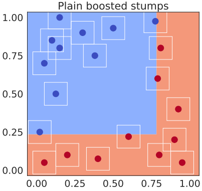
Provably Robust Boosted Decision Stumps and Trees against Adversarial Attacks

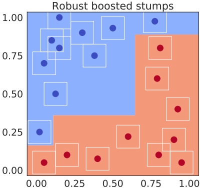
Maksym Andriushchenko (EPFL*)
Matthias Hein (University of Tübingen)

*Work done at the University of Tübingen

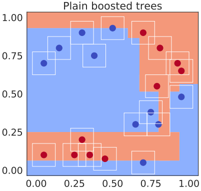
SMLD 2019, NeurIPS 2019

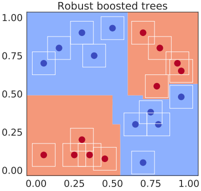
Stumps



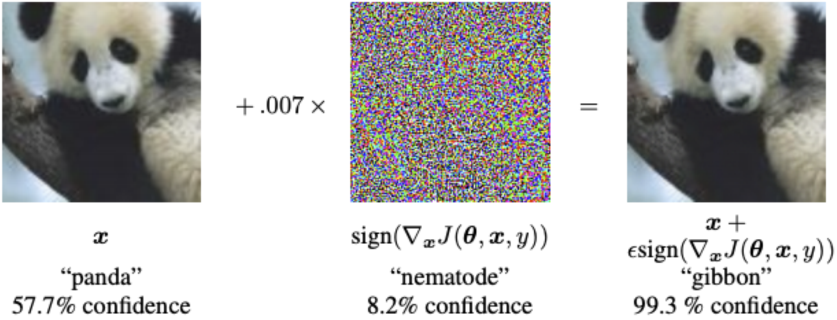


Trees



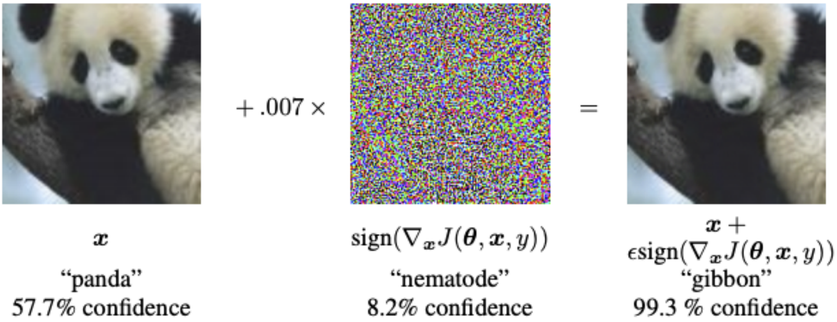


Adversarial vulnerability



Source: Goodfellow et al, "Explaining and Harnessing Adversarial Examples", 2014

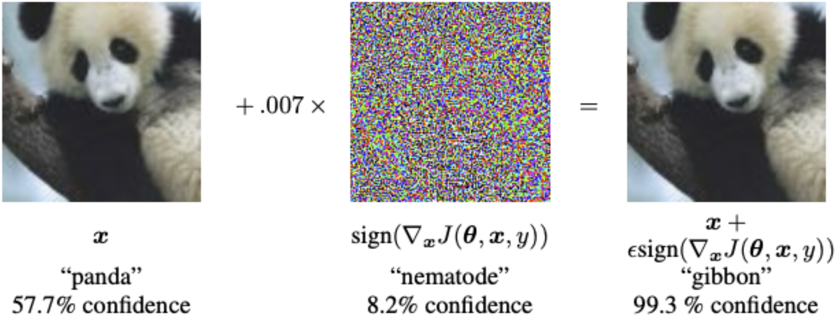
Adversarial vulnerability



Source: Goodfellow et al, "Explaining and Harnessing Adversarial Examples", 2014

- **Problem**: small changes in the input ⇒ large changes in the output

Adversarial vulnerability



Source: Goodfellow et al, "Explaining and Harnessing Adversarial Examples", 2014

- **Problem**: small changes in the input $\Rightarrow$ large changes in the output
- Topic of active research for neural networks and image recognition, but what about **other domains** and **other classifiers**?

