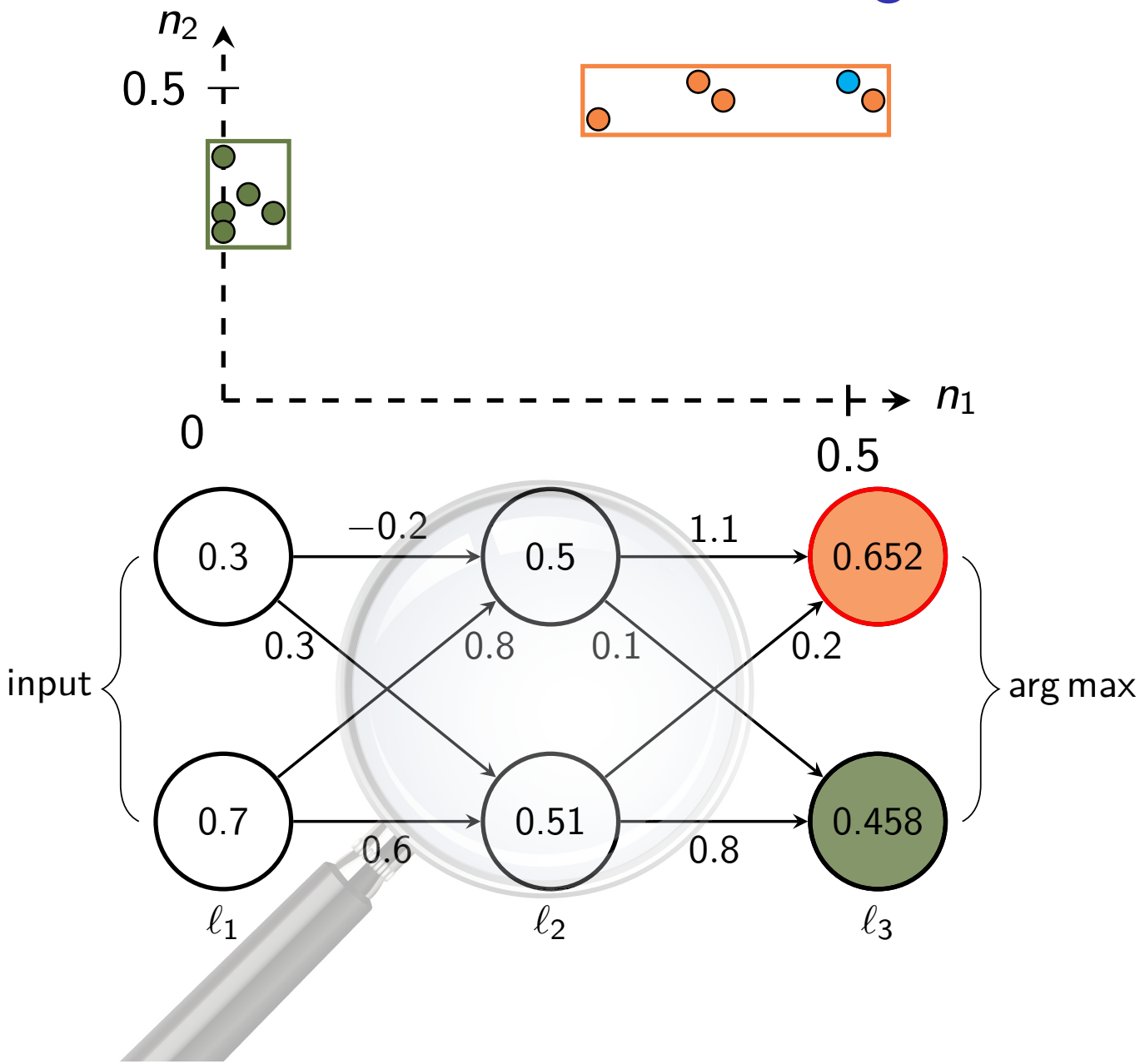
Constructing an interval abstraction



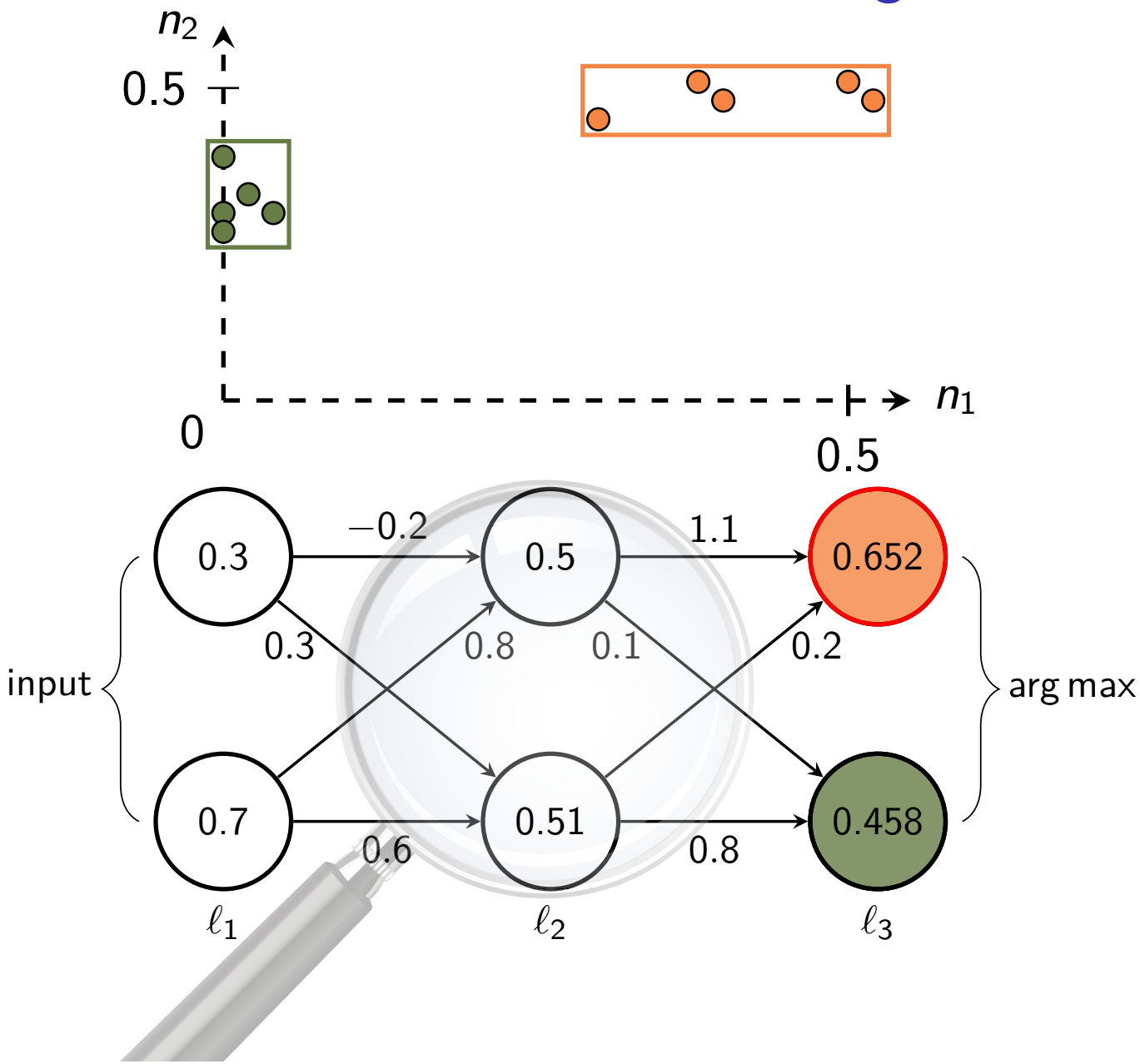
Constructing an interval abstraction



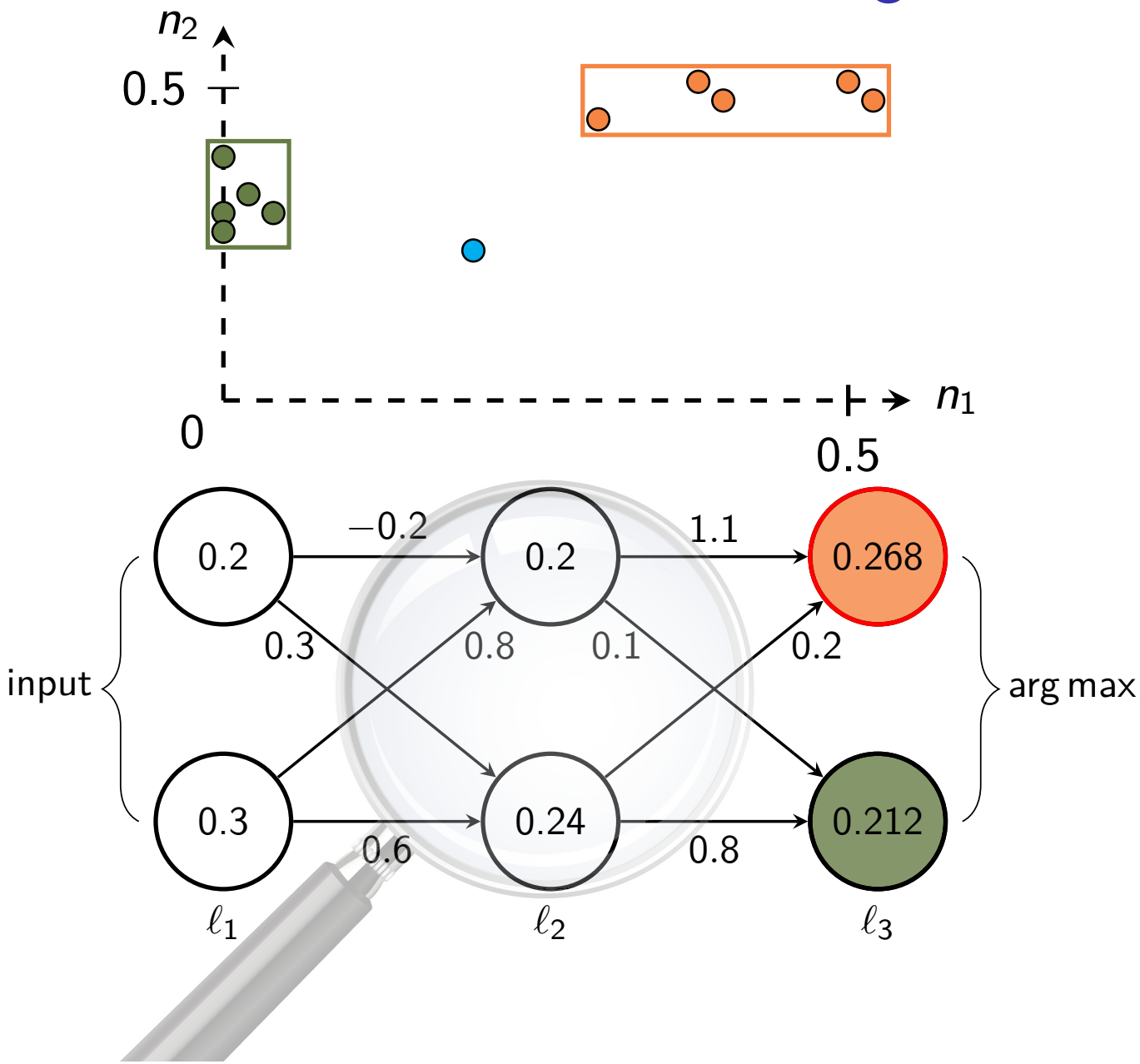
