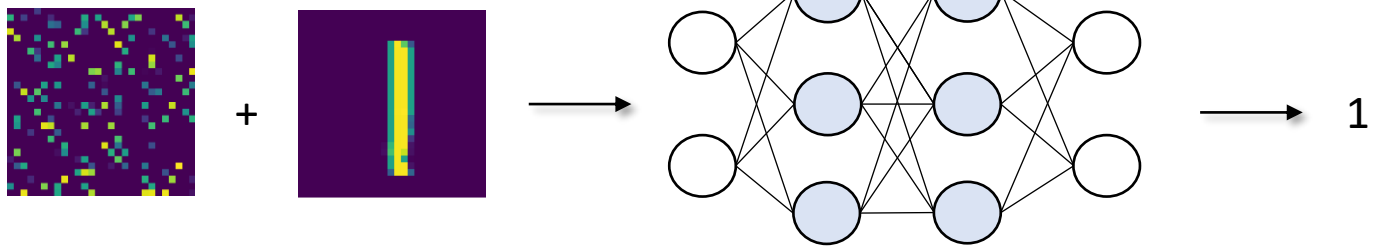
Towards Reliable Neural Specifications

Chuqin Geng* · Van Nham Le* · Xiaojie Xu · Zhaoyue Wang · Arie Gurfinkel · Xujie Si

The robustness verification problem

Verification Goal: The perturbation to an input, bounded within some L_∞-ball, should yield the same prediction result as the original input.

$$\forall \vec{\delta}\ s.t.\ max\vec{\delta} \leq \epsilon$$
$$(or\ \|\vec{\delta}\|_\infty \leq \epsilon)$$

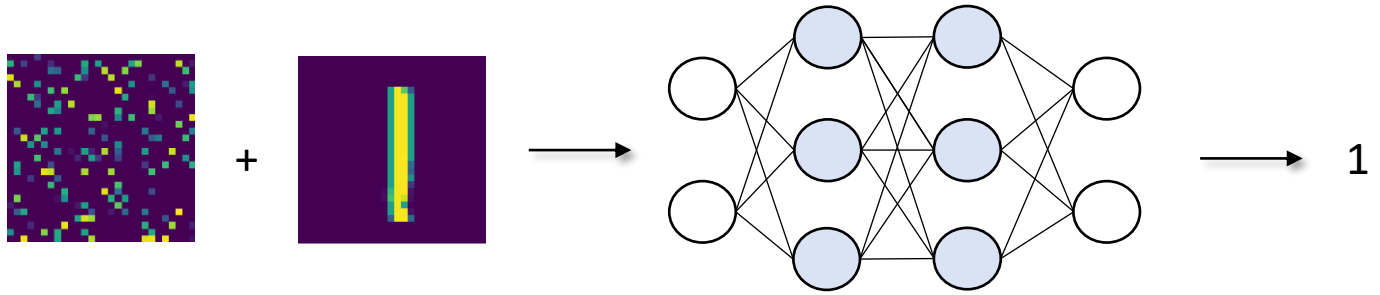


Research target: Build better tools to verify bigger epsilon on bigger networks.

The robustness verification problem

Verification Goal: The perturbation to an input, bounded within some L_∞-ball, should yield the same prediction result as the original input.

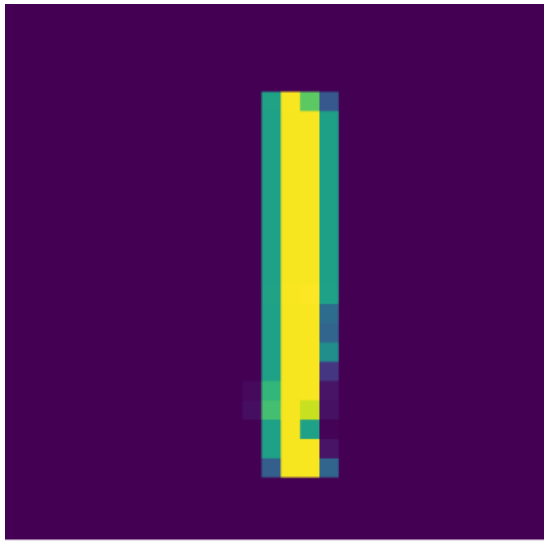
$$\forall \vec{\delta}\ s.t.\ max\vec{\delta} \leq \epsilon$$
$$(or\ \|\vec{\delta}\|_\infty \leq \epsilon)$$



Research target: Build better tools to verify bigger epsilon on bigger networks.

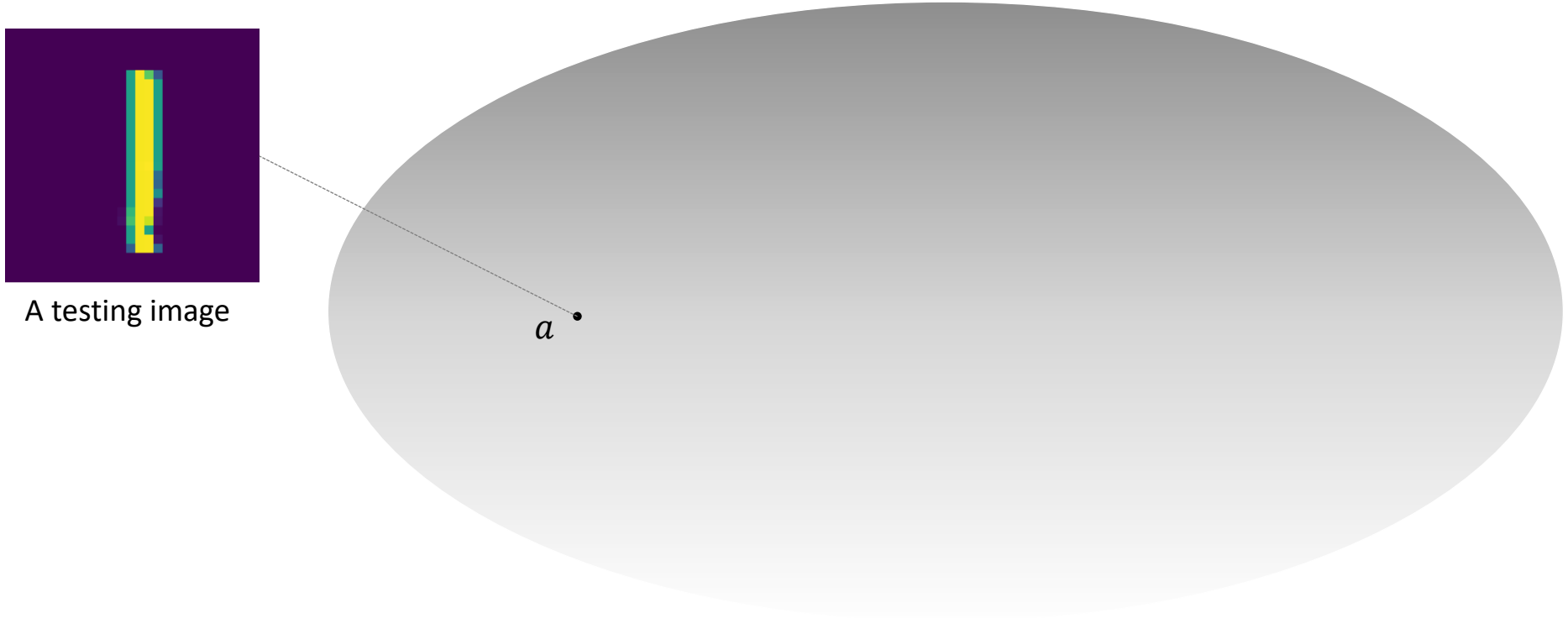
However, it is important to acknowledge a limitation regarding the L_∞ -ball specifications.