Motivation: other domains (going beyond images)

occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
NaN	Wife	White	Female	0	1902	40	United-States	>=50k
Exec-managerial	Not-in-family	White	Male	10520	0	45	United-States	>=50k
NaN	Unmarried	Black	Female	0	0	32	United-States	<50k
Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	United-States	>=50k
Other-service	Wife	Black	Female	0	0	50	United-States	<50k

- Some input feature values can be **incorrect**: measurement noise, a human mistake, an adversarially crafted change, etc.

Motivation: other domains (going beyond images)

occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
NaN	Wife	White	Female	0	1902	40	United-States	>=50k
Exec-managerial	Not-in-family	White	Male	10520	0	45	United-States	>=50k
NaN	Unmarried	Black	Female	0	0	32	United-States	<50k
Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	United-States	>=50k
Other-service	Wife	Black	Female	0	0	50	United-States	<50k

- Some input feature values can be **incorrect**: measurement noise, a human mistake, an adversarially crafted change, etc.
- For **high-stakes** decision making, it's **necessary** to ensure a reasonable worst-case error rate under *possible* noise perturbations

Motivation: other domains (going beyond images)

occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
NaN	Wife	White	Female	0	1902	40	United-States	>=50k
Exec-managerial	Not-in-family	White	Male	10520	0	45	United-States	>=50k
NaN	Unmarried	Black	Female	0	0	32	United-States	<50k
Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	United-States	>=50k
Other-service	Wife	Black	Female	0	0	50	United-States	<50k

- Some input feature values can be **incorrect**: measurement noise, a human mistake, an adversarially crafted change, etc.
- For **high-stakes** decision making, it's **necessary** to ensure a reasonable worst-case error rate under *possible* noise perturbations
- The expected perturbation range can be specified by **domain experts**

Motivation: other classifiers

- **Our paper**: we concentrate on **boosted decision stumps and trees**

Motivation: other classifiers

- **Our paper**: we concentrate on **boosted decision stumps and trees**
- They are widely adopted in practice – implementations like **XGBoost** or **LightGBM** are used almost in every Kaggle competition

Motivation: other classifiers

- **Our paper**: we concentrate on **boosted decision stumps and trees**
- They are widely adopted in practice – implementations like **XGBoost** or **LightGBM** are used almost in every Kaggle competition
- Moreover, boosted trees are **interpretable** which is also an important practical aspect. **Who wants to deploy a black-box?**

Motivation: other classifiers

- **Our paper**: we concentrate on **boosted decision stumps and trees**
- They are widely adopted in practice – implementations like **XGBoost** or **LightGBM** are used almost in every Kaggle competition
- Moreover, boosted trees are **interpretable** which is also an important practical aspect. **Who wants to deploy a black-box?**
- $\Longrightarrow$ it is important to develop boosted trees which are **robust**, but first we need to understand the reason of their **vulnerability**

Motivation: other classifiers

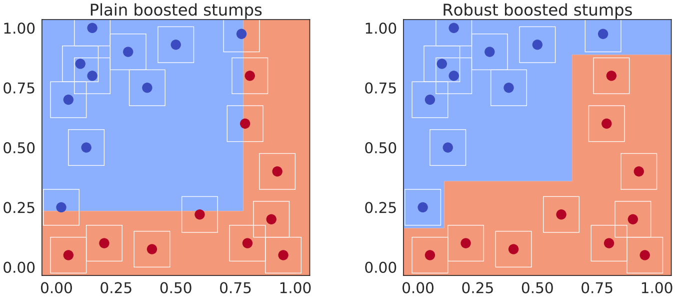
- **Our paper**: we concentrate on **boosted decision stumps and trees**
- They are widely adopted in practice – implementations like **XGBoost** or **LightGBM** are used almost in every Kaggle competition
- Moreover, boosted trees are **interpretable** which is also an important practical aspect. **Who wants to deploy a black-box?**
- $\Longrightarrow$ it is important to develop boosted trees which are **robust**, but first we need to understand the reason of their **vulnerability**

So why do adversarial examples exist?

