Adversarial Training and Provable Defenses: Bridging the Gap

Mislav Balunović, Martin Vechev
ETH Zurich

SRILAB

http://www.sri.inf.ethz.ch

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Robustness of Neural Networks

Original image

L_∞ perturbation

Brightness

Geometric

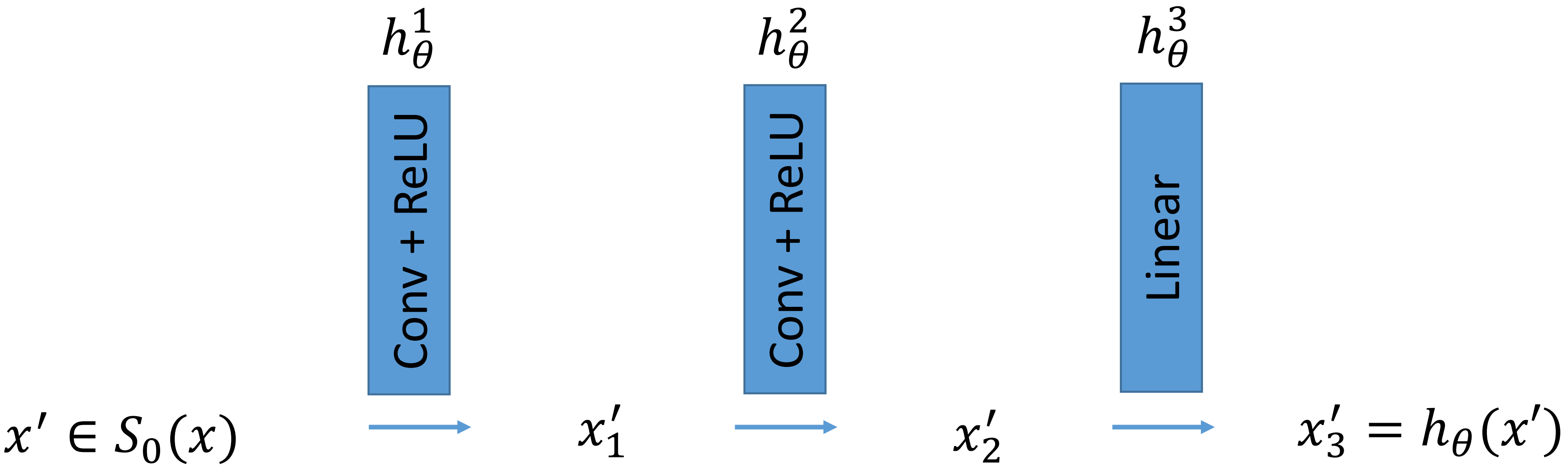
Classified as “panda”

Not classified as “panda”

Background

Given input x, we define convex region $S_0(x)$ as a set of all inputs that attacker can obtain under the specified threat model