Runtime monitoring



Runtime monitoring



Runtime monitoring



Overview

Preliminaries

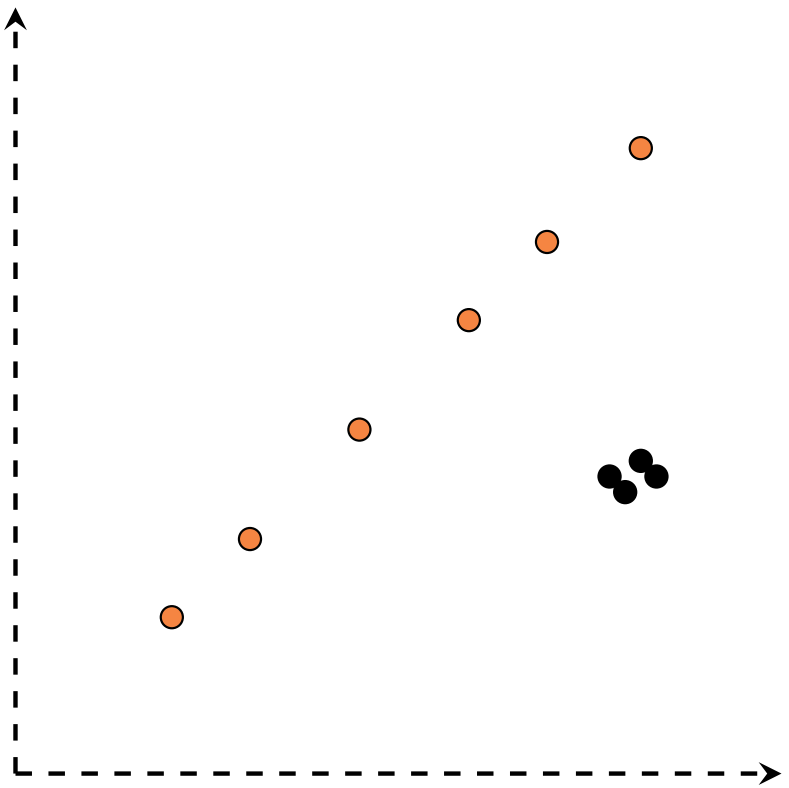
Interval abstraction (basic idea)

Box abstraction (extension to clustered data)