The L_∞-ball specifications



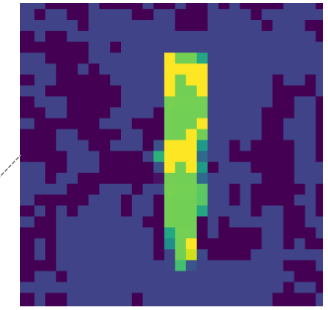
a) A testing image from MNIST, classified as 1

The L_∞-ball specifications

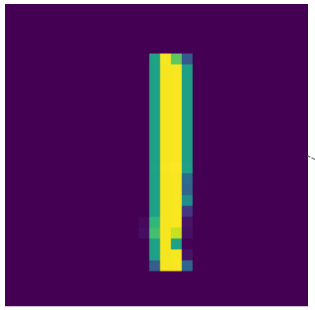


The input space of MNIST

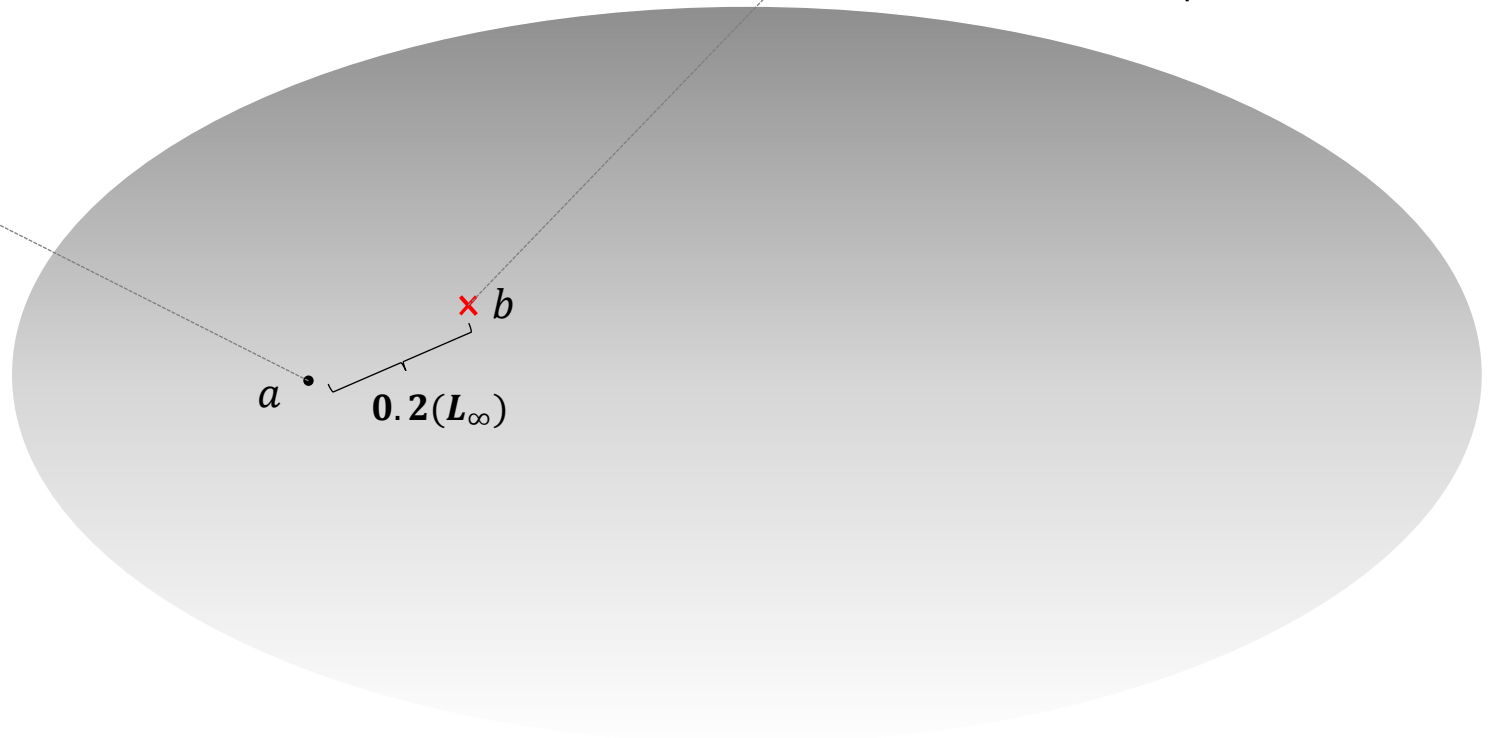
The L_∞-ball specifications



An adv. example

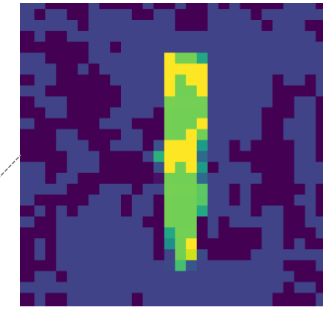


A testing image

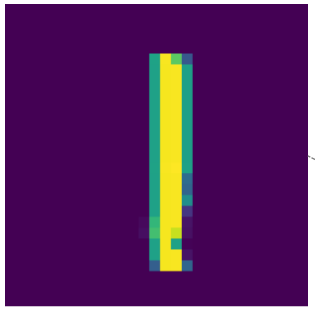


The input space

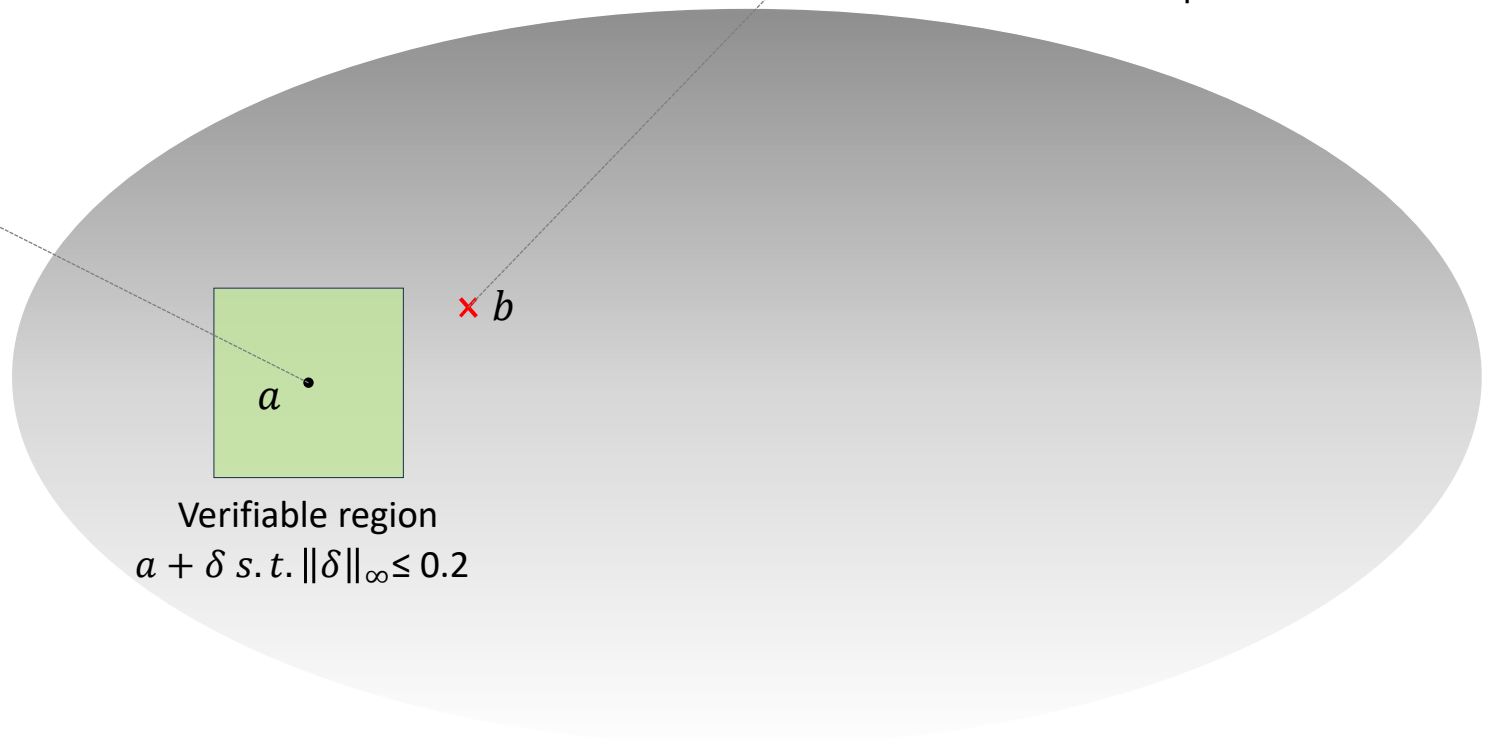
The L_∞-ball specifications



An adv. example



A testing image

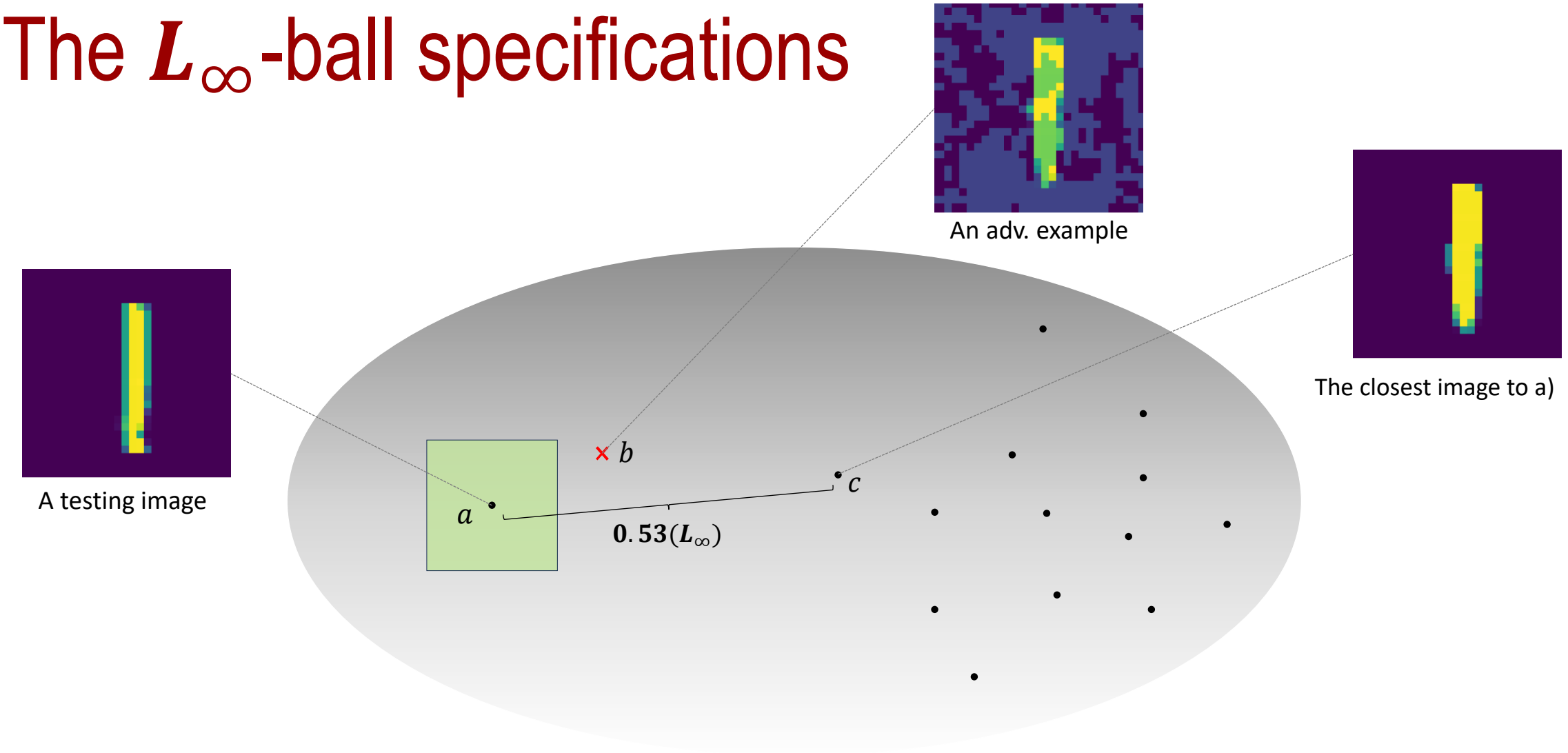