- What goes **wrong** and how to fix it?

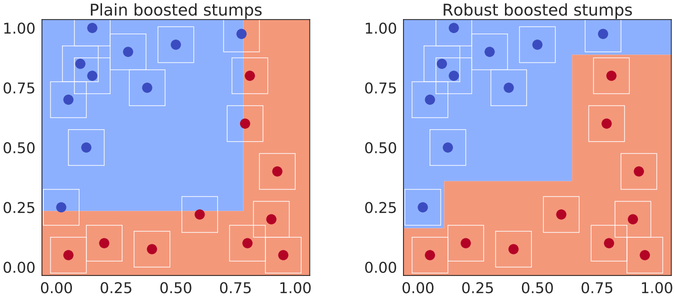
Understanding adversarial vulnerability

- What goes **wrong** and how to fix it?
- We would like to have a **large geometric margin** for every point



Understanding adversarial vulnerability

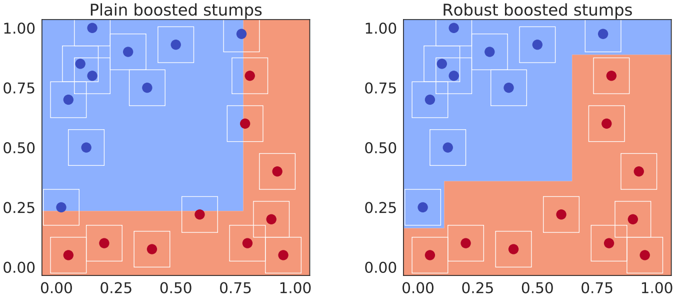
- What goes **wrong** and how to fix it?
- We would like to have a **large geometric margin** for every point



Empirical risk minimization does not distinguish the two types of solutions $\Rightarrow$ we need to use a robust objective

Understanding adversarial vulnerability

- What goes **wrong** and how to fix it?
- We would like to have a **large geometric margin** for every point



Empirical risk minimization does not distinguish the two types of solutions $\Rightarrow$ we need to use a robust objective

Let's formalize the problem!

Adversarial robustness

- What is an **adversarial example**? Consider $x \in \mathbb{R}^d$, $y \in \{-1, 1\}$, classifier $f : \mathbb{R}^d \to \mathbb{R}$, some L_p-norm threshold ϵ:

$$\min_{\delta \in \mathbb{R}^d} \; yf(x + \delta)$$
$$\|\delta\|_p \leq \epsilon, \quad x + \delta \in C$$

Adversarial robustness

- What is an **adversarial example**? Consider $x \in \mathbb{R}^d$, $y \in \{-1, 1\}$, classifier $f : \mathbb{R}^d \to \mathbb{R}$, some L_p-norm threshold ϵ:

$$\min_{\delta \in \mathbb{R}^d} \; yf(x + \delta)$$
$$\|\delta\|_p \leq \epsilon, \quad x + \delta \in C$$

- Assume x is correctly classified ($yf(x) > 0$), then $x + \delta^*$ is an **adversarial example** if $x + \delta^*$ is incorrectly classified ($yf(x + \delta^*) < 0$)

Adversarial robustness

- What is an **adversarial example**? Consider $x \in \mathbb{R}^d$, $y \in \{-1, 1\}$, classifier $f : \mathbb{R}^d \to \mathbb{R}$, some L_p-norm threshold ϵ:

$$\min_{\delta \in \mathbb{R}^d} \; yf(x+\delta)$$
$$\|\delta\|_p \leq \epsilon, \quad x + \delta \in C$$

- Assume x is correctly classified ($yf(x) > 0$), then $x + \delta^*$ is an **adversarial example** if $x + \delta^*$ is incorrectly classified ($yf(x+\delta^*) < 0$)
- How to measure robustness? **Robust test error** (RTE):

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{yf(x)<0}}_{\text{standard zero-one loss}} \quad \to \quad \underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{yf(x+\delta^*)<0}}_{\textbf{robust}\text{ zero-one loss}}$$