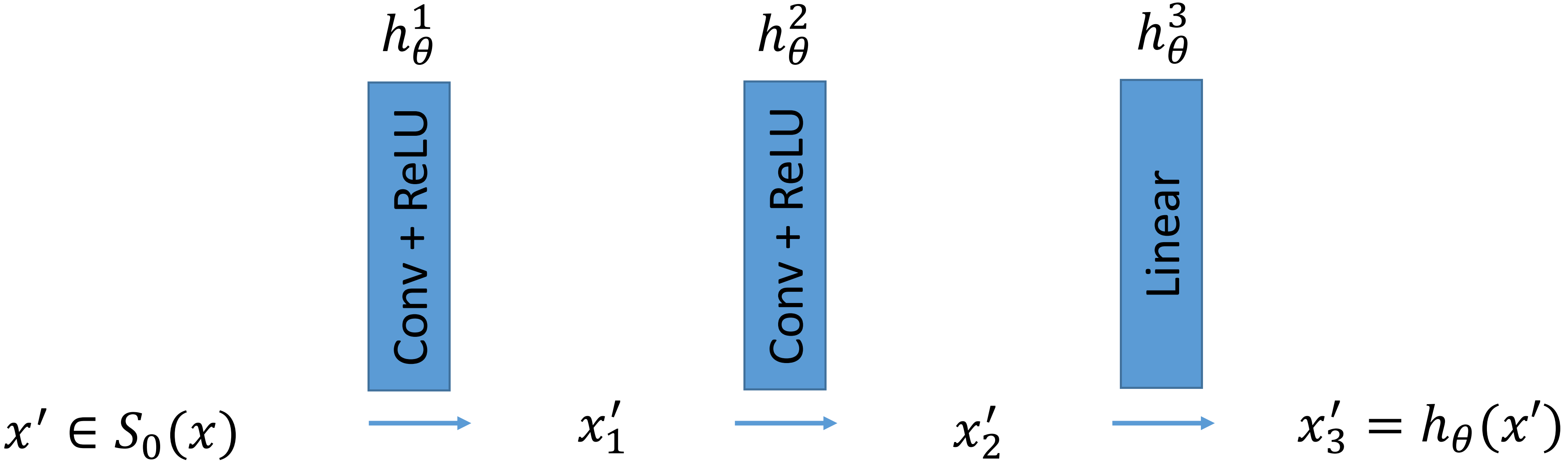
We represent neural network as a function $h_\theta = h_\theta^k \circ h_\theta^{k-1} \circ \cdots \circ h_\theta^1$



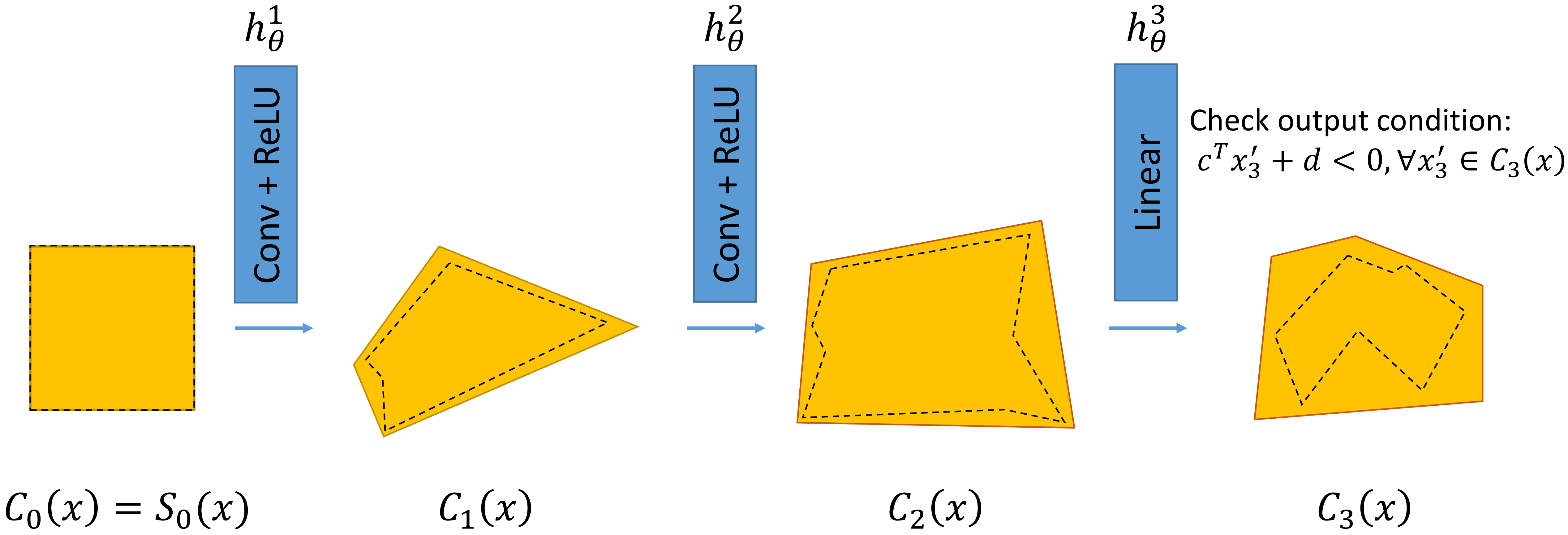
Background

The goal is to prove a property on the output of the network:

$$c^T h_\theta(x') + d < 0, \forall x' \in S_0(x)$$



Certification via convex relaxations



Guarantees: $c^T h_\theta(x') + d < 0, \forall x' \in S_0(x)$

Min-max optimization problem

To train a model which satisfies the constraint, we can define surrogate loss $\mathcal{L}$ and solve the following min-max formulation (Madry et al. 2017):

$$\min_{\theta} E_{(x,y)\sim D} \max_{x' \in S_0(x)} \mathcal{L}(h_\theta(x'), y)$$

This optimization problem can not be solved exactly, so the inner max is usually replaced with an approximation based on **lower** or **upper** bound.

Existing work

Adversarial training

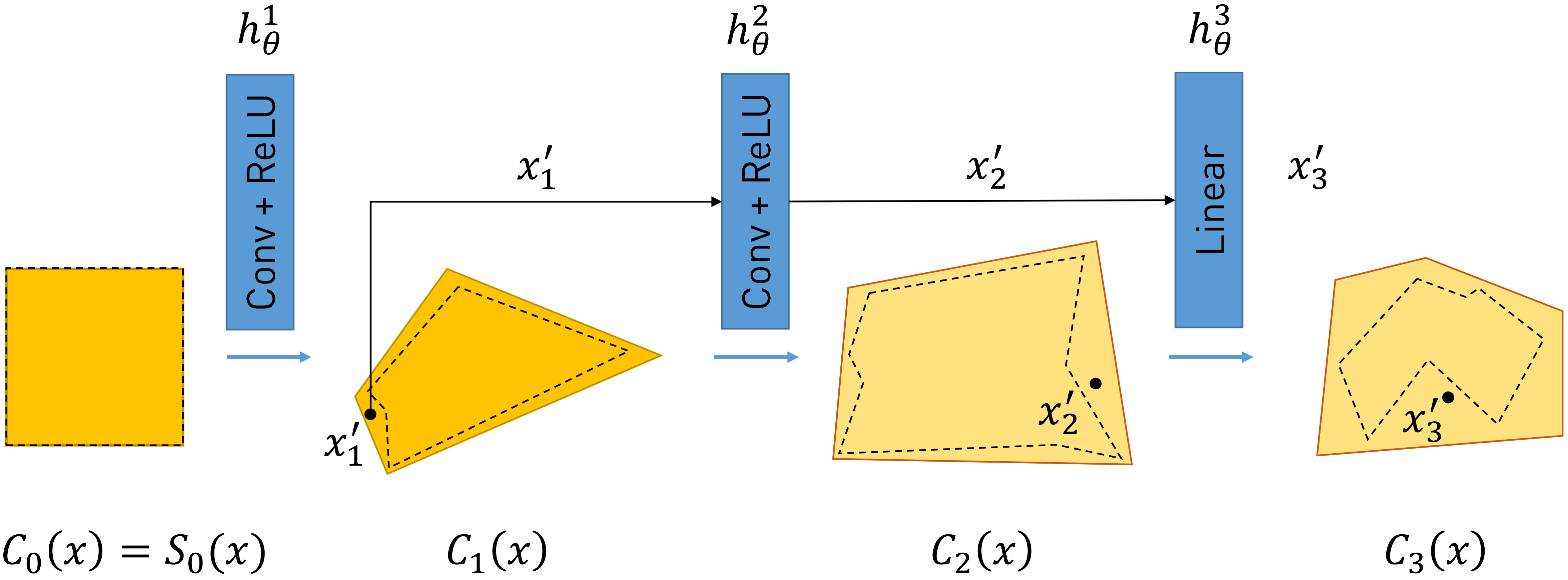
- Replaces inner loss with a **lower** bound
- Szegedy et al. (2014), Goodfellow et al. (2014), Madry et al. (2017)
- Lacks guarantees on robustness of the resulting model

Provable defenses

- Replaces inner loss with an **upper** bound
- Wong et al (2017)., Ragunathan et al. (2018), Mirman et al. (2018)
- Provides guaranetees on robustness, but models have lower accuracy

Our work: Can we combine benefits of both approaches to obtain provably robust networks with high accuracy?