Verifiable region

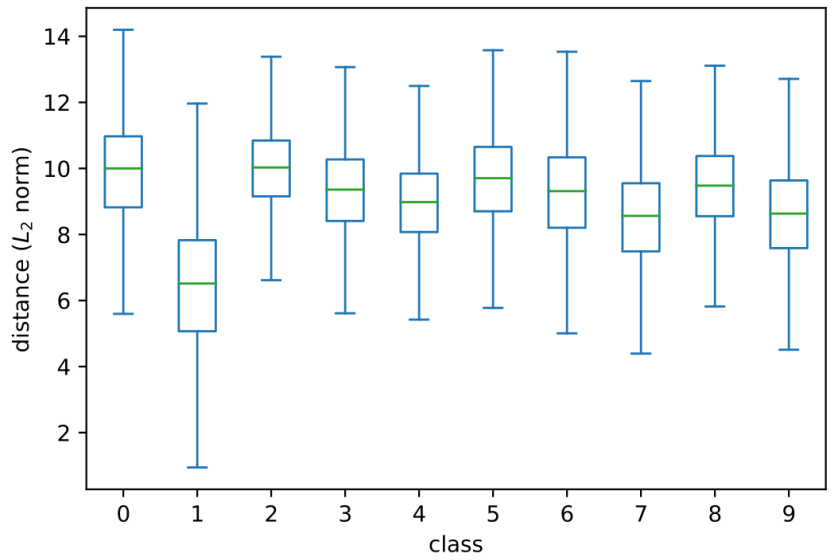
$a + \delta \; s.t. \; \|\delta\|_\infty \leq 0.2$

The input space

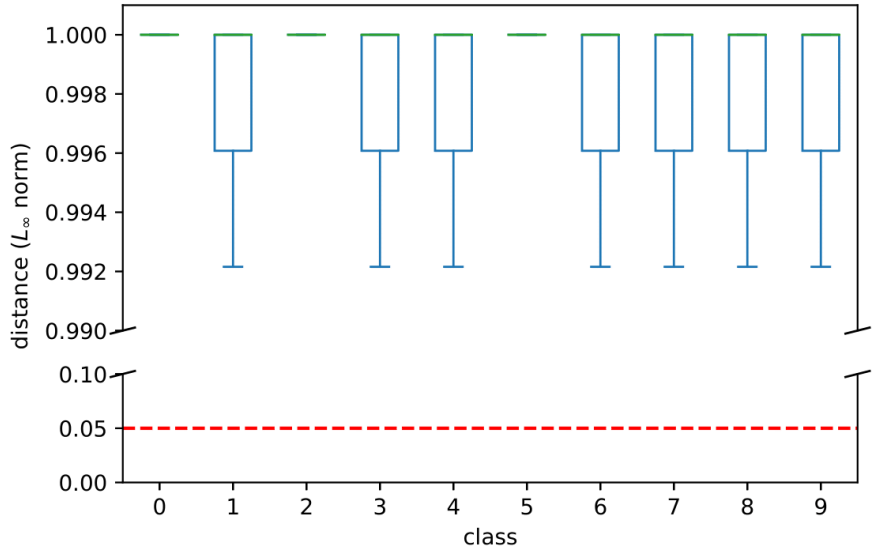
The L_∞-ball specifications



The L_∞-ball specifications

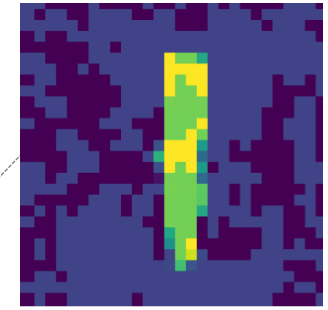


The distribution of L_2-norms between any two image with the label. Higher → further away.

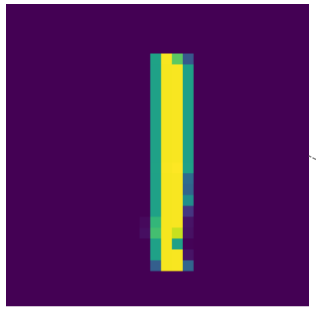


The distribution of L_∞-norms between any two images with the same label. The red line is drawn at 0.05 – the largest ϵ used in the VNNCOMP 2021.