Adversarial robustness

- What is an **adversarial example**? Consider $x \in \mathbb{R}^d$, $y \in \{-1, 1\}$, classifier $f : \mathbb{R}^d \to \mathbb{R}$, some L_p-norm threshold ϵ:

$$\min_{\delta \in \mathbb{R}^d} \; yf(x+\delta)$$

$$\|\delta\|_p \leq \epsilon, \quad x + \delta \in C$$

- Assume x is correctly classified ($yf(x) > 0$), then $x + \delta^*$ is an **adversarial example** if $x + \delta^*$ is incorrectly classified ($yf(x + \delta^*) < 0$)
- How to measure robustness? **Robust test error** (RTE):

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{yf(x)<0}}_{\text{standard zero-one loss}} \quad \to \quad \underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{yf(x+\delta^*)<0}}_{\textbf{robust} \text{ zero-one loss}}$$

- Finding δ^*: **non-convex** opt. problem for NNs and BTs. Exact **mixed integer formulations** exist for ReLU-NNs and BTs (**slow**).

Training adversarially robust models

- **Robust optimization** problem wrt the set $\Delta(\epsilon)$:

$$\min_{\theta} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} L(f(x_i + \delta; \theta), y_i)$$

Training adversarially robust models

- **Robust optimization** problem wrt the set $\Delta(\epsilon)$:

$$\min_{\theta} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} L(f(x_i + \delta; \theta), y_i)$$

- L is a usual margin-based loss function (cross-entropy, exp. loss, etc)

Training adversarially robust models

- **Robust optimization** problem wrt the set $\Delta(\epsilon)$:

$$\min_{\theta} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} L(f(x_i + \delta; \theta), y_i)$$

- L is a usual margin-based loss function (cross-entropy, exp. loss, etc)
- $\epsilon = 0 \implies$ just well-known **Empirical Risk Minimization**

Training adversarially robust models

- **Robust optimization** problem wrt the set $\Delta(\epsilon)$:

$$\min_{\theta} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} L(f(x_i + \delta; \theta), y_i)$$

- L is a usual margin-based loss function (cross-entropy, exp. loss, etc)
- $\epsilon = 0 \implies$ just well-known **Empirical Risk Minimization**
- **Goal**: small loss ($\Rightarrow$ large margin) not only at x_i, but for every $x_i + \delta \in \Delta(\epsilon)$

Training adversarially robust models

- **Robust optimization** problem wrt the set $\Delta(\epsilon)$:

$$\min_{\theta} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} L(f(x_i + \delta; \theta), y_i)$$

- L is a usual margin-based loss function (cross-entropy, exp. loss, etc)
- $\epsilon = 0 \implies$ just well-known **Empirical Risk Minimization**
- **Goal**: small loss ($\Rightarrow$ large margin) not only at x_i, but for every $x_i + \delta \in \Delta(\epsilon)$
- **Adversarial training**: approximately solve the robust loss
  $\implies$ minimization of **a lower bound** on the objective

Training adversarially robust models

- **Robust optimization** problem wrt the set $\Delta(\epsilon)$:

$$\min_{\theta} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} L(f(x_i + \delta; \theta), y_i)$$

- L is a usual margin-based loss function (cross-entropy, exp. loss, etc)
- $\epsilon = 0 \implies$ just well-known **Empirical Risk Minimization**
- **Goal**: small loss ($\Rightarrow$ large margin) not only at x_i, but for every $x_i + \delta \in \Delta(\epsilon)$
- **Adversarial training**: approximately solve the robust loss
  $\implies$ minimization of **a lower bound** on the objective
- **Provable defenses**: upper bound the robust loss
  $\implies$ minimization of **an upper bound** on the objective

Robustness Certification and Robust Optimization for Boosted Trees

Tree ensemble: robustness certification

- The exact certification is **NP-hard** [Kantchelian et al, ICML 2016]

