Deep Networks as Logical Circuits: Generalization and Interpretation

The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

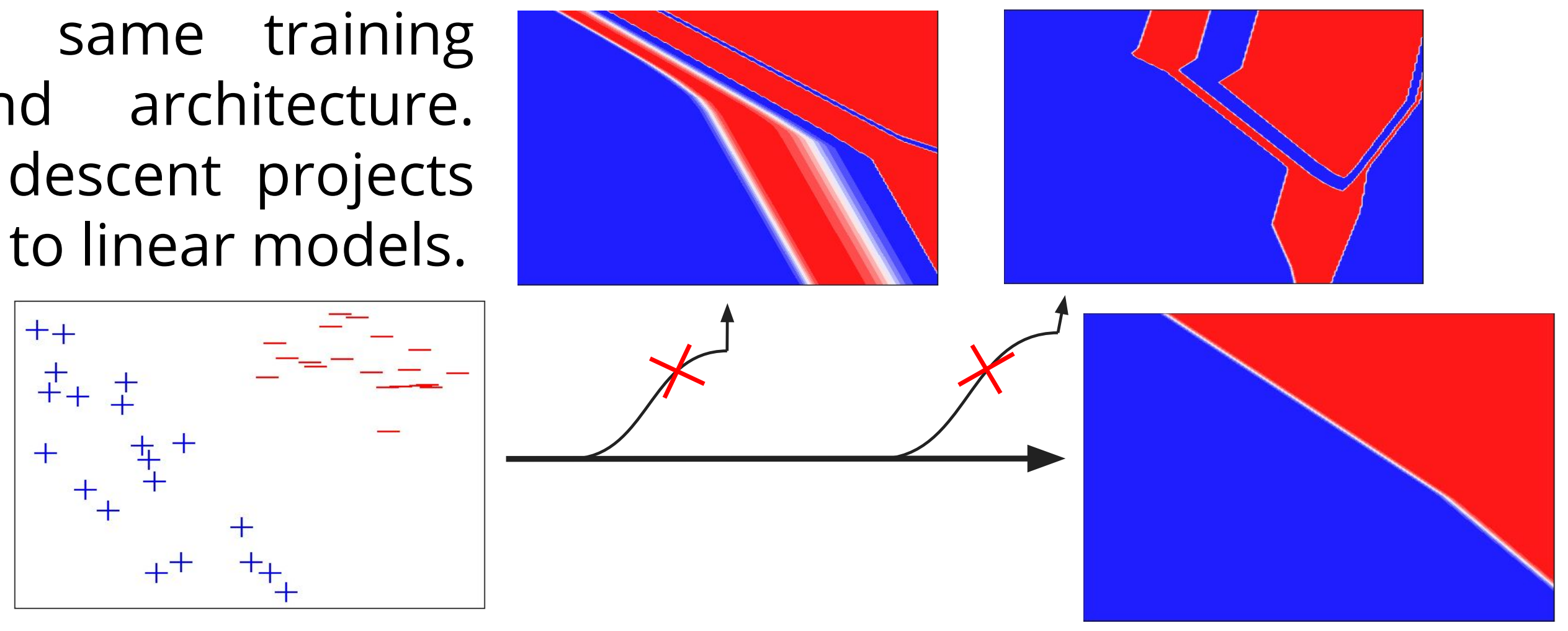
Christopher Snyder[1]. Sriram Vishwanath[2].

[1] *Department of Biomedical Engineering, University of Texas at Austin, christopher.g.snyder@utexas.edu*

[2] *Department of Electrical and Computer Engineering, University of Texas at Austin, Professor, sriram@austin.utexas.edu*

CASE STUDIES

Figure 1. Three models with the same training error and architecture. Gradient descent projects initialized to linear models.