Evaluation

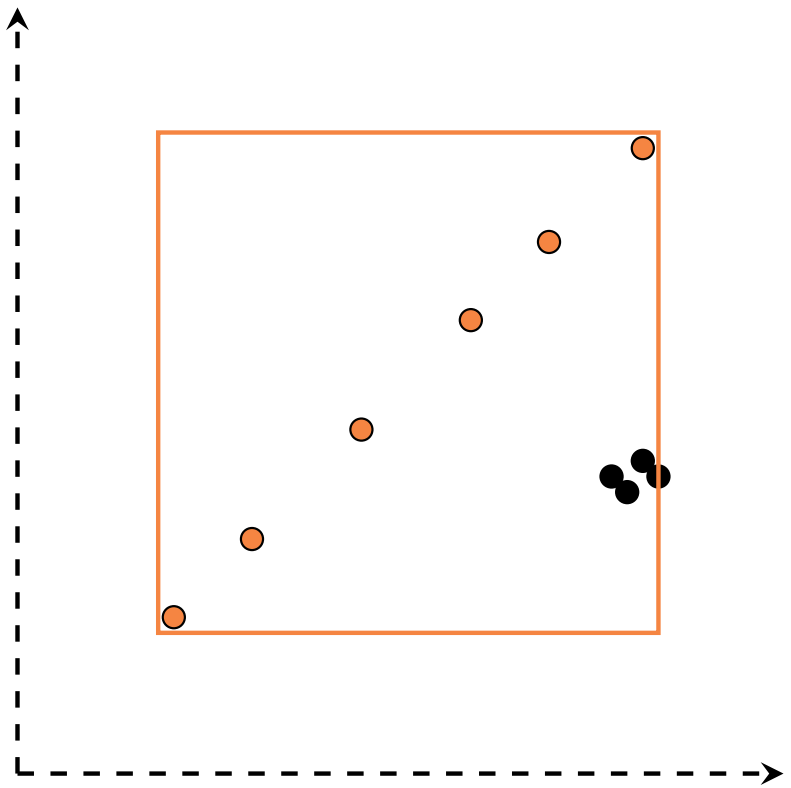
Problems with interval abstraction

- A box may be **too coarse**



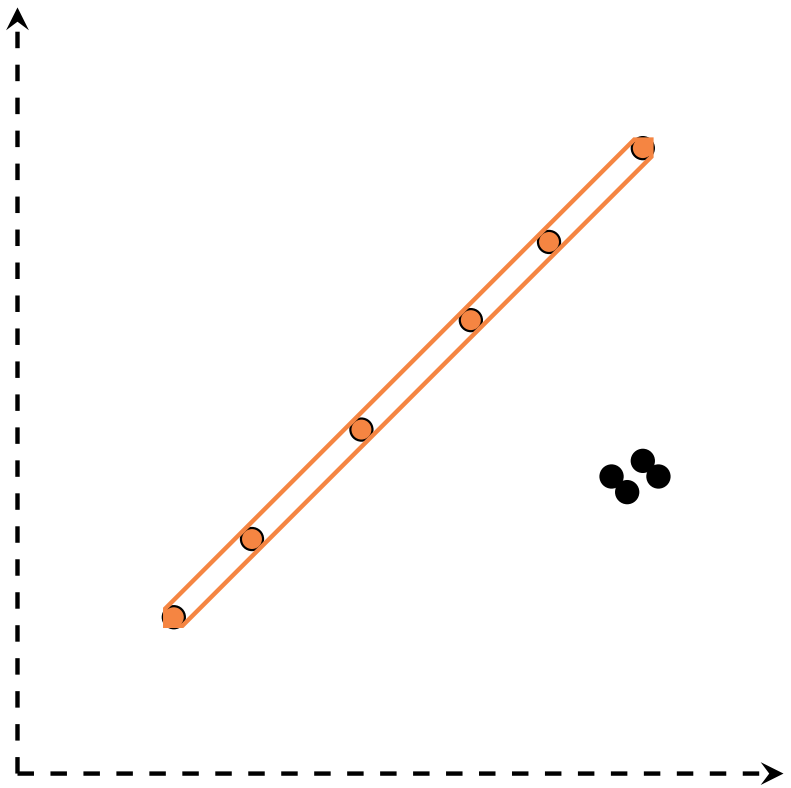
Problems with interval abstraction

- A box may be **too coarse**



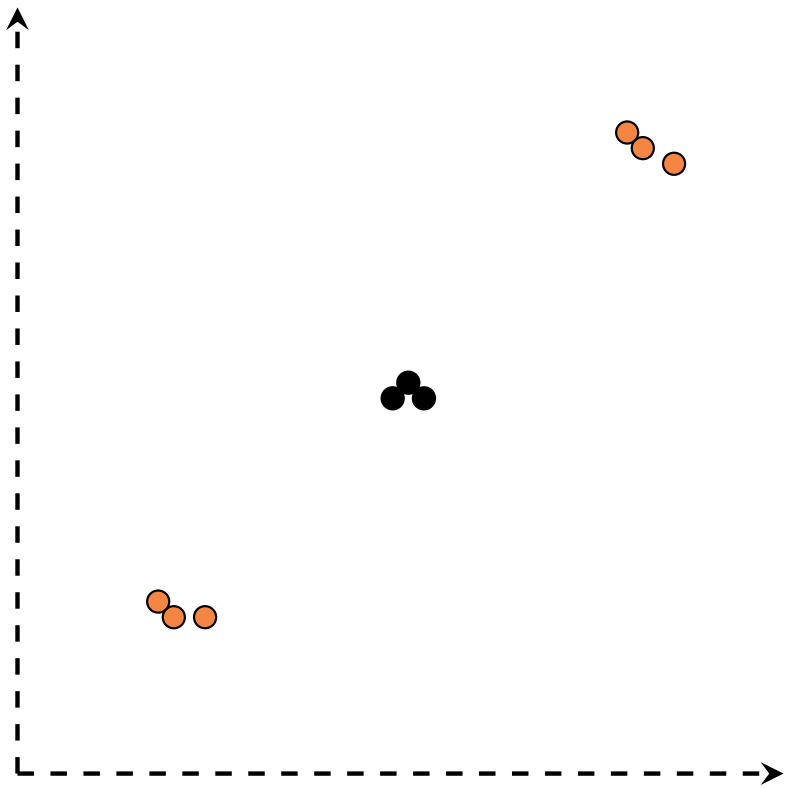
Problems with interval abstraction

- **More precision** with, e.g., an octagon (**more expensive**)