Tree ensemble: robustness certification

- The exact certification is **NP-hard** [Kantchelian et al, ICML 2016]
- But we can derive a tractable **lower bound** $\tilde{G}(x, y)$ on $G(x, y)$ for an ensemble of trees:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta) = \min_{\|\delta\|_\infty \leq \epsilon} \sum_{t=1}^{T} y u^{(t)}_{q_t(x+\delta)} \geq \sum_{t=1}^{T} \min_{\|\delta\|_\infty \leq \epsilon} y u^{(t)}_{q_t(x+\delta)} := \tilde{G}(x, y)$$

Tree ensemble: robustness certification

- The exact certification is **NP-hard** [Kantchelian et al, ICML 2016]
- But we can derive a tractable **lower bound** $\tilde{G}(x, y)$ on $G(x, y)$ for an ensemble of trees:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta) = \min_{\|\delta\|_\infty \leq \epsilon} \sum_{t=1}^{T} y u^{(t)}_{q_t(x+\delta)} \geq \sum_{t=1}^{T} \min_{\|\delta\|_\infty \leq \epsilon} y u^{(t)}_{q_t(x+\delta)} := \tilde{G}(x, y)$$

- $\tilde{G}(x, y) \geq 0 \implies G(x, y) \geq 0$, i.e. x is **provably robust**.

Tree ensemble: robustness certification

- The exact certification is **NP-hard** [Kantchelian et al, ICML 2016]
- But we can derive a tractable **lower bound** $\tilde{G}(x, y)$ on $G(x, y)$ for an ensemble of trees:

$$\min_{\|\delta\|_\infty \le \epsilon} yF(x+\delta) = \min_{\|\delta\|_\infty \le \epsilon} \sum_{t=1}^{T} y u^{(t)}_{q_t(x+\delta)} \ge \sum_{t=1}^{T} \min_{\|\delta\|_\infty \le \epsilon} y u^{(t)}_{q_t(x+\delta)} := \tilde{G}(x, y)$$

- $\tilde{G}(x, y) \ge 0 \implies G(x, y) \ge 0$, i.e. x is **provably robust**.
- $\tilde{G}(x, y) < 0 \implies x$ is either robust or non-robust.

Tree ensemble: robustness certification

- The exact certification is **NP-hard** [Kantchelian et al, ICML 2016]
- But we can derive a tractable **lower bound** $\tilde{G}(x, y)$ on $G(x, y)$ for an ensemble of trees:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta) = \min_{\|\delta\|_\infty \leq \epsilon} \sum_{t=1}^{T} yu^{(t)}_{q_t(x+\delta)} \geq \sum_{t=1}^{T} \min_{\|\delta\|_\infty \leq \epsilon} yu^{(t)}_{q_t(x+\delta)} := \tilde{G}(x, y)$$

- $\tilde{G}(x, y) \geq 0 \implies G(x, y) \geq 0$, i.e. x is **provably robust**.
- $\tilde{G}(x, y) < 0 \implies x$ is either robust or non-robust.
- We get an *upper bound* on the number of non-robust points, which yields an *upper bound* on the robust test error.

Tree ensemble: robustness certification

- The exact certification is **NP-hard** [Kantchelian et al, ICML 2016]
- But we can derive a tractable **lower bound** $\tilde{G}(x, y)$ on $G(x, y)$ for an ensemble of trees:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta) = \min_{\|\delta\|_\infty \leq \epsilon} \sum_{t=1}^{T} yu^{(t)}_{q_t(x+\delta)} \geq \sum_{t=1}^{T} \min_{\|\delta\|_\infty \leq \epsilon} yu^{(t)}_{q_t(x+\delta)} := \tilde{G}(x, y)$$

- $\tilde{G}(x, y) \geq 0 \implies G(x, y) \geq 0$, i.e. x is **provably robust**.
- $\tilde{G}(x, y) < 0 \implies x$ is either robust or non-robust.
- We get an *upper bound* on the number of non-robust points, which yields an *upper bound* on the robust test error.
- For a decision tree: $\min_{\|\delta\|_\infty \leq \epsilon} yu^{(t)}_{q_t(x+\delta)}$ can be found **exactly** by checking all leafs which are reachable in $B_\infty(x, \epsilon)$ ($O(l)$ time)

Tree ensemble: from certification to robust optimization

- Now we know how to lower bound the certification problem:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