We can generalize past linear models by finding model features we can use to deduce the regularity of the original data

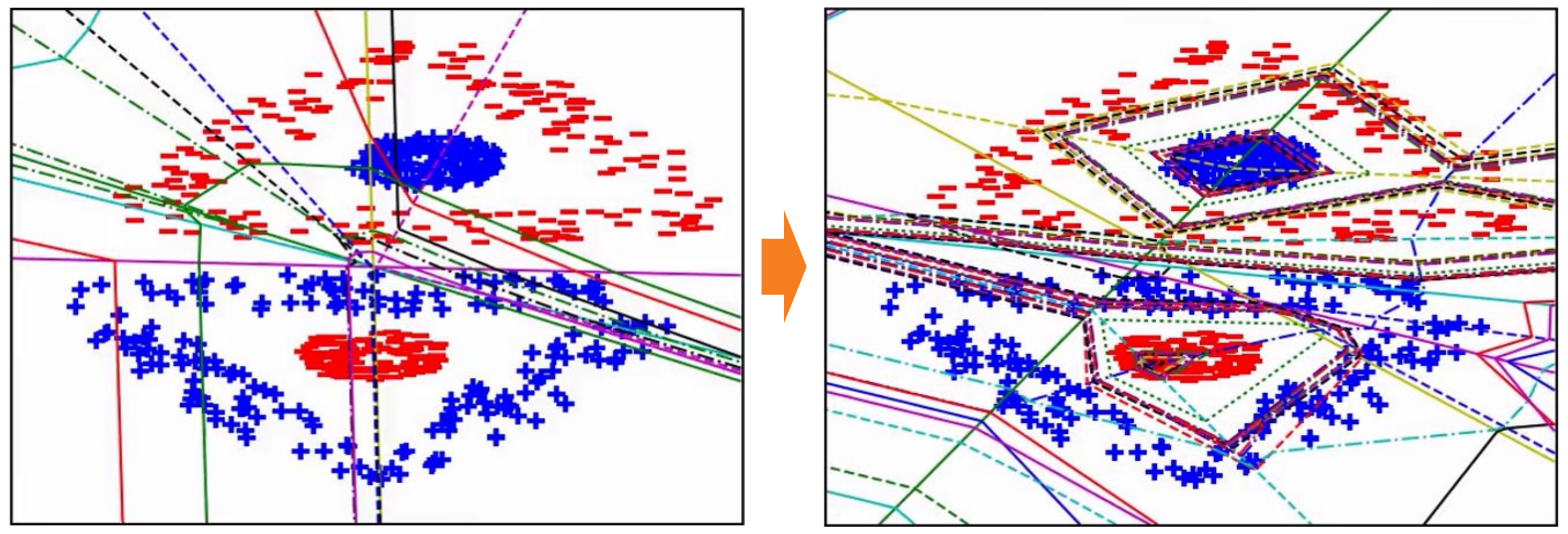
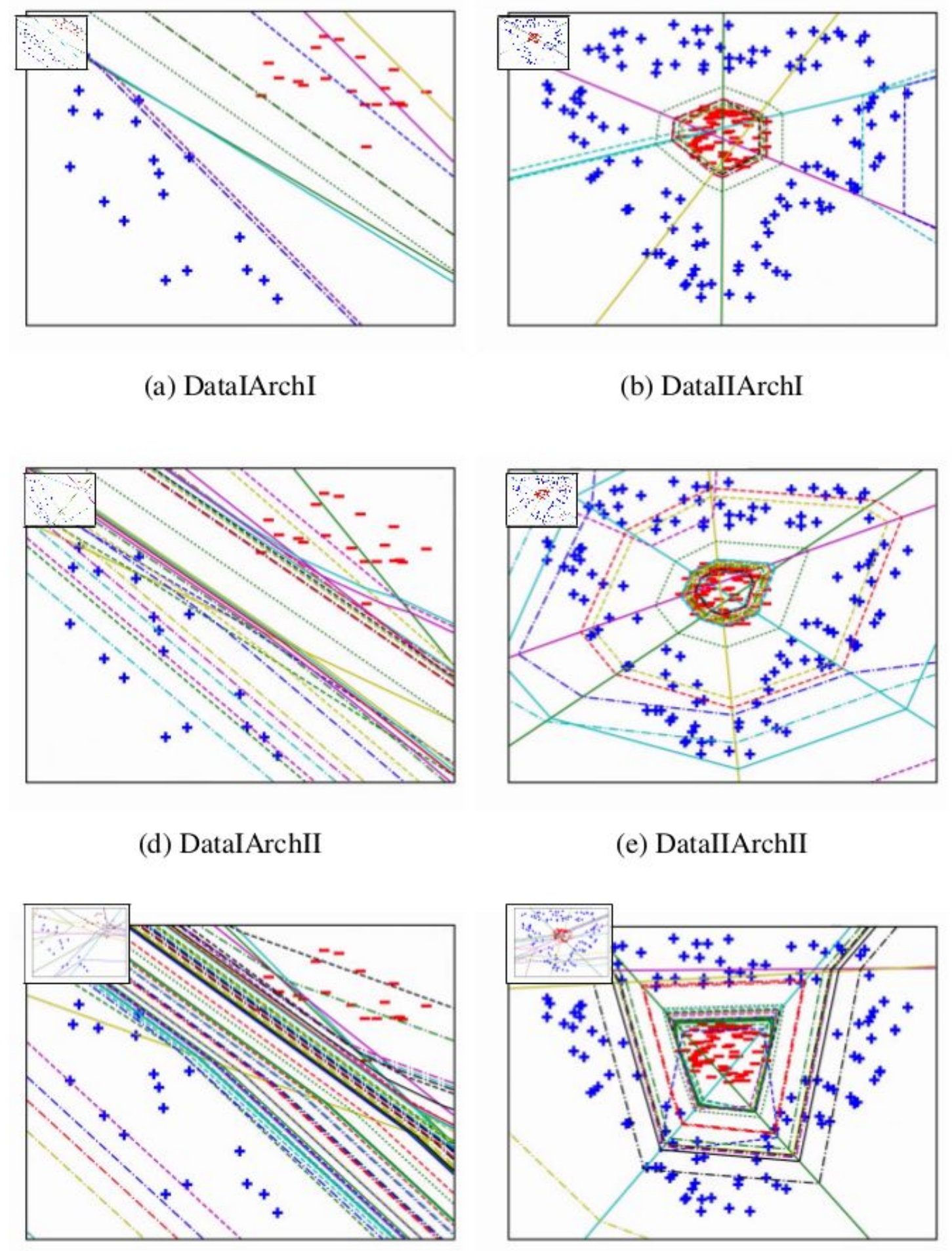