Latent Adversarial Examples



$c^T x_3' + d < 0$ → certification fails

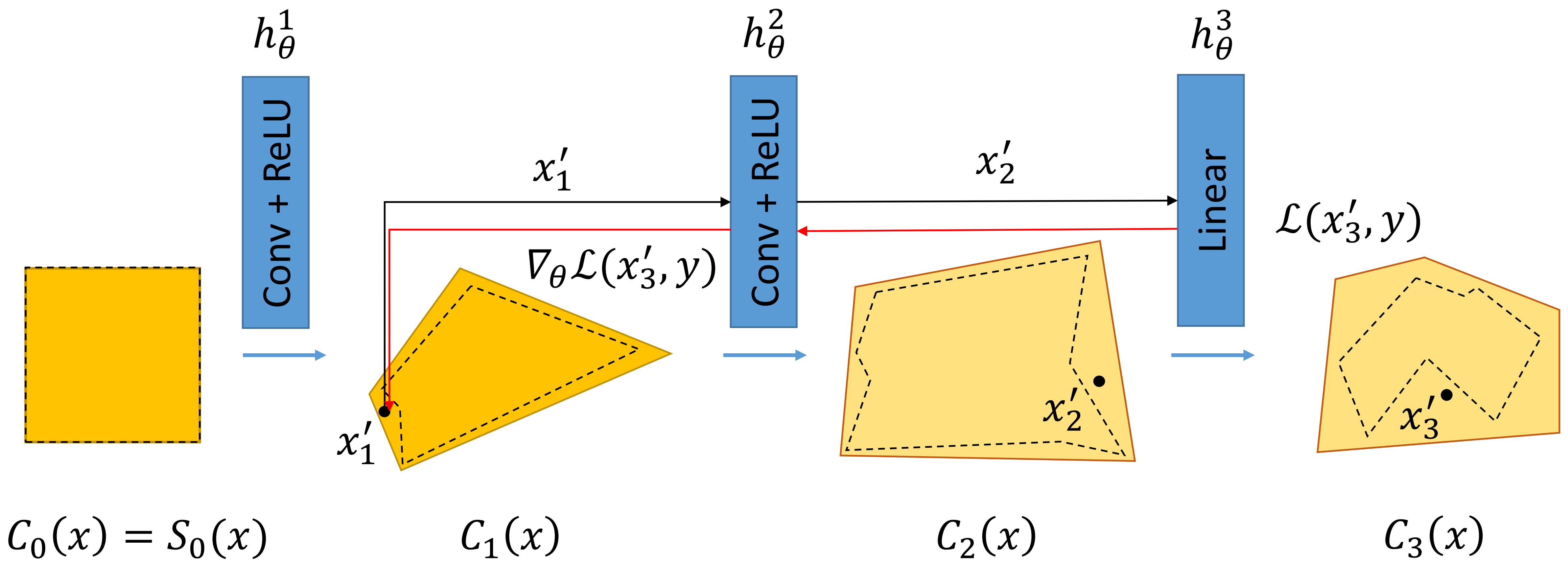
Key idea

We can find latent adversarial examples and use them for training (an instance of adversarial training)

In the first phase, we search for adversarial examples in the region $S_0(x)$ and perform adversarial training on those, which is equivalent with Madry et al. (2017)

In the next phases, we search for latent adversarial examples in regions $C_1(x), C_2(x), C_3(x)$ and perform adversarial training using those examples

Fixing Latent Adversarial Examples



Backpropagate the loss at the output
through all intermediate layers

Convex Layerwise Adversarial Training (COLT)

Algorithm 1: Convex layerwise adversarial training via convex relaxations

Data: k-layer network h_θ, training set $(\mathcal{X}, \mathcal{Y})$, learning rate η, step size α, inner steps n