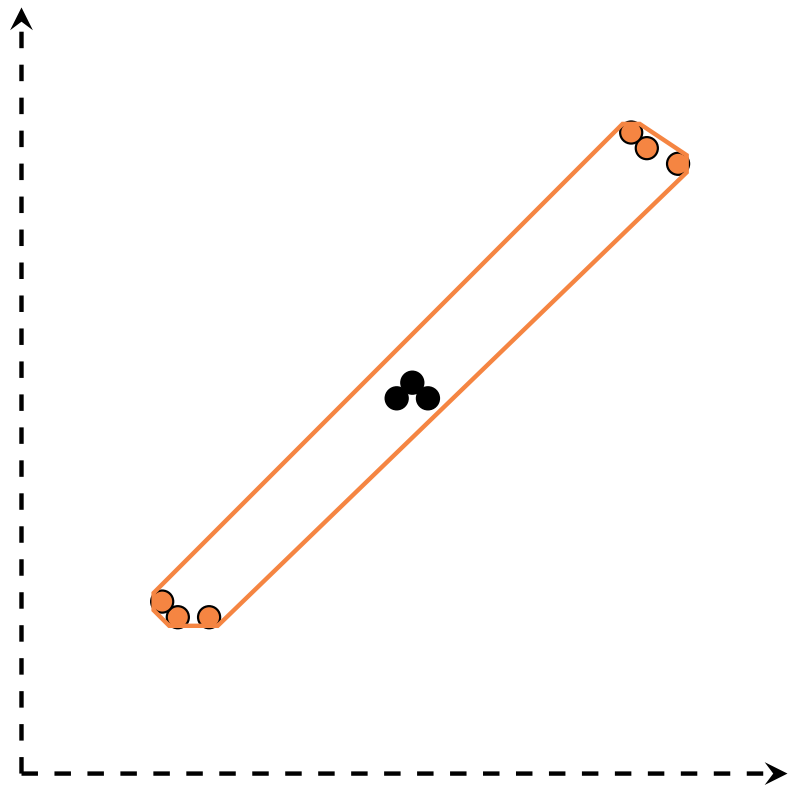
Problems with interval abstraction

- **Convex** abstractions are not always suitable



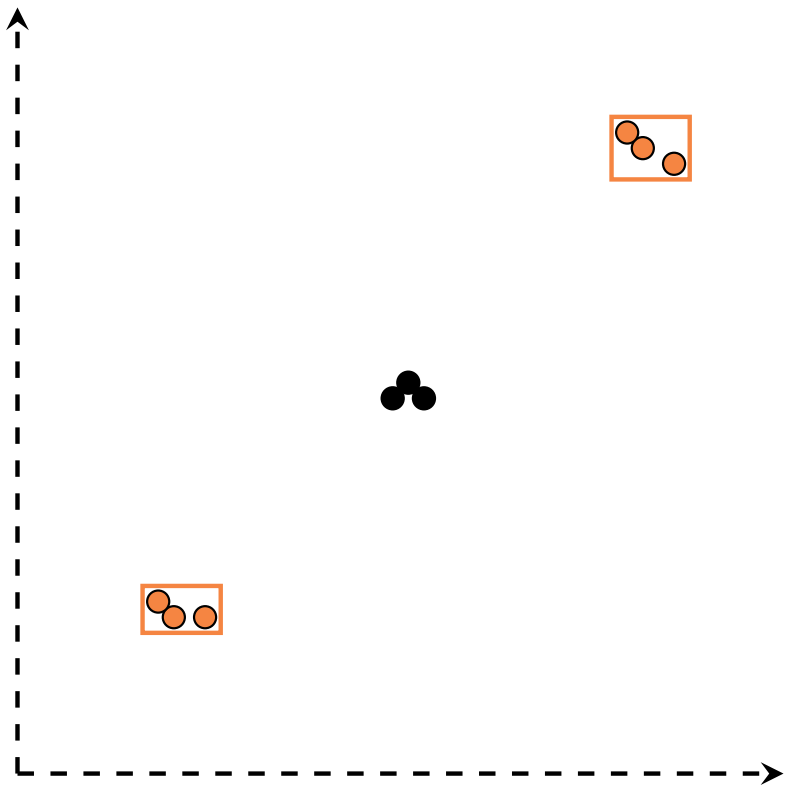
Problems with interval abstraction

- **Convex** abstractions are not always suitable



Problems with interval abstraction

- Trade-off between precision and efficiency: **union of boxes**
- We use **clustering** to split into a suitable number of subsets



Properties of abstraction-based monitor

- Monitor = **supervisor** for decision making
- Monitor **always** accepts **training** inputs
- Assuming **convergence**:
  Monitor **always** accepts **correctly classified** inputs

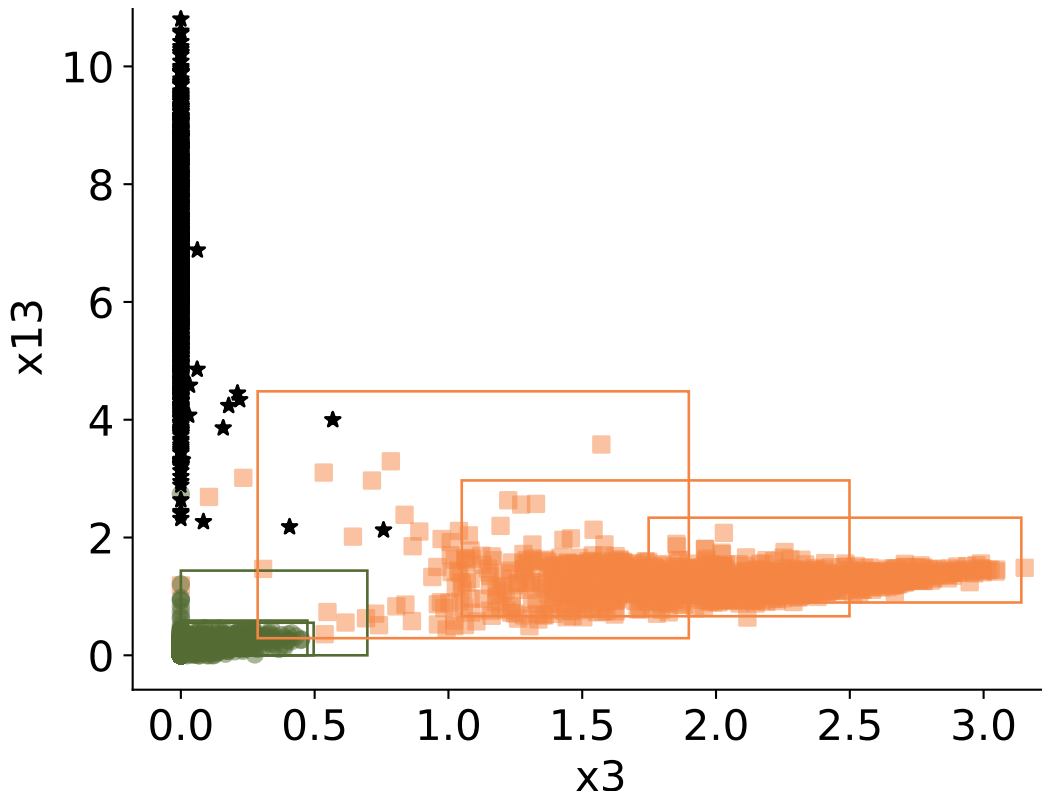
  (i.e., **rejected** inputs are **always misclassified**)

Properties of box abstraction

- **Precision** (increases with the number of boxes)
- **Efficiency** (decreases with the number of boxes)
  - Per box: **linear** in the number of (watched) neurons
  - **Data independent** at runtime

Example

- Sample projection on MNIST dataset on last hidden layer (40 neurons) with two known classes (**green**, **orange**; three boxes each) and one novel class (**black**)



Overview

Preliminaries

Interval abstraction (basic idea)

Box abstraction (extension to clustered data)