Figure 2. Initialized and Learned State Boundaries(SBs), where neurons (counting the output) switch from "on/off". Each SB may bend when intersecting another from a previous layer, establishing dependence. A linear classifier is one where the decision boundary intersects no NSB.

Figure 3. We see that for fixed dataset, increasing the architecture size (moving down a column) does not qualitatively change the learned SBs. Additional layers may add more SBs, but these organize during training in redundant, parallel shells that do not contribute to the decision bdry.



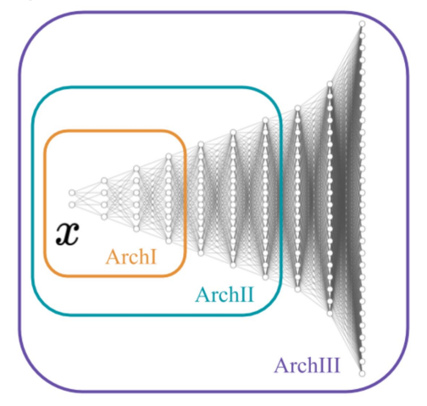
(a) DataIArchI

(b) DataIIArchI

(d) DataIArchII

(e) DataIIArchII

This holds even for giant architectures **~1e6 param**



<u>Insight</u>: represent the network with combinations of linear classifiers

INTRODUCTION

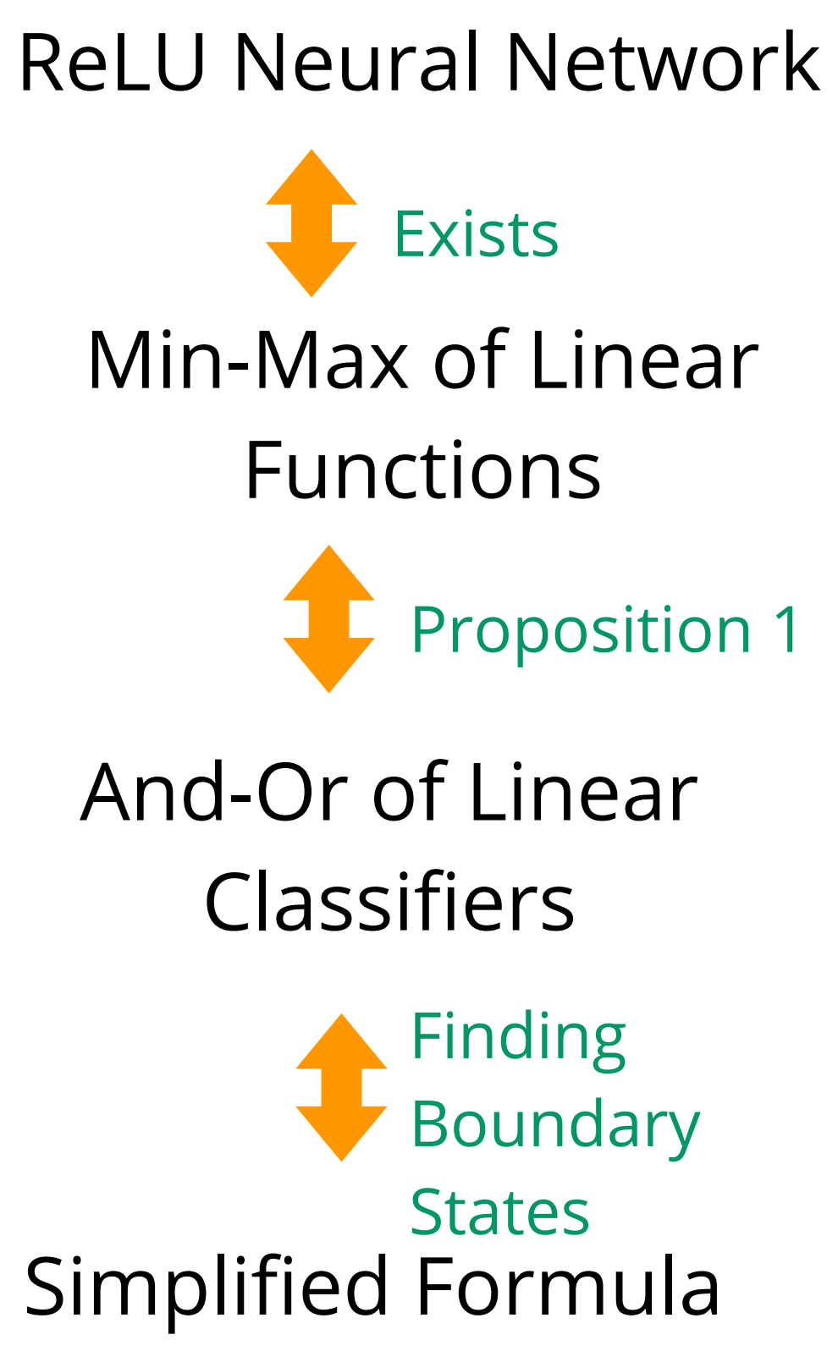
- ❑ Deep Neural Networks(DNNs) are widely used but poorly understood
- ❑ How and why Deep Learning works is a major open question.
- ❑ To rule out failure cases, we *must* make assumptions about the data

We find in our experiments that DNNs of any architecture trained on linearly classifiable data are almost always linear classifiers

- ❑ Practical DNNs are not well understood even as function space

- ❑ We present a DNN representation in terms of a hierarchical AND/OR combination of linear classifiers.
- ❑ The structure of this "Logical Circuit":
  - ❑ Reflects the network function, not the architecture
  - ❑ Can require many orders fewer parameters
  - ❑ Encodes the training data spatial structure, not the depth.

Reparameterization Pathway



ReLU Neural Network

Exists

Min-Max of Linear Functions

Proposition 1

And-Or of Linear Classifiers

Finding Boundary States

Simplified Formula

Key Point:

- ❑ All networks may be represented this way, but overfit networks will not simplify

f,g a pair of linear functions per linear region

$$\mathcal{N}^l(x) = f^l_{\sigma(x)}(x) - g^l_{\sigma(x)}(x) \text{ for } l = 1, \dots, d+1$$

$$\mathcal{N}(x) = \left(\max_{\mu\in\{0,1\}^n} f_\mu(x)\right) - \left(\max_{\tau\in\{0,1\}^n} g_\tau(x)\right)$$

For example: [Zhang et al, 2017. Tropical Geometry of Deep Neural Networks]

$$\mathcal{N}(x) = \max_{\mu^d}\min_{\tau^d}\cdots\max_{\mu^1}\min_{\tau^1}\Big(f_\mu(x) - g_\tau(x)\Big)$$