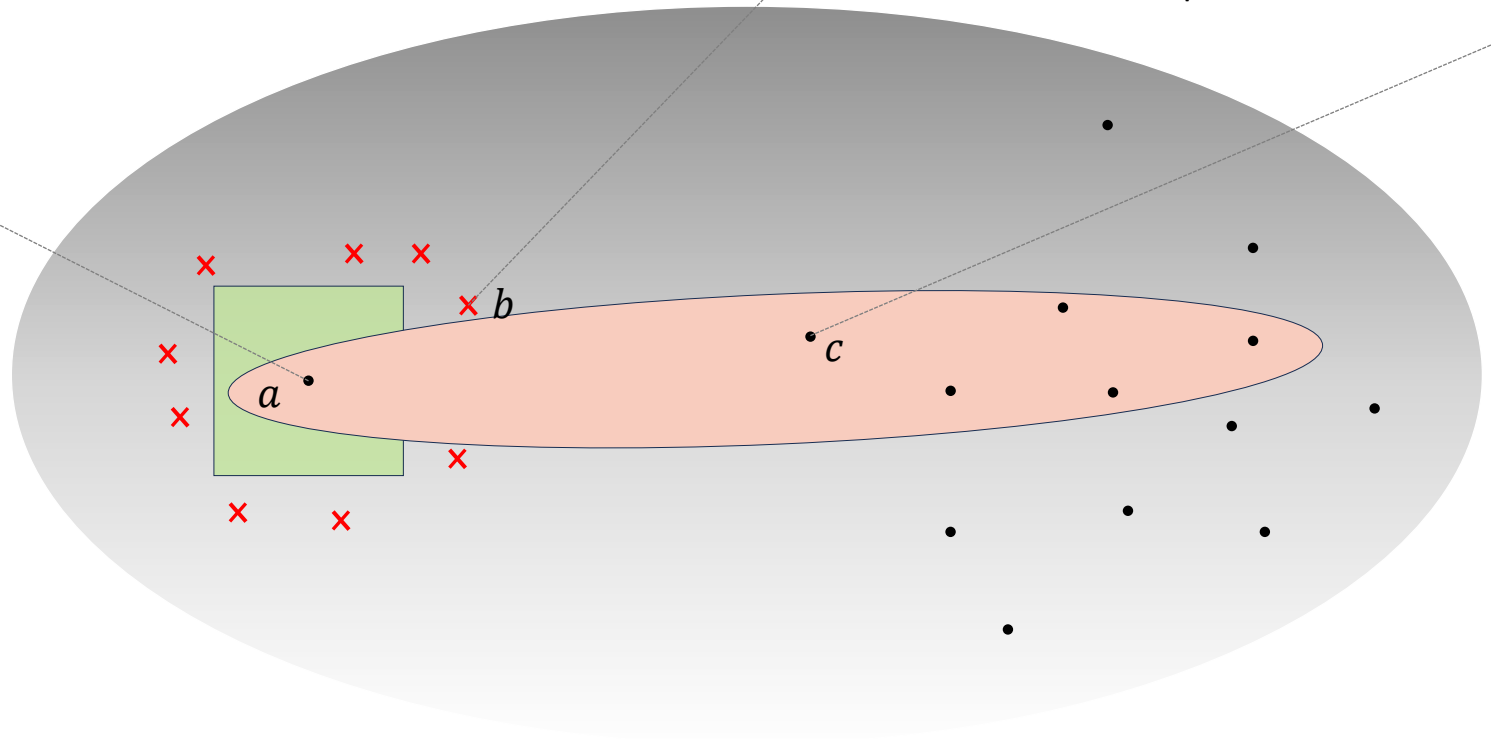
The L_∞-ball specifications



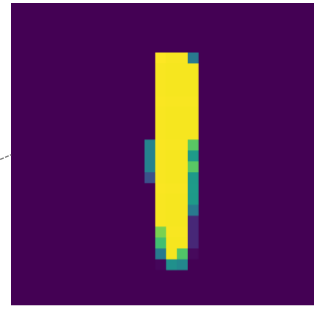
An adv. example



A testing image

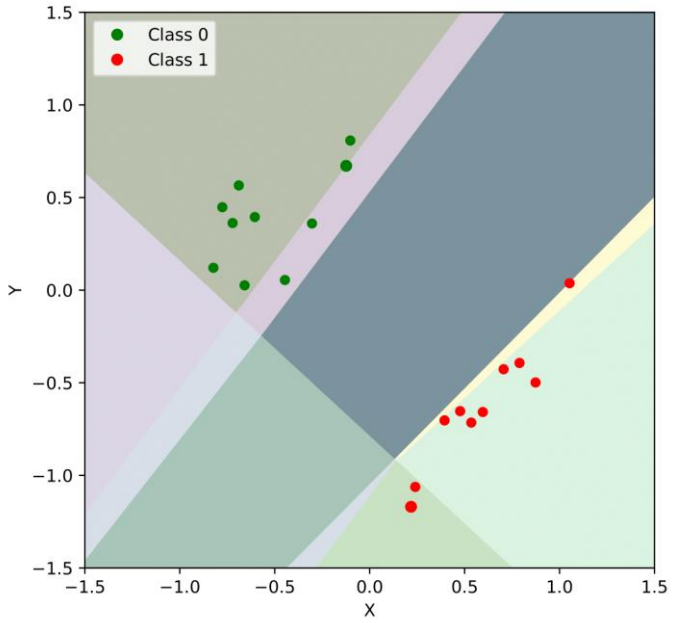


The input space

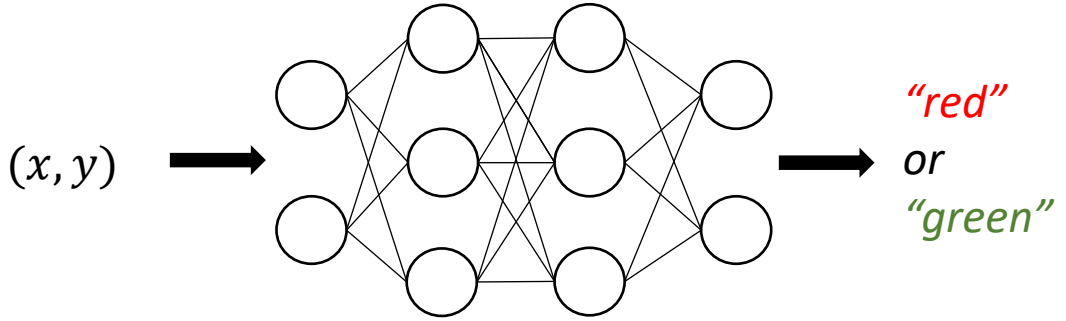


The closest image to a)

The neural activation pattern specifications

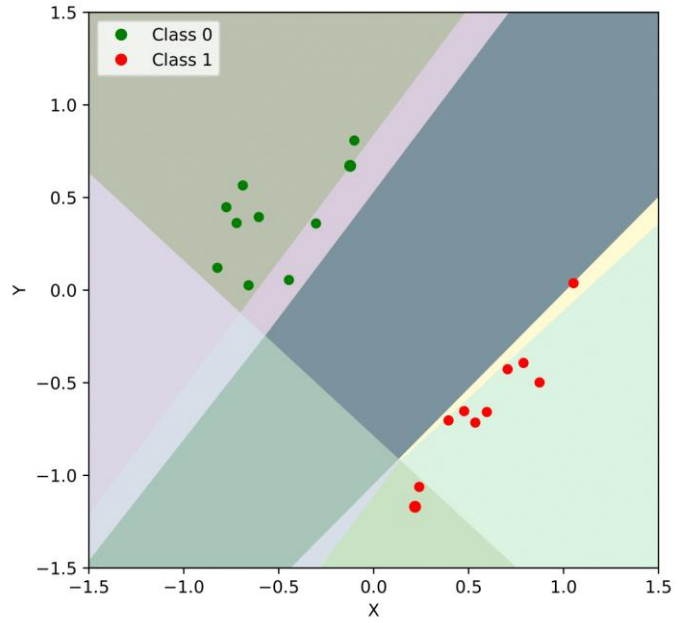


Linear regions in different colors are determined by weights and biases of the neural network. Points colored either red or green constitute the training set.

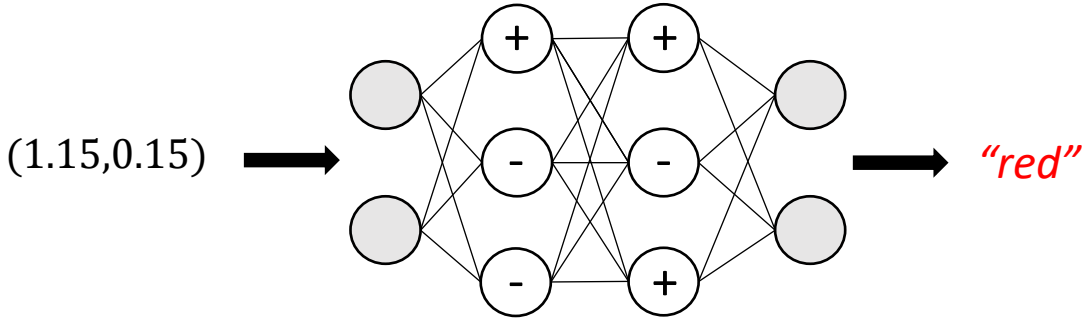


A simple NN with six neurons.