Evaluation

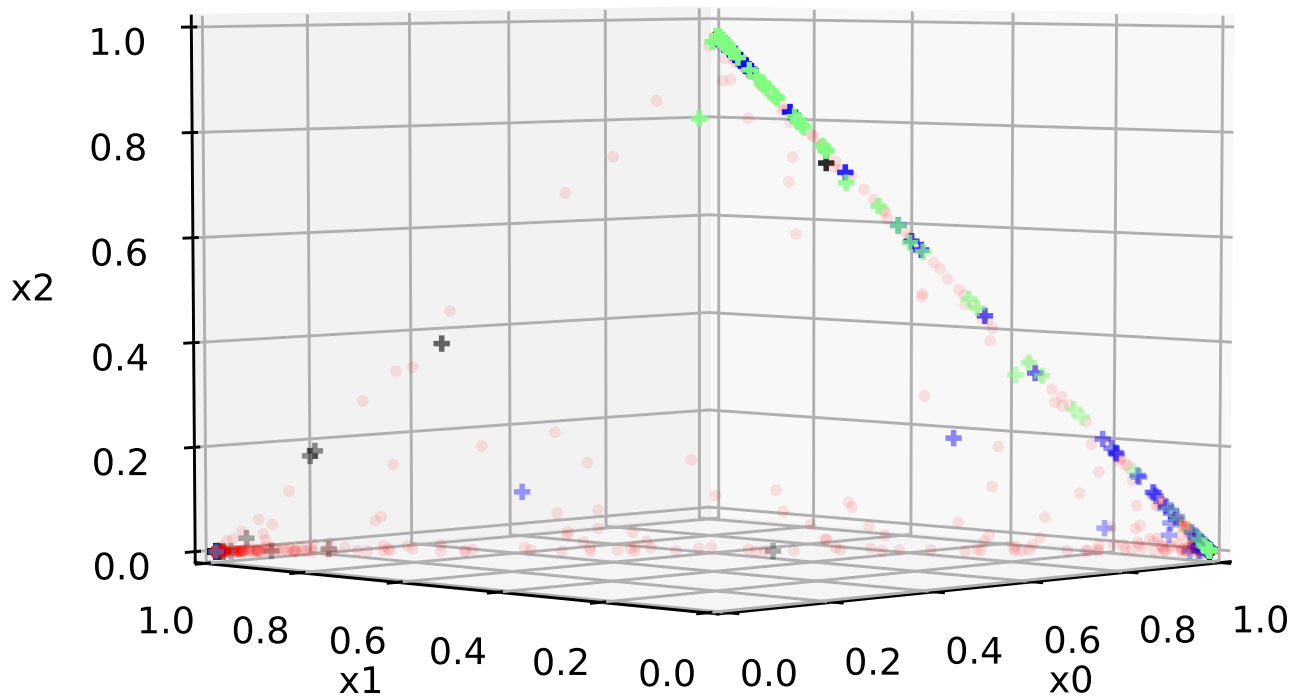
Setup

- Neural networks:
  NN_1: 8 hidden layers with 8280 neurons
  NN_2: 10 hidden layers with 77884 neurons
- Datasets: MNIST, F_MNIST (Fashion MNIST), CIFAR-10, GTSRB (German Traffic Signs)
- Train networks on $k \in [2, n)$ known classes (n = total number)

Dataset	Classes	Inputs train / test	Network	Epochs	Acc. train / test %
MNIST	10	60,000 / 10,000	NN_1	10	99/99
F_MNIST	10	60,000 / 10,000	NN_1	30	$98-99/91-99$
CIFAR-10	10	60,000 / 10,000	NN_2	200	$99/71-95$
GTSRB	43	39,209 / 12,630	NN_2	10	$98-99/88-97$

Softmax-score approach [1]

- Reject outputs with **confidence** $< \alpha$ for selected class
- Values at output layer for F_MNIST benchmark with three known classes (**blue**, **black**, **green**) and one novel class (**red**); 1,000 inputs per class



[1] Hendrycks and Gimpel. *ICLR*. 2017.

Boolean abstraction [1]

- Monitor with **union of bit vectors**

  Neuron abstraction: $x \mapsto \begin{cases} 0 & x \leq 0 \\ 1 & \text{otherwise} \end{cases}$

  Example: $(0.1, 0, 2.1, -1.5, 3.3) \rightsquigarrow (1, 0, 1, 0, 1)$

- Designed for ReLU activation functions
- **Expensive** $\rightarrow$ only used in NN_1 in last layer (40 neurons)

[1] Cheng, Nührenberg, and Yasuoka. *DATE*. 2019.

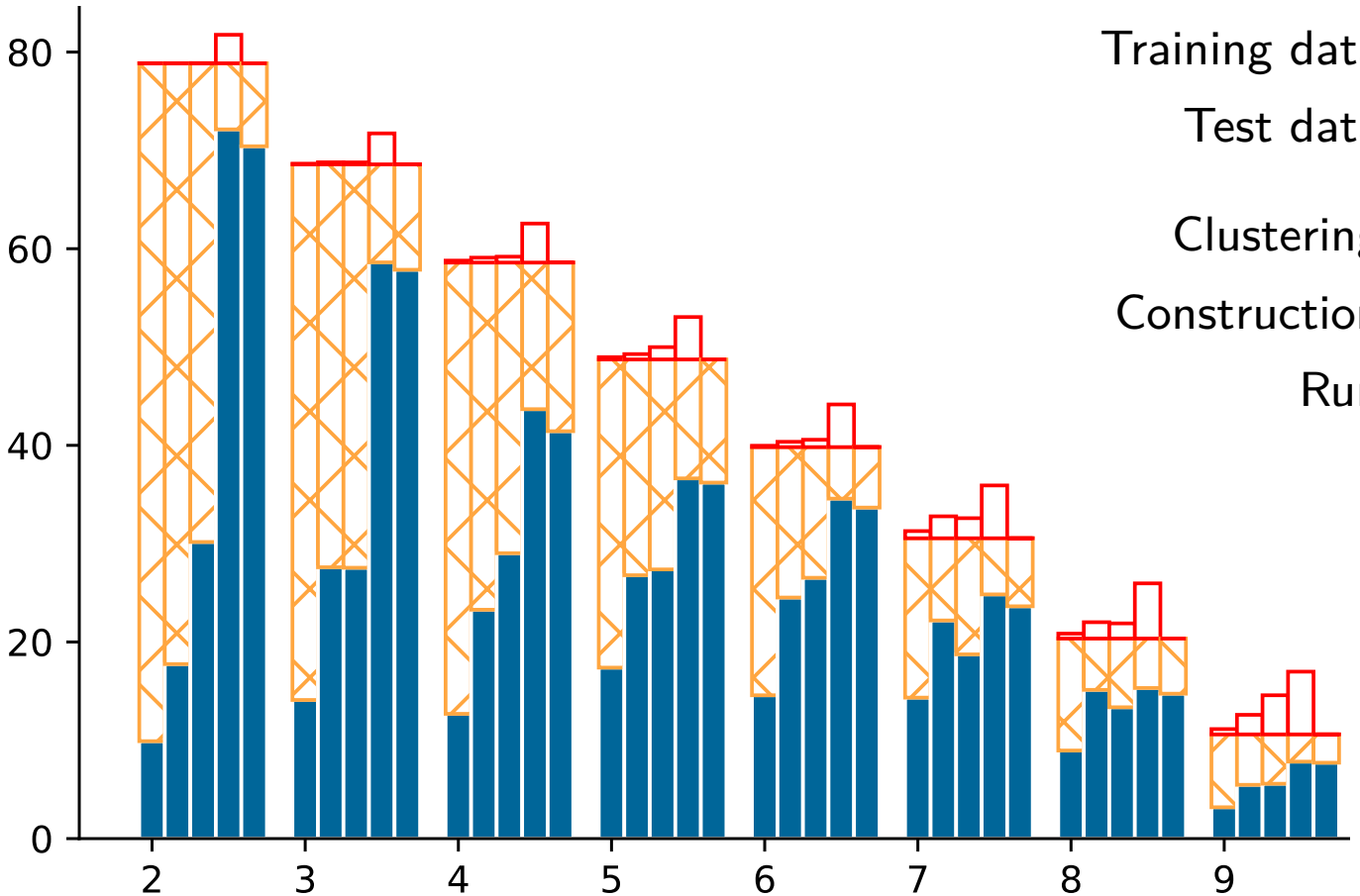
Setup of box abstraction

- Monitored layers:

Dataset	Layers
MNIST	last four (283 neurons)
F_MNIST	last five (602 neurons)
CIFAR-10	last four (826 neurons)
GTSRB	last hidden (84 neurons)

- Additional instance trained on test dataset

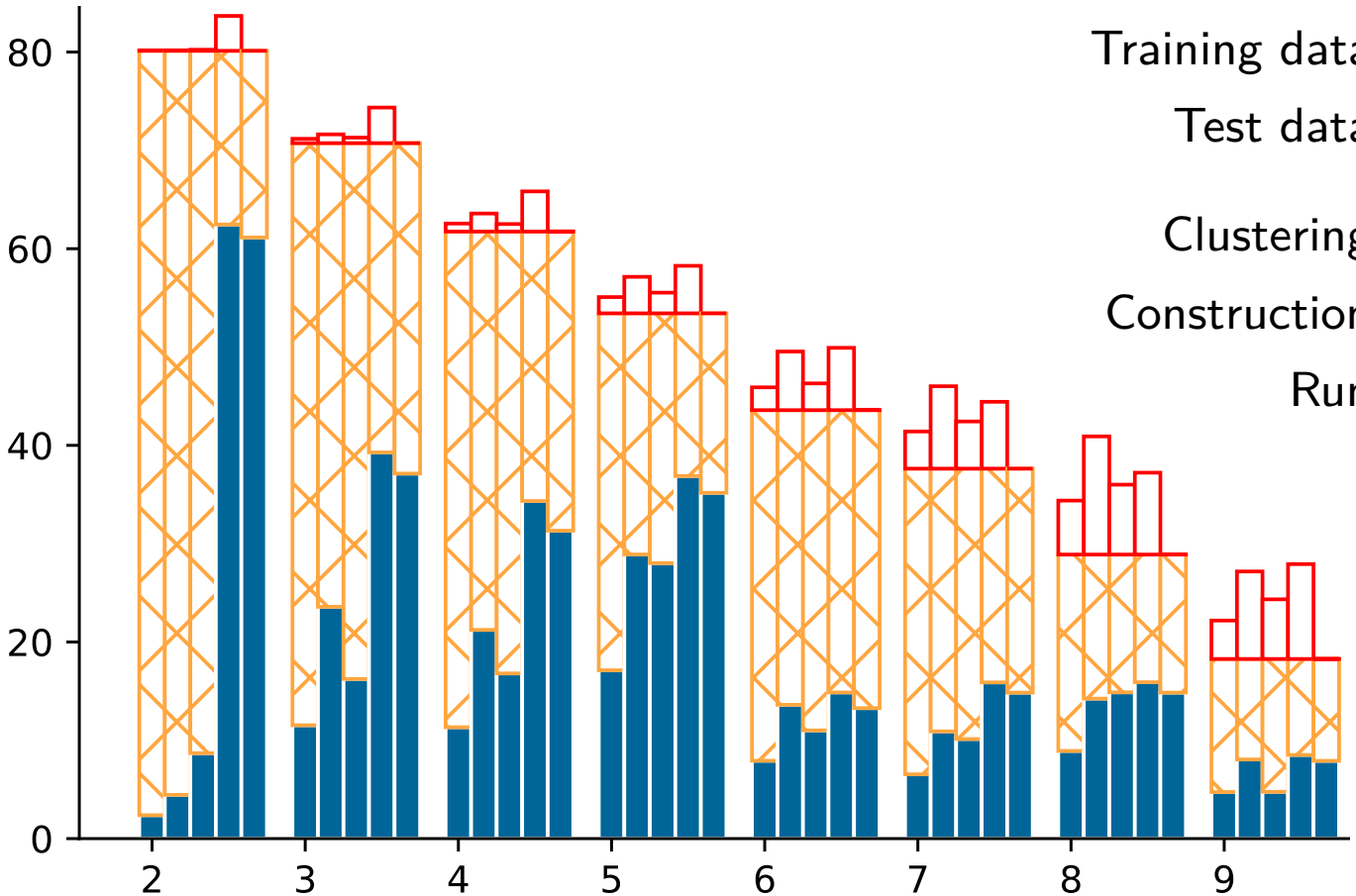
MNIST dataset (varying known classes)



Training data	60,000
Test data	10,000
Clustering	158.8 s
Construction	7.6 s
Run	2.1 s

- Legend: detected, undetected, false warning
- Bars: softmax score ($\alpha = 0.9/0.99$), Boolean abstraction, box abstraction (w/o and w/ training on test dataset)

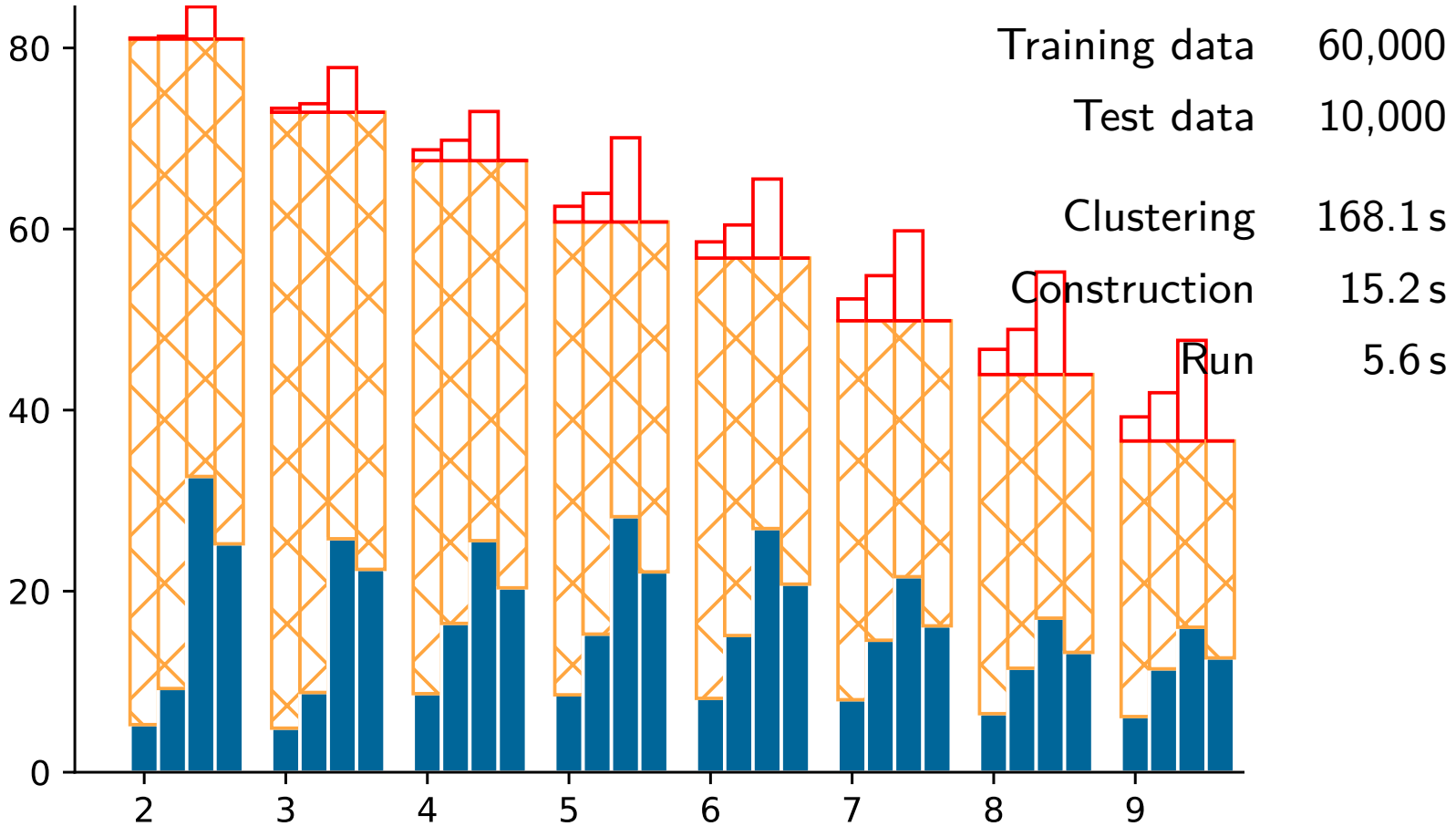
F_MNIST dataset (varying known classes)