Proposition 1. *Let $f : \mathcal{A}\times\mathcal{B} \mapsto \mathbb{R}$. Then we have the following logical equivalence:*

$$\left[\max_{\alpha\in\mathcal{A}}\min_{\beta\in\mathcal{B}} f(\alpha,\beta) \geq 0\right] \iff \bigvee_{\alpha\in\mathcal{A}}\bigwedge_{\beta\in\mathcal{B}}\Big[f(\alpha,\beta)\geq 0\Big]$$

$$\Rightarrow \Big[\mathcal{N}(x)\geq 0\Big] \Leftrightarrow \bigvee_{\mu}\bigwedge_{\tau}\Big[f_\mu(x) - g_\tau(x) \geq 0\Big]$$

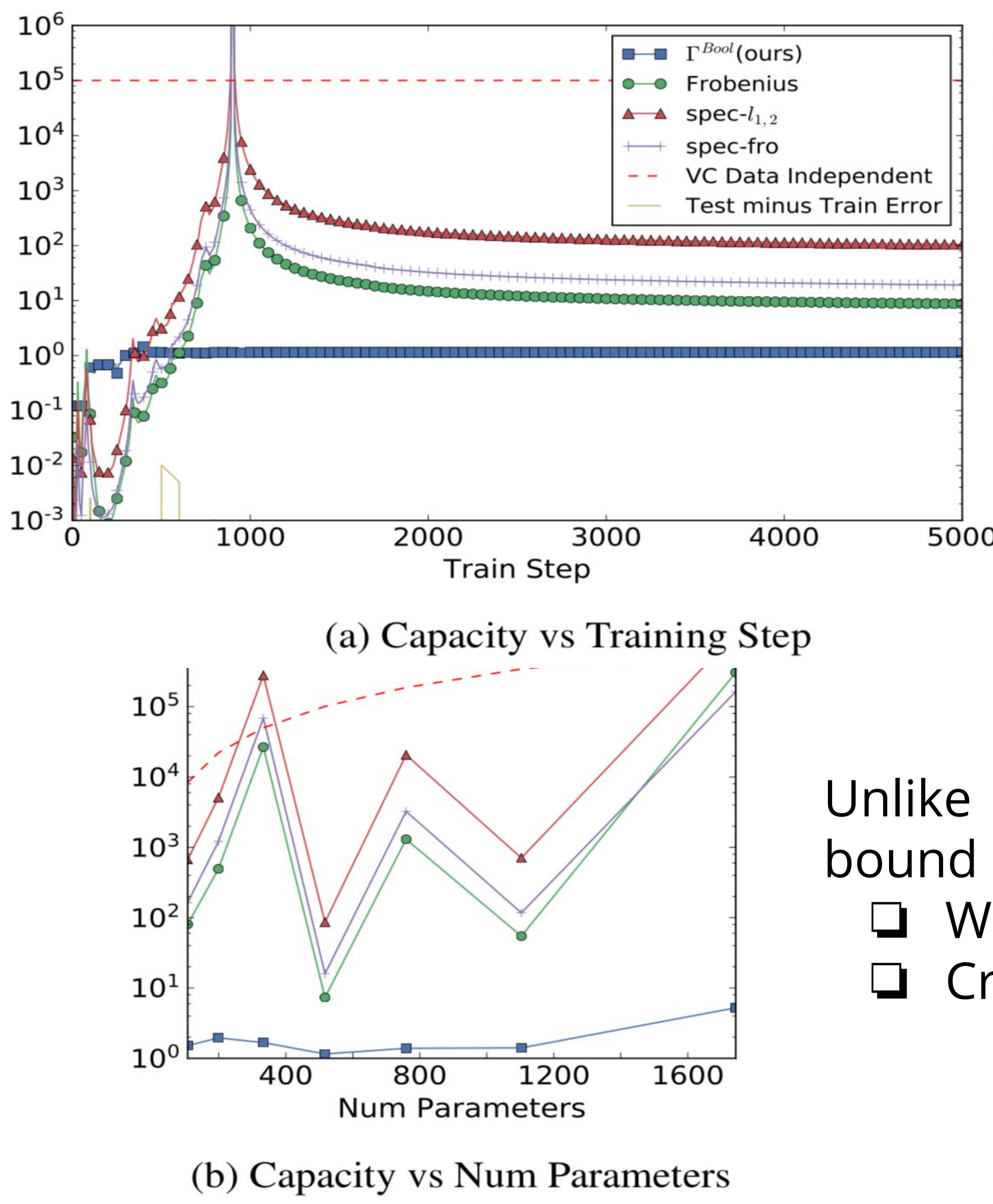
$$\Rightarrow \Big[\mathcal{N}(x)\geq 0\Big] \Leftrightarrow \bigvee_{\mu^d}\bigwedge_{\tau^d}\cdots\bigvee_{\mu^1}\bigwedge_{\tau^1}\Big[f_\mu(x) - g_\tau(x) \geq 0\Big]$$