The neural activation pattern specifications

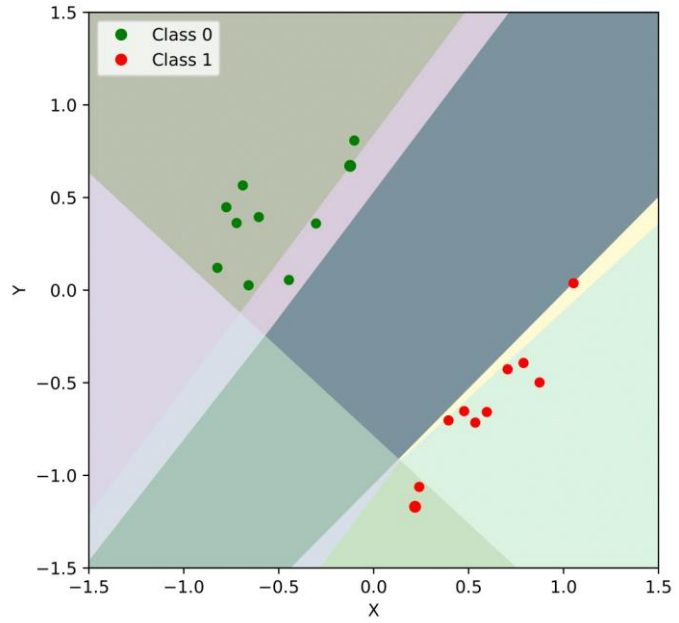


Linear regions in different colors are determined by weights and biases of the neural network. Points colored either red or green constitute the training set.

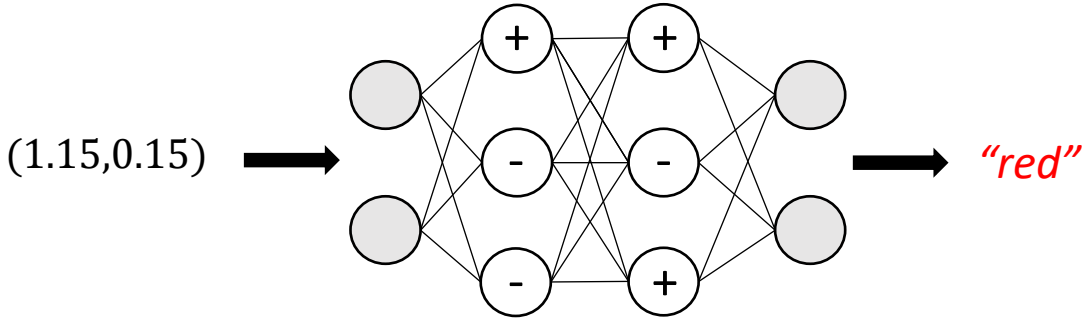


Activated and deactivated neurons are denoted by + and -.

The neural activation pattern specifications



Linear regions in different colors are determined by weights and biases of the neural network. Points colored either red or green constitute the training set.

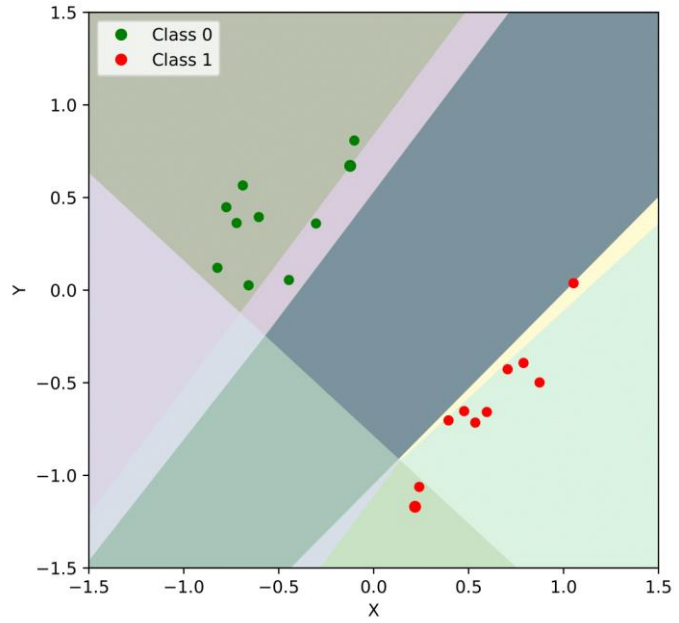


Activated and deactivated neurons are denoted by + and -.

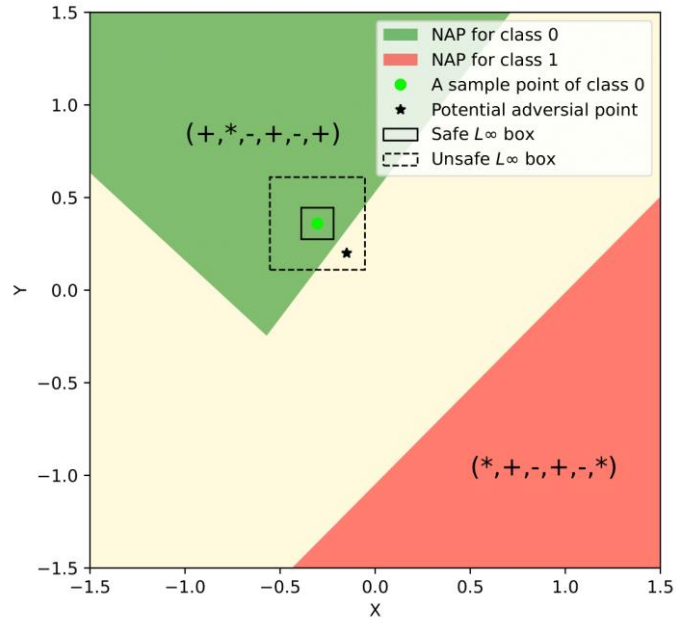
Label	Neuron states	#samples	NAP
0 (Green)	$(+,-,-,+,-,+)$	8	$(+,*,-,+,-,+)$
	$(+,+,-,+,-,+)$	2	
1 (Red)	$(+,+,-,-,+,-)$	7	$(*,+,-,+,-,*)$
	$(-,+,-,-,+,-)$	2	
	$(+,+,-,-,+,+)$	1	

The frequency of each ReLU and the NAPs for each label.
* denotes an arbitrary neuron state.

The neural activation pattern specifications



Linear regions in different colors are determined by weights and biases of the neural network. Points colored either red or green constitute the training set.



NAPs are more flexible than L_∞ norm-balls (boxes) in terms of covering verifiable regions.

Neural activation pattern (NAP)

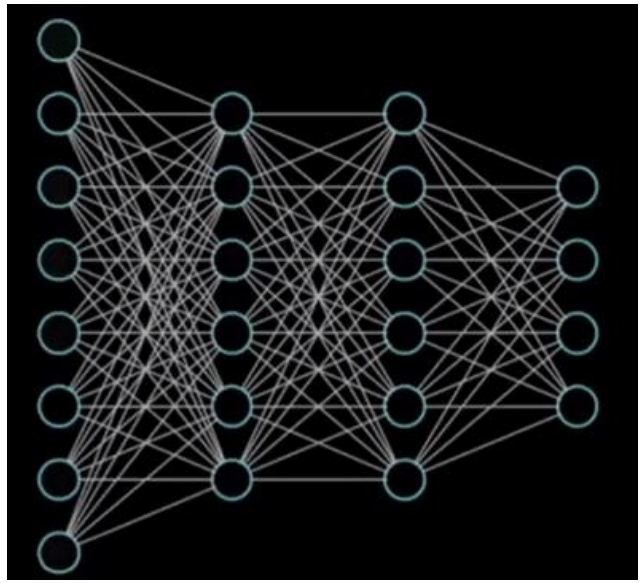
A ReLU activated neuron output = *max(0, input)* is either "activated" or "deactivated"

Running each input through the network emits an activation pattern

Insights:

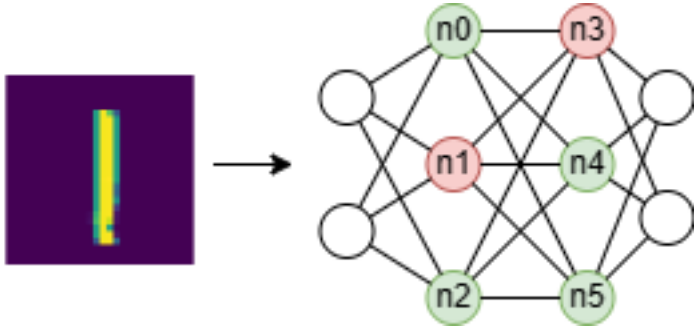
- Do images of the same class emit *similar* patterns?
- Can we use a NAP $\mathcal{P}$ to *certify* all inputs associated with a specific label?