Result: Certifiably robust neural network h_θ

for $l \leq k$ **do**
    for $i \leq n_{epochs}$ **do**
        Sample mini-batch $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_b, y_b)\} \sim (\mathcal{X}, \mathcal{Y})$;
        Compute convex relaxations $\mathbb{C}_l(\boldsymbol{x}_1), \mathbb{C}_l(\boldsymbol{x}_2), \ldots, \mathbb{C}_l(\boldsymbol{x}_b)$;
        Initialize $\boldsymbol{x}'_1 \sim \mathbb{C}_l(\boldsymbol{x}_1), \boldsymbol{x}'_2 \sim \mathbb{C}_l(\boldsymbol{x}_2), \ldots, \boldsymbol{x}'_b \sim \mathbb{C}_l(\boldsymbol{x}_b)$;
        for $j \leq b$ **do**
            Update in parallel n times: $\boldsymbol{x}'_j \leftarrow \Pi_{\mathbb{C}_l(\boldsymbol{x}_j)}(\boldsymbol{x}'_j + \alpha \nabla_{\boldsymbol{x}'_j} \mathcal{L}(h_\theta^{l+1:k}(\boldsymbol{x}'_j), y_j))$;
        end
        Update parameters $\theta \leftarrow \theta - \eta \cdot \frac{1}{b} \sum_{j=1}^{b} \nabla_\theta \mathcal{L}(h_\theta^{l+1:k}(\boldsymbol{x}'_j), y_j)$;
    end
    Freeze parameters θ_{l+1} of layer function h_θ^{l+1};
end

Instantianting the framework

Algorithm 1 can be instantiated using any convex relaxation, for example:

- Box (Mirman et al. 2018, Gowal et al. 2018)
- Zonotope/FastLin (Wong et al. 2018, Zhang et al. 2018, Singh et al. 2018)
- CROWN/DeepPoly (Zhang et al. 2019, Singh et al. 2020)

To apply our algorithm in practice, we need to perform **projection** on the convex set induced by the relaxation