Theorem 4. *Let $\mathcal{N} : \mathbb{R}^{n_0} \mapsto \mathbb{R}$ be a fully-connected ReLU network. Suppose the Boolean formula, $\Phi(\mathcal{N})$, is of class (k,s,d). Define the hypothesis class $\mathcal{H}_{\Phi(\mathcal{N})} \triangleq \{x \mapsto \Phi(\mathcal{N})(w,x) | w \in \mathbb{R}^k\}$. Then*

1. $x \mapsto [\mathcal{N}(x)\geq 0] \in \mathcal{H}_{\Phi(\mathcal{N})}$
2. $VCDim(\mathcal{H}_{\Phi(\mathcal{N})}) \leq 2k\log_2(8esd)$

IMPACT

Generalization



(a) Capacity vs Training Step

(b) Capacity vs Num Parameters

Frobenius: $\frac{1}{m}\frac{1}{\gamma^2}\prod_{l=1}^d \|A^{(l)}\|_F^2$(Neyshabur et al., 2015)

spec-$l_{1,2}$: $\frac{1}{m}\frac{1}{\gamma^2}\prod_{l=1}^d \|A^{(l)}\|_2^2 \sum_{l=1}^d \frac{\|A^{(l)}\|_{1,2}^2}{\|A^{(l)}\|_2^2}$(Bartlett et al., 2017b)

spec-fro: $\frac{1}{m}\frac{1}{\gamma^2}\prod_{l=1}^d \|A^{(l)}\|_2^2 \sum_{l=1}^d h_l \frac{\|A^{(l)}\|_F^2}{\|A^{(l)}\|_2^2}$(Neyshabur et al., 2017)

Γ^{Bool}(ours):$\min_{k,s,d}\sqrt{\frac{VC^{Bool}_{k,s,d}(\mathcal{N})}{m}}$.

Figure 4. Comparison with standard layer-wise bounds that depend each weight norm.

Our Model performs very competitively both throughout training (**a**), and as we increase the model size (**b**)

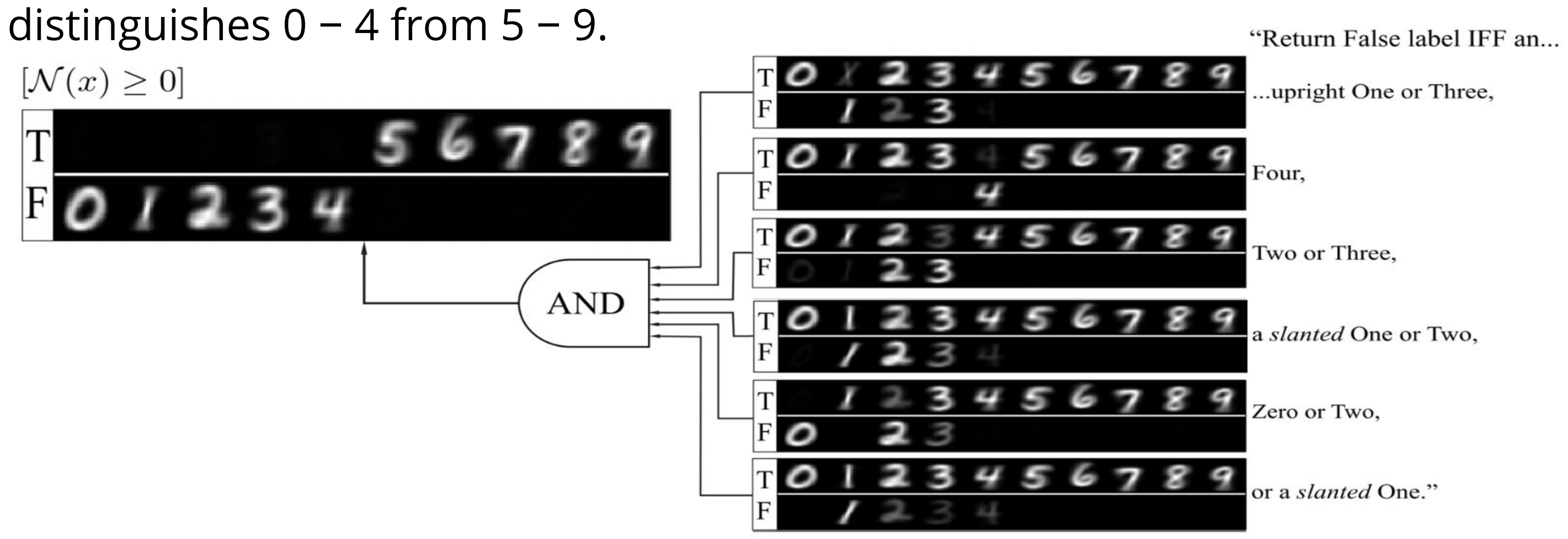
Unlike other learning methods, our bound benefits from information on