  (In other words, given a pattern $\mathcal{P}$, if x follows $\mathcal{P}$, must x be classified as a certain label?)



Credit to:
https://youtu.be/aircAruvnKk

Formal definitions of NAP



A NAP $\mathcal{P}$ is a tuple (A, D)

- A and D are two disjoint subsets of activated and deactivated neurons.
- $\mathcal{P}$ may or may not contain all neurons in the network

Ordering of NAPs: $\mathcal{P}_1 \leq \mathcal{P}_2$

iff A_1 is a superset of A_2

and D_1 is a superset of D_2

NAP examples:
$\mathcal{P}_0 = ((n_0), (n_1))$
$\mathcal{P}_1 = ((n_0, n_2, n_5), (n_1, n_3))$
$\mathcal{P}_2 = ((), ())$
$\mathcal{P}_E = ((n_0, n_2, n_4, n_5), (n_1, n_3))$
$\mathcal{P}_E \leq \mathcal{P}_1 \leq \mathcal{P}_0 \leq \mathcal{P}_2$

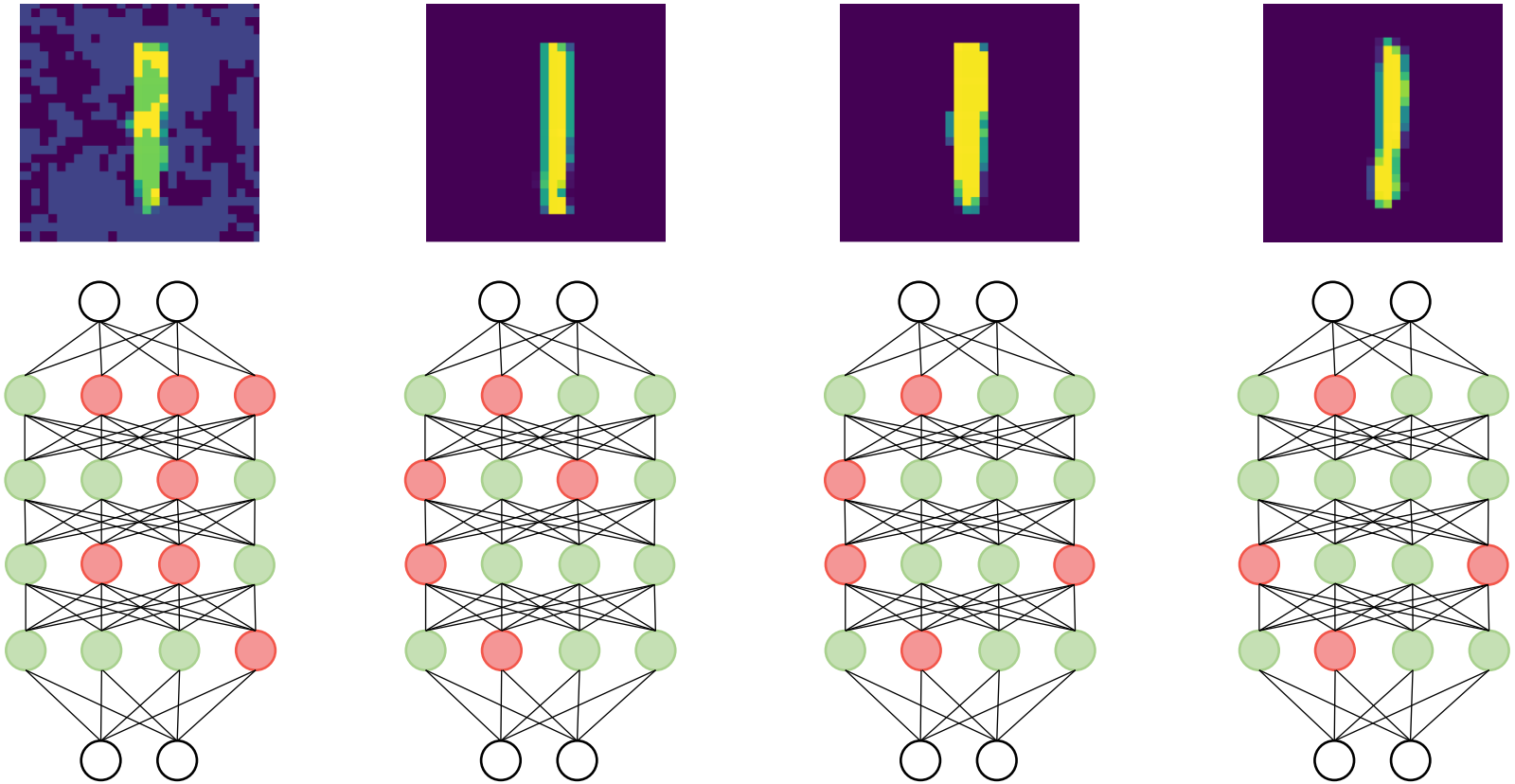
Extraction function:

- $E(\mathcal{N}, x)$ returns $\mathcal{P}_E = (A_E , D_E)$
- $\mathcal{P}_E$ represents all the activated and deactivated neurons when passing x through $\mathcal{N}$

An input x *follows* a NAP $\mathcal{P}$ of a neural network $\mathcal{N}$ if $E(\mathcal{N}, x) \leq \mathcal{P}$

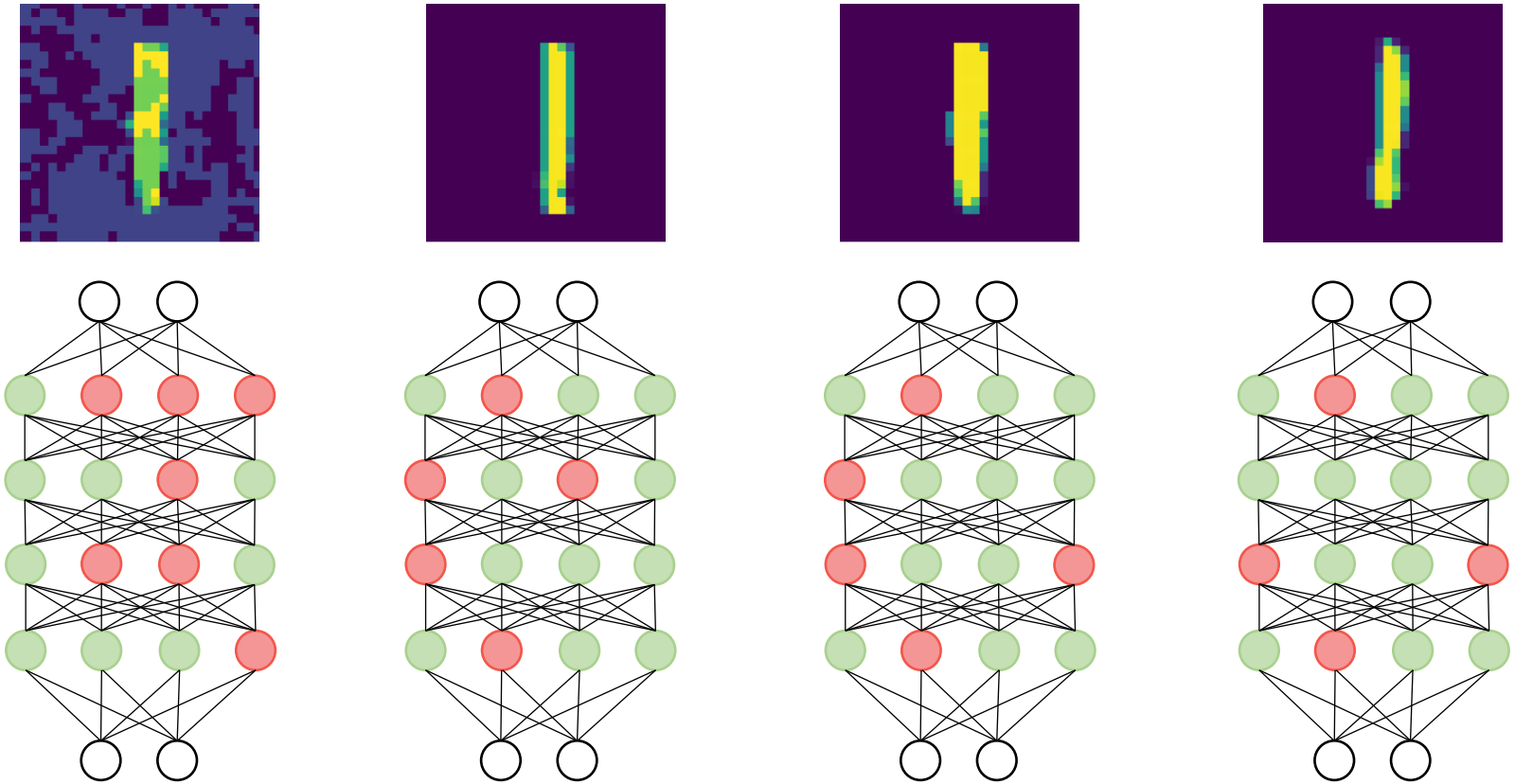
Mining NAPs for label/class

"Do images of the same class emit *similar* patterns?"