Training data	60,000
Test data	10,000
Clustering	198.1 s
Construction	14.3 s
Run	4.4 s

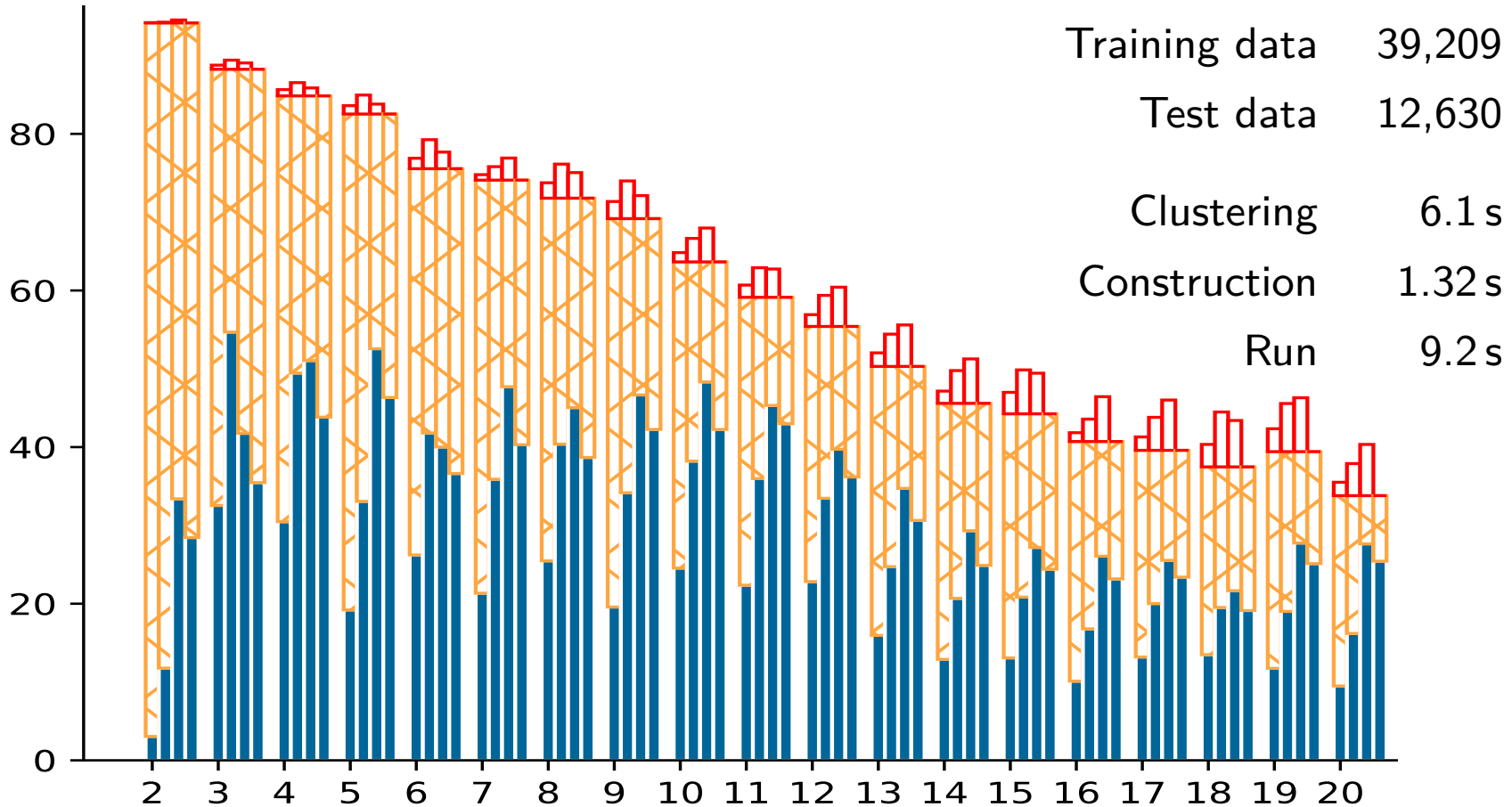
- Legend: detected, undetected, false warning
- Bars: softmax score ($\alpha = 0.9/0.99$), Boolean abstraction, box abstraction (w/o and w/ training on test dataset)

CIFAR-10 dataset (varying known classes)



- Legend: detected, undetected, false warning
- Bars: softmax score ($\alpha = 0.9/0.99$),
  box abstraction (w/o and w/ training on test dataset)

GTSRB dataset (varying known classes)



- Legend: detected, undetected, false warning
- Bars: softmax score ($\alpha = 0.9/0.99$),
  box abstraction (w/o and w/ training on test dataset)

Other experiments

- Monitoring different layers
- Other geometric abstractions
- Increasing box sizes

Summary

- **Novelty detection** for **neural-network classifiers**
- Compare **internal behavior** at training time and runtime
- **Abstract neuron values** with **concise boxes**
- At training time: create **box abstraction**
- At runtime: check if **neuron values** belong to abstraction
- **Precise**
- **Efficient** (< 1 ms per input)