- ❑ Weight sign
- ❑ Cross-layer coordination

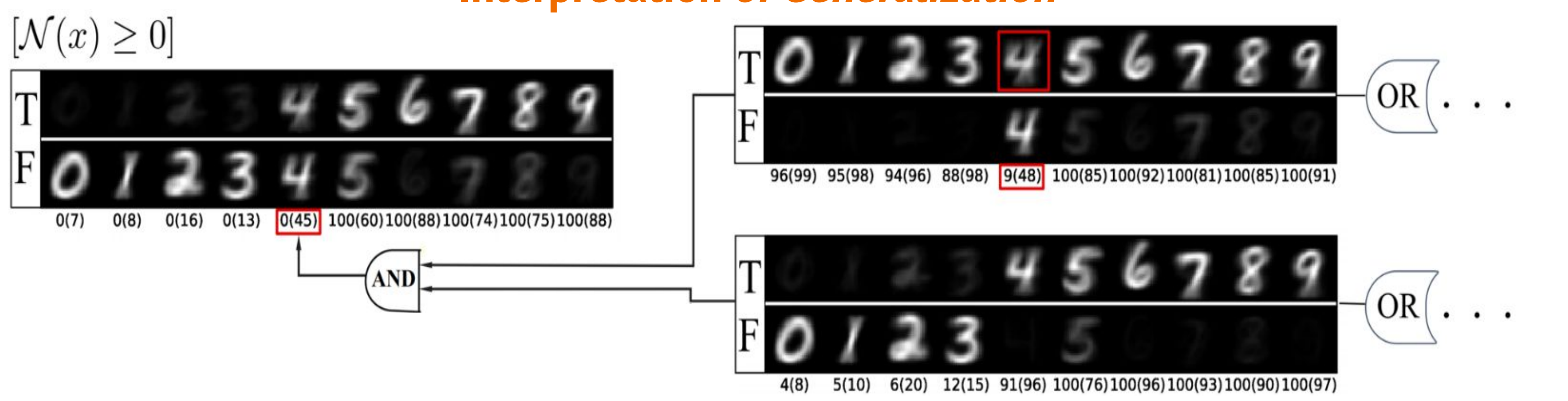
Interpretation

Figure 5. DNNs with Data can be clustered according to classificat

Portions of logical circuits from DNNs trained on MNIST. Each 2 × 10 array represents a different binary classifier within the network circuit, which assigns True or False to every input image. The training objective only distinguishes 0 − 4 from 5 − 9.



Interpretation *of Generalization*



$[\mathcal{N}(x)\geq 0]$
T
F
0(7)
0(8)
0(16)
0(13)
0(45)
100(60)
100(88)
100(74)
100(75)
100(88)
AND
96(99)
95(98)
94(96)
88(98)
9(48)
100(85)
100(92)
100(81)
100(85)
100(91)
OR
4(8)
5(10)
6(20)
12(15)
91(96)
100(76)
100(96)
100(93)
100(90)
100(97)