Mining NAPs for label/class

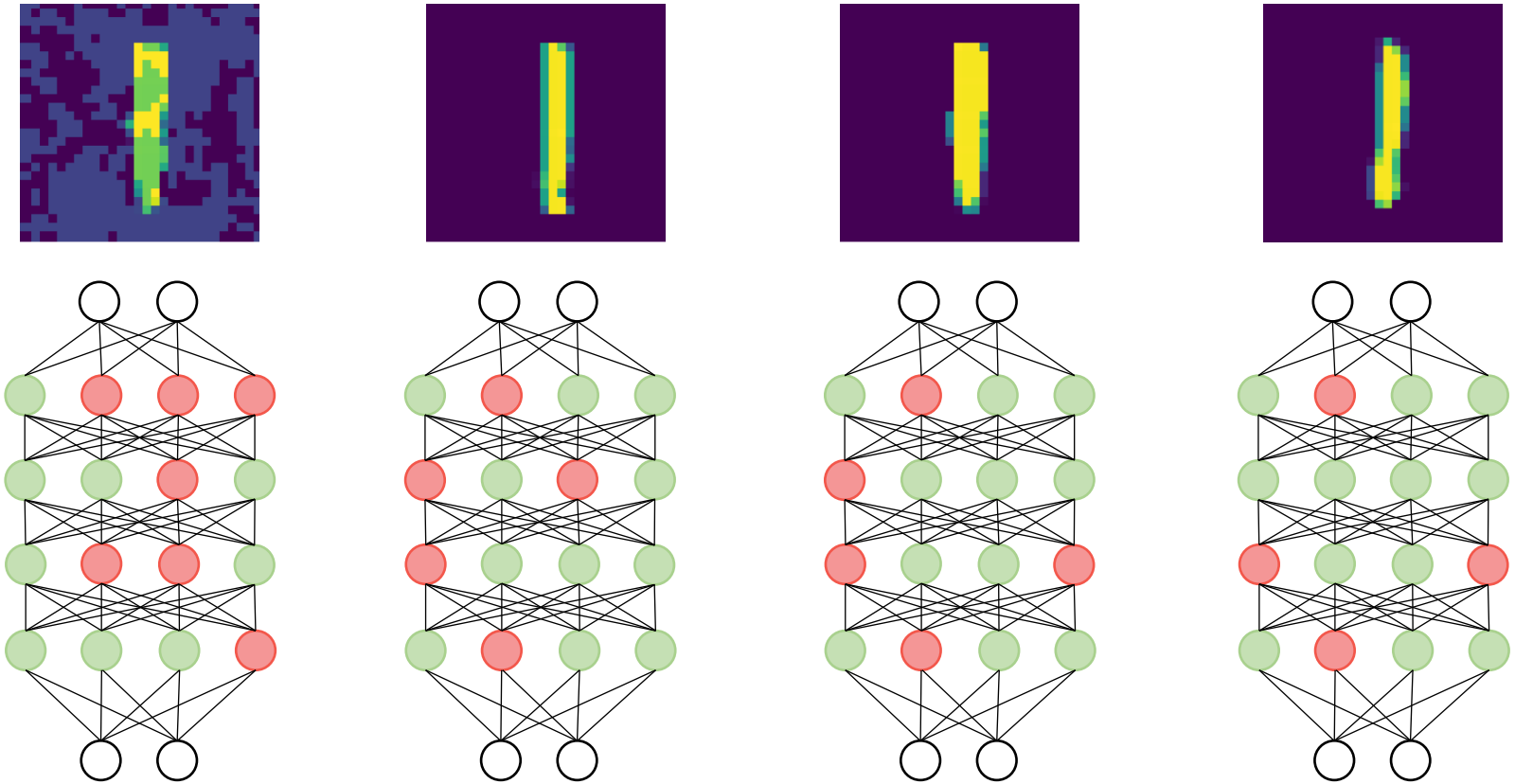
"Do images of the same class emit *similar* patterns?"



Goal: Identify neurons that exhibit consistent behavior across all inputs associated with a specific label.

Mining NAPs for label/class

"Do images of the same class emit *similar* patterns?"



Goal: Identify neurons that exhibit consistent behavior across all inputs associated with a specific label.

NAP for label 1

Mining NAPs for label/class

"Do images of the same class emit *similar* patterns?"

Goal: Find a NAP $\mathcal{P}_l$ such that the majority of inputs with label l follow $\mathcal{P}_l$

The "majority" is controlled by the relaxing factor δ: should $\mathcal{P}_l$ be followed by 100% inputs, or 99.5% inputs with label l

Note:

- lower δ -> more neurons are fixed -> many inputs with label *l* may not follow the NAP (low recall)
- higher δ -> fewer neurons are fixed -> inputs of other labels may also follow the NAP (low precision)

Algorithm 1 NAP Mining Algorithm

Input: relaxing factor δ, neural network $\mathcal{N}$, dataset S_ℓ
Initialize a counter c_k for each neuron v_k
for $x \in S_\ell$ **do**
  compute $E(\mathcal{N}, x)$
  if v_k is activated **then**
    $c_k \mathrel{+}= 1$
$A_\ell \leftarrow \{v_k \mid \frac{c_k}{|S_\ell|} \geq \delta\}$, $D_\ell \leftarrow \{v_k \mid \frac{c_k}{|S_\ell|} \leq 1 - \delta\}$
$\mathcal{P}_\ell^\delta \leftarrow (A_\ell, D_\ell)$

Intriguing NAP properties

"Can we use a NAP $\mathcal{P}$ to certify all inputs associated with a specific label?"

Non-ambiguity: An input can't follow two different NAPs at the same time.

NAP-robustness: Inputs that follows a $\mathcal{P}_l$ for label l are predicted as l by the network.

NAP-augmented robustness: Inputs within some ϵ-balls that follow a $\mathcal{P}_l$ are predicted as l by the network.

Intriguing NAP properties

"Can we use a NAP $\mathcal{P}$ to certify all inputs associated with a specific label?"

Non-ambiguity: An input can't follow two different NAPs at the same time.

$$\forall x \cdot \forall i \neq j \cdot E(N, x) \leq P_i \implies E(N, x) \not\leq P_j$$

NAP-robustness: Inputs that follows a $\mathcal{P}_l$ for label l are predicted as l by the network.

$$\forall x \in \{x \mid E(N, x) \leq P_i\} \cdot \forall j \neq i \cdot F(x)[i] - F(x)[j] > 0$$

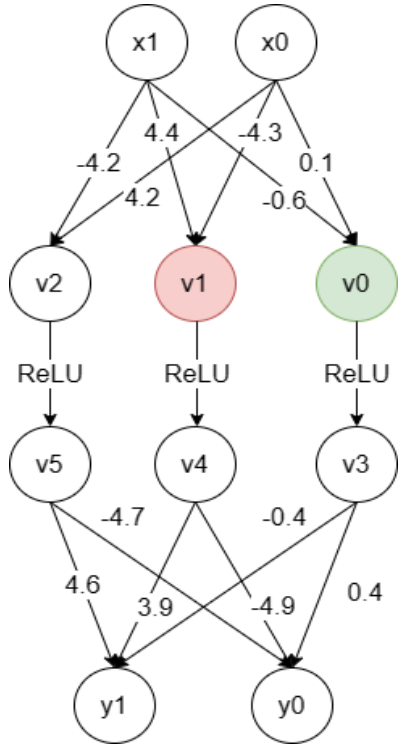
NAP-augmented robustness: Inputs within some ϵ-balls that follow a $\mathcal{P}_l$ are predicted as l by the network.

$$\forall x' \in B^+(x, \epsilon, P_i) \cdot \forall j \neq i \cdot F(x')[i] - F(x')[j] > 0$$

$$B^+(x, \epsilon, P_i) = \{x' \mid ||x - x'||_\infty \leq \epsilon, E(N, x') \leq P_i\}$$