Future work

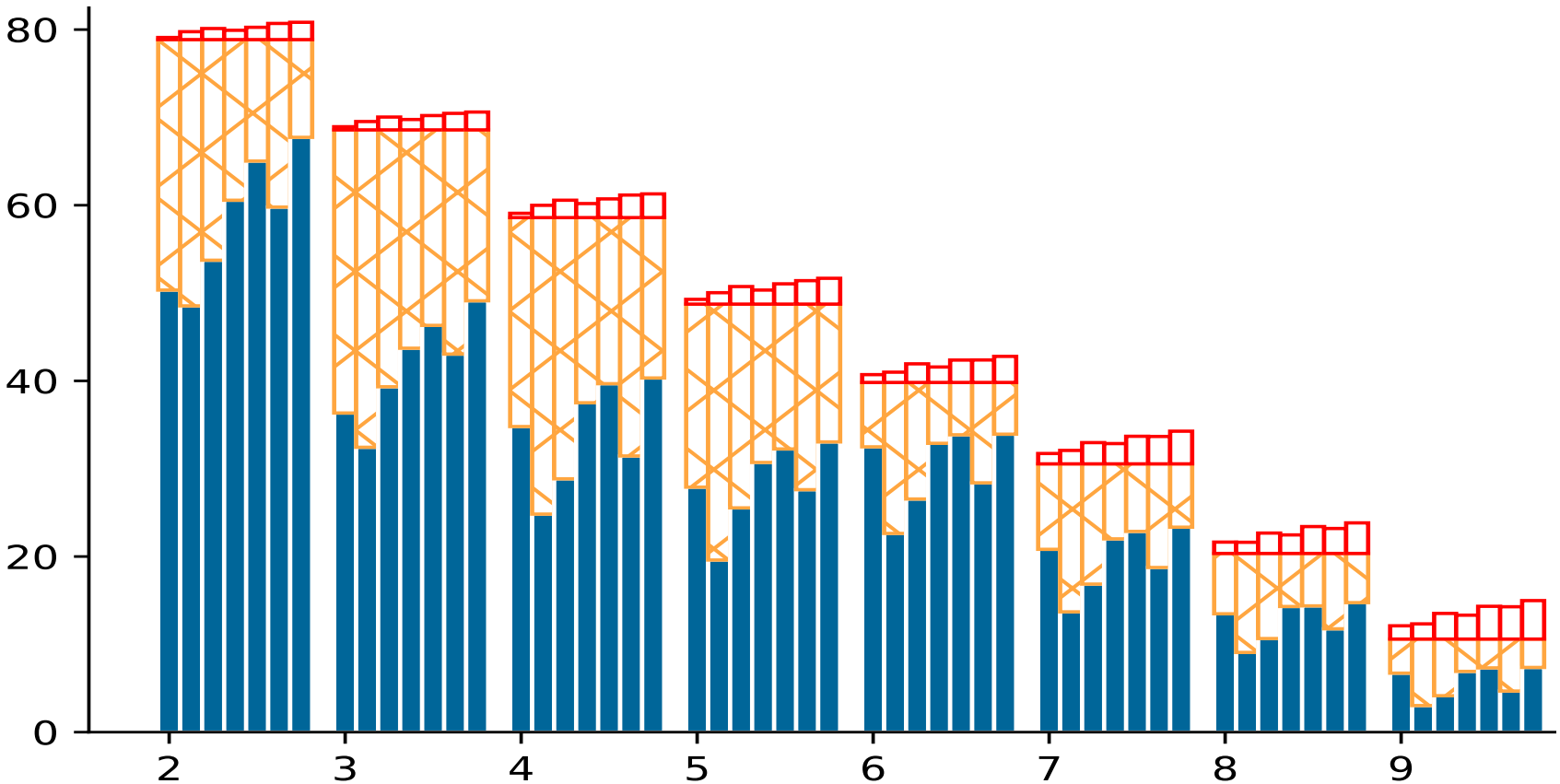
- Evaluation on **larger neural networks and datasets**

  Scalability issues expected; countermeasures:
  - Current bottleneck: **clustering** $\rightsquigarrow$ **online clustering**
  - **Fast box selection**: **hierarchy of boxes** (quadtrees)
  - **Selective choice** of watched layers/neurons per class
- Enable rejection of **points inside the boxes**
  - Assign **confidence** to boxes
  - Fall back to **other approaches** for those cases
- Use the **learned box abstraction** for other purposes:
  - Identify **outliers** to make network more robust
  - Identify **features** in the network
- **Recurrent neural networks**

Monitoring different layers

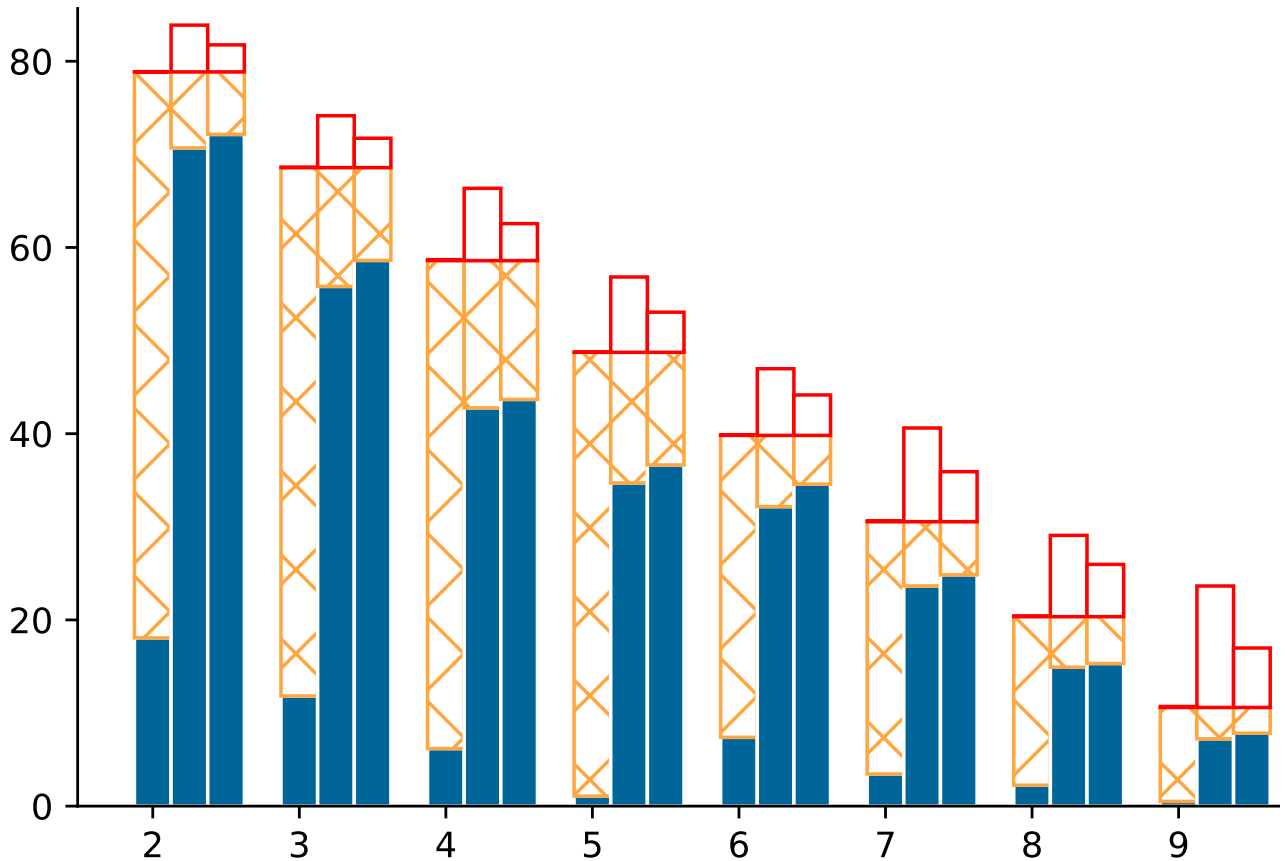
- All combinations of the last three layers (ℓ_x, ℓ_y, ℓ_z) (on MNIST); order of the bars from left to right:

$$\ell_z \quad \ell_y \quad \ell_x \quad \ell_y, \ell_z \quad \ell_x, \ell_z \quad \ell_x, \ell_y \quad \ell_x, \ell_y, \ell_z$$



Other geometric abstractions

- Ball abstraction and octagon abstraction (last hidden layer) vs. box abstraction (last four layers as before) (on MNIST)



Increasing box sizes

- Increase boxes by a factor (x axis)
- MNIST instance with five known and unknown classes