Tree ensemble: from certification to robust optimization

- Now we know how to lower bound the certification problem:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

- **Does it help to solve the min-max problem?**

$$\min_{\theta} \sum_{i=1}^{n} \max_{\|\delta\|_\infty \leq \epsilon} L(f(x_i+\delta;\theta), y_i)$$

- **Yes!** For monotonically decreasing L (e.g. exp. loss):

$$\max_{\|\delta\|_\infty \leq \epsilon} L(y\,F(x+\delta)) = L\Big(\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)\Big),$$

Tree ensemble: from certification to robust optimization

- Now we know how to lower bound the certification problem:

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

- **Does it help to solve the min-max problem?**

$$\min_{\theta} \sum_{i=1}^{n} \max_{\|\delta\|_\infty \leq \epsilon} L(f(x_i+\delta;\theta), y_i)$$

- **Yes!** For monotonically decreasing L (e.g. exp. loss):

$$\max_{\|\delta\|_\infty \leq \epsilon} L(y\,F(x+\delta)) = L\Big(\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)\Big),$$

- $\Longrightarrow$ we can calculate an upper bound on the **robust loss**.

Tree ensemble: from certification to robust optimization

- Now we know how to lower bound the certification problem:

$$\min_{\|\delta\|_\infty \le \epsilon} yF(x+\delta)$$

- **Does it help to solve the min-max problem?**

$$\min_\theta \sum_{i=1}^{n} \max_{\|\delta\|_\infty \le \epsilon} L(f(x_i+\delta;\theta), y_i)$$

- **Yes!** For monotonically decreasing L (e.g. exp. loss):

$$\max_{\|\delta\|_\infty \le \epsilon} L(y\,F(x+\delta)) = L\Big(\min_{\|\delta\|_\infty \le \epsilon} yF(x+\delta)\Big),$$

- $\implies$ we can calculate an upper bound on the **robust loss**.
- Now: come up with a proper update **for a new weak learner**.

Tree ensemble: robust optimization

- The robust loss for a tree ensemble can be **upper bounded** as

$$\max_{\|\delta\|_\infty \le \epsilon} L\Big(y_i F(x_i+\delta) + y_i f(x_i+\delta)\Big) = L\Big(\min_{\|\delta\|_\infty \le \epsilon} \Big[\sum_{t=1}^{T} y_i f_t(x_i+\delta) + y_i f(x_i+\delta)\Big]\Big)$$

$$\le L\Big(\sum_{t=1}^{T} \min_{\|\delta\|_\infty \le \epsilon} y_i f_t(x_i+\delta) + \min_{\|\delta\|_\infty \le \epsilon} y_i f(x_i+\delta)\Big) = L\Big(\tilde{G}(x_i, y_i) + \min_{\|\delta\|_\infty \le \epsilon} y_i f(x_i+\delta)\Big)$$

Tree ensemble: robust optimization

- The robust loss for a tree ensemble can be **upper bounded** as

$$\max_{\|\delta\|_\infty \le \epsilon} L\Big(y_i F(x_i+\delta) + y_i f(x_i+\delta)\Big) = L\Big(\min_{\|\delta\|_\infty \le \epsilon}\Big[\sum_{t=1}^{T} y_i f_t(x_i+\delta) + y_i f(x_i+\delta)\Big]\Big)$$

$$\le L\Big(\sum_{t=1}^{T} \min_{\|\delta\|_\infty \le \epsilon} y_i f_t(x_i+\delta) + \min_{\|\delta\|_\infty \le \epsilon} y_i f(x_i+\delta)\Big) = L\Big(\tilde{G}(x_i,y_i) + \min_{\|\delta\|_\infty \le \epsilon} y_i f(x_i+\delta)\Big)$$

- For a **particular node** during the tree construction process, the **robust objective** is (I: set of points **reachable** for the current leaf):

$$\min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i,y_i) + y_i w_l + \min_{|\delta_j| \le \epsilon} y_i w_r \mathbb{1}_{x_{ij}+\delta_j \ge b}\right)$$