Zonotope relaxation

Each convex region is represented as a set

$$C_l(x) = \{a_l + A_l e \mid e \in [-1, 1]^{m_l}\}$$

a_l - center of the convex set

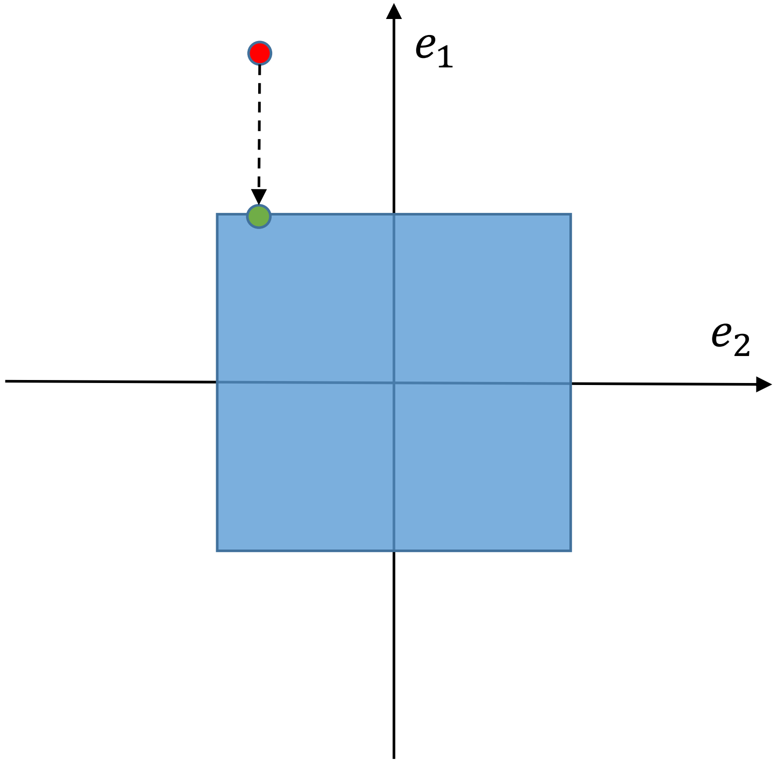
A_l - affine transformation matrix

L_∞ threat model (with radius ϵ): $a_0 = x$ and $A_0 = \epsilon I$

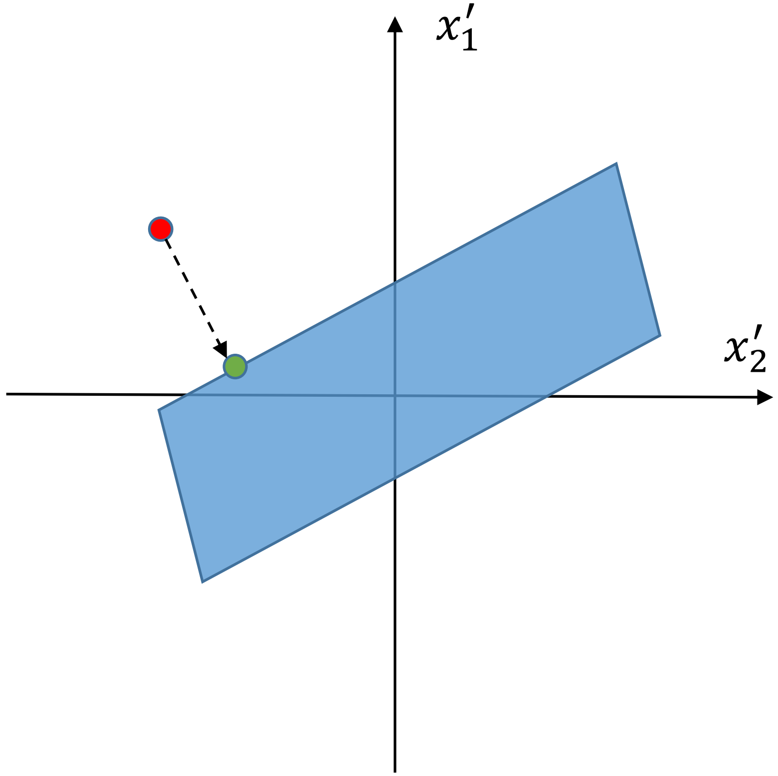
Above formulation is from Singh et al. (2018), other variants with same precision are in Wong et al. (2017) and Zhang et al. (2018)

Projection on Zonotope

Key idea: projection on Zonotope can be performed efficiently using change of variables $x' = a_l + A_l e$



$$x'_1 \coloneqq 2e_1 - e_2$$
$$x'_2 \coloneqq e_1 + e_2$$



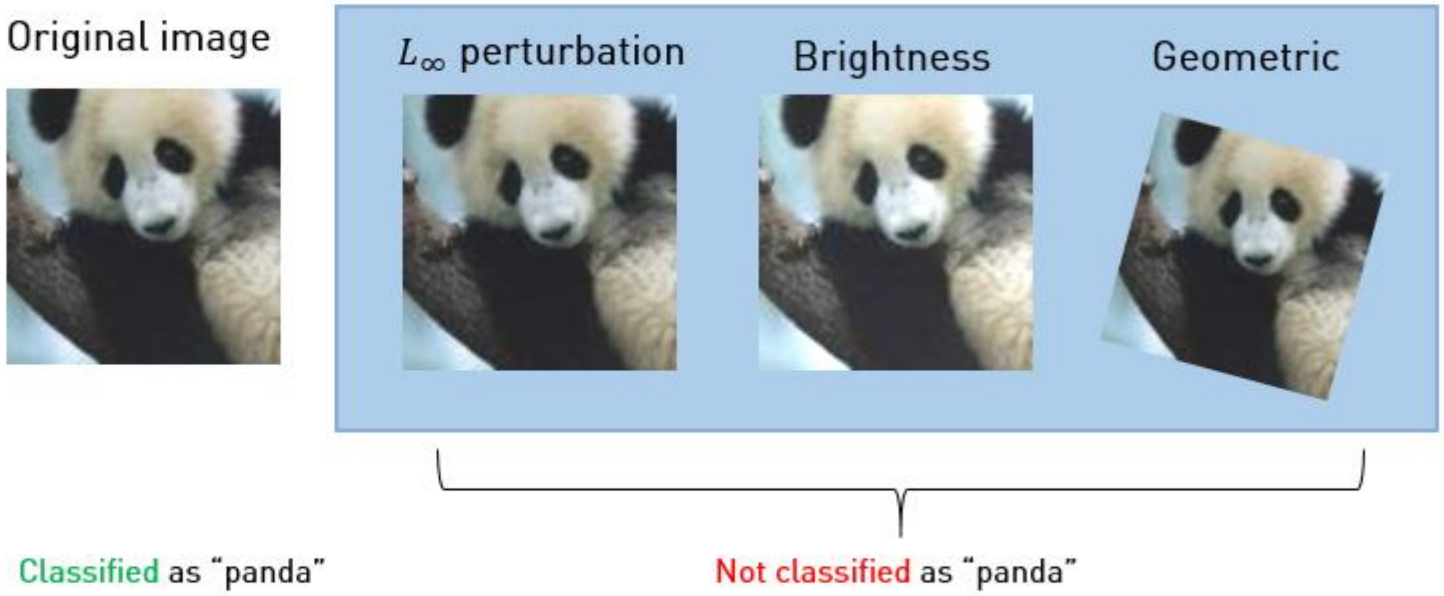
Experimental results, CIFAR-10 with 2/255 perturbation