Working with NAPs

- NAP properties can be encoded as extra QF-LRA constraints
- Verify the property using any off-the-shelf neural network verification tool
- Example: check the right NN is NAP-robust with $\mathcal{P}_0$ = ((v0), (v1))

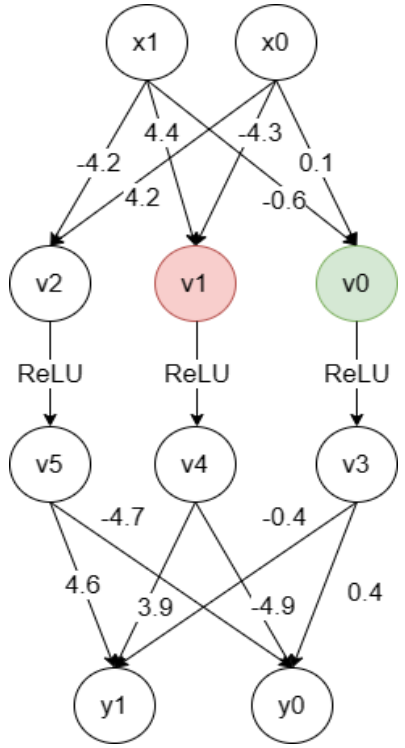


Working with NAPs

- NAP properties can be encoded as extra QF-LRA constraints
- Verify the property using any off-the-shelf neural network verification tool
- Example: check the right NN is NAP-robust with $\mathcal{P}_0$ = ((v0), (v1))

$$
\begin{aligned}
v_0 &= 0.1x_0 - 0.6x_1 \\
v_1 &= -4.3x_0 + 4.4x_1 \\
v_2 &= 4.2x_0 - 4.2x_1 \\
v_3 &= max(v_0, 0) \\
v_4 &= max(v_1, 0) \\
v_5 &= max(v_2, 0) \\
y_0 &= 0.4v_3 - 4.9v_4 + 3.9v_5 + 6.7 \\
y_1 &= -0.4v_3 + 3.9v_4 + 4.6v_5 - 7.4
\end{aligned}
$$

the network encoded as a set of constraints



Working with NAPs

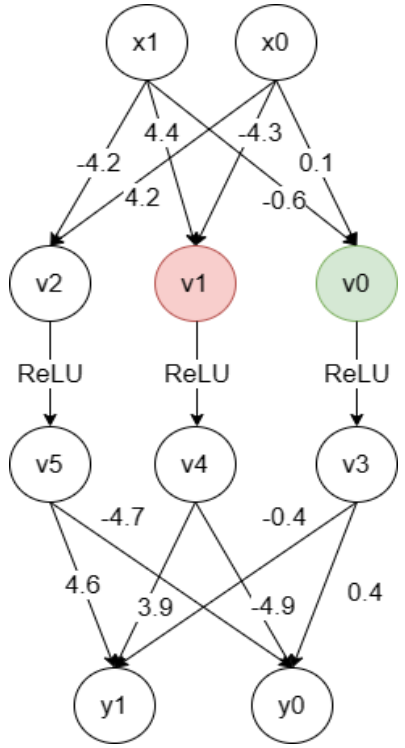
- NAP properties can be encoded as extra QF-LRA constraints
- Verify the property using any off-the-shelf neural network verification tool
- Example: check the right NN is NAP-robust with $\mathcal{P}_0$ = ((v0), (v1))

$$v_0 = 0.1x_0 - 0.6x_1$$
$$v_1 = -4.3x_0 + 4.4x_1$$
$$v_2 = 4.2x_0 - 4.2x_1$$
$$v_3 = max(v_0, 0)$$
$$v_4 = max(v_1, 0)$$
$$v_5 = max(v_2, 0)$$
$$y_0 = 0.4v_3 - 4.9v_4 + 3.9v_5 + 6.7$$
$$y_1 = -0.4v_3 + 3.9v_4 + 4.6v_5 - 7.4$$

$v_o > o$ *#NAP constraint*
$v_1 \leq o$ *#NAP constraint*
$y_o < y_1$ *#Negation of the property*

NN-Verifier

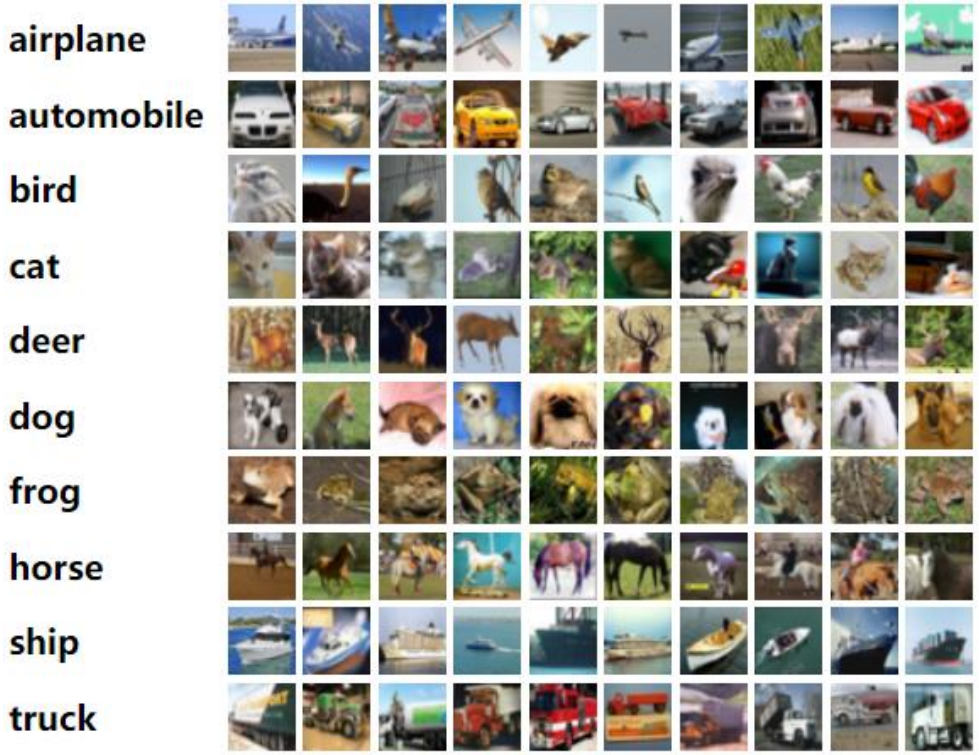
UNSAT
(SAFE)



Results

Datasets

CIFAR10



MNIST

MNIST with FCN

	0	1	2	3	4	5	6	7	8	9
$\epsilon = 0.2$, no NAP	N	-	N	T/o	T/o	T/o	T/o	T/o	N	T/o
$\epsilon = 0.2, \delta = 1.0$	N	-	N	Y	T/o	T/o	Y	T/o	N	N
$\epsilon = 0.2, \delta = 0.99$	Y	-	Y	Y	Y	Y	Y	Y	Y	Y
$\epsilon = 1, \delta = 0.99$ (NAP robustness property)	Y	-	Y	Y	Y	Y	Y	Y	Y	Y

- With L_∞ balls, the neighborhood of ϵ=0.2 of the reference point cannot be verified.
- Augmenting with the NAP $\mathcal{P}_1^{0.99}$succeed in verifying the neighborhood of ϵ=0.2 of the reference point.
- $\mathcal{P}_1^{0.99}$ can verify the whole input space, covering 84% of test images.

CIFAR10 with CNN

ϵ	0.012			0.024			0.12		
δ	0.99	0.95	0.9	0.99	0.95	0.9	0.99	0.95	0.9
$\mathcal{O}(x_0) = 8$	Y	Y	Y	N	T/o	Y	T/o	Y	Y
$\mathcal{O}(x_1) = 6$	T/o	N	Y	N	N	Y	N	N	Y
$\mathcal{O}(x_2) = 0$	Y	Y	Y	Y	Y	Y	N	N	N
$\mathcal{O}(x_3) = 1$	N	N	N	N	N	N	N	N	N
$\mathcal{O}(x_4) = 9$	N	Y	Y	N	N	N	N	N	N
$\mathcal{O}(x_5) = 7$	Y	Y	Y	N	T/o	Y	N	Y	Y
$\mathcal{O}(x_6) = 3$	Y	Y	Y	Y	Y	Y	N	N	N

- With NAPs, we are able to verify more examples at every ϵ.
- We push the known verifiable bound to 10 times larger on the CIFAR10 benchmark.

Thank You!

Please contact: chuqin.geng@mail.mcgill.ca if you have any questions!