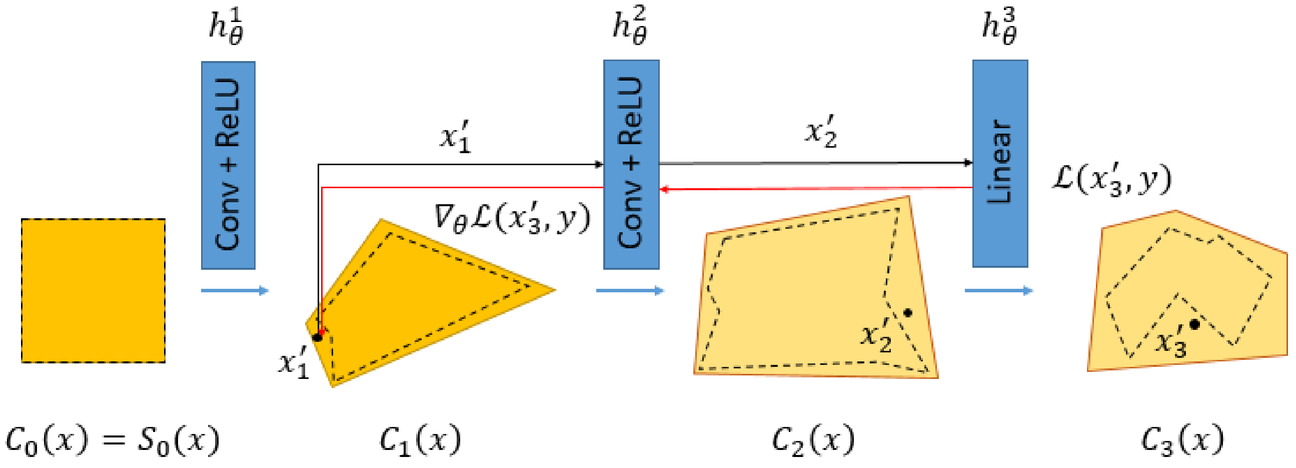
Method	Accuracy (%)	Certified Robustness (%)
Our work	78.4	60.5
Zhang et al. (2020)	71.5	54.0
Wong et al. (2018)	68.3	53.9
Gowal et al. (2018)	70.2	50.0
Xiao et al. (2019)	61.1	45.9
Mirman et al. (2019)	62.3	45.5

Experimental results, CIFAR-10 with 8/255 perturbation

Method	Accuracy (%)	Certified Robustness (%)
Our work	51.7	27.5
Zhang et al. (2020)	54.5	30.5
Mirman et al. (2019)	46.2	27.2
Wong et al. (2018)	28.7	21.8
Xiao et al. (2019)	40.5	20.3

Conclusion





Code:

http://www.sri.inf.ethz.ch