Tree ensemble: robust optimization

- The robust loss for a tree ensemble can be **upper bounded** as

$$\max_{\|\delta\|_\infty \leq \epsilon} L\Big(y_i F(x_i+\delta) + y_i f(x_i+\delta)\Big) = L\Big(\min_{\|\delta\|_\infty \leq \epsilon} \Big[\sum_{t=1}^{T} y_i f_t(x_i+\delta) + y_i f(x_i+\delta)\Big]\Big)$$

$$\leq L\Big(\sum_{t=1}^{T} \min_{\|\delta\|_\infty \leq \epsilon} y_i f_t(x_i+\delta) + \min_{\|\delta\|_\infty \leq \epsilon} y_i f(x_i+\delta)\Big) = L\Big(\tilde{G}(x_i, y_i) + \min_{\|\delta\|_\infty \leq \epsilon} y_i f(x_i+\delta)\Big)$$

- For a **particular node** during the tree construction process, the **robust objective** is (I: set of points **reachable** for the current leaf):

$$\min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i, y_i) + y_i w_l + \min_{|\delta_j| \leq \epsilon} y_i w_r \mathbb{1}_{x_{ij}+\delta_j \geq b}\right)$$

- How to solve the **minimization problem**? Just a case distinction:

$$\min_{|\delta_j| \leq \epsilon} y_i w_r \mathbb{1}_{x_{ij}+\delta_j \geq b} = y_i w_r \cdot \begin{cases} 1 & \text{if } b - x_{ij} < -\epsilon \text{ or } (|b - x_{ij}| \leq \epsilon \text{ and } y_i w_r < 0) \\ 0 & \text{if } b - x_{ij} > \epsilon \quad \text{ or } (|b - x_{ij}| \leq \epsilon \text{ and } y_i w_r \geq 0) \end{cases}$$

Tree ensemble: robust optimization

- Denoting the case distinction as $\mathbb{1}(x_i, y_i; w_r)$, our final robust objective is:

$$L^*(j, b) = \min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i, y_i) + y_i w_l + y_i w_r \mathbb{1}(x_i, y_i; w_r)\right)$$

Tree ensemble: robust optimization

- Denoting the case distinction as $\mathbb{1}(x_i, y_i; w_r)$, our final robust objective is:

$$L^*(j, b) = \min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i, y_i) + y_i w_l + y_i w_r \mathbb{1}(x_i, y_i; w_r)\right)$$

- The minimization wrt w_l, w_r can be done using **coordinate descent** (the objective is **convex** in w_l, w_r)

Tree ensemble: robust optimization

- Denoting the case distinction as $\mathbb{1}(x_i, y_i; w_r)$, our final robust objective is:

$$L^*(j, b) = \min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i, y_i) + y_i w_l + y_i w_r \mathbb{1}(x_i, y_i; w_r)\right)$$

- The minimization wrt w_l, w_r can be done using **coordinate descent** (the objective is **convex** in w_l, w_r)
- **Important**: we are guaranteed to decrease the robust loss after every weak learner

Tree ensemble: robust optimization

- Denoting the case distinction as $\mathbb{1}(x_i, y_i; w_r)$, our final robust objective is:

$$L^*(j, b) = \min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i, y_i) + y_i w_l + y_i w_r \mathbb{1}(x_i, y_i; w_r)\right)$$

- The minimization wrt w_l, w_r can be done using **coordinate descent** (the objective is **convex** in w_l, w_r)
- **Important**: we are guaranteed to decrease the robust loss after every weak learner
- **Complexity**: $O(n^2)$, while XGBoost has $O(n \log n)$

- Denoting the case distinction as $\mathbb{1}(x_i, y_i; w_r)$, our final robust objective is:

$$L^*(j,b) = \min_{w_l, w_r \in \mathbb{R}} \sum_{i \in I} L\left(\tilde{G}(x_i, y_i) + y_i w_l + y_i w_r \mathbb{1}(x_i, y_i; w_r)\right)$$

- The minimization wrt w_l, w_r can be done using **coordinate descent** (the objective is **convex** in w_l, w_r)
- **Important**: we are guaranteed to decrease the robust loss after every weak learner
- **Complexity**: $O(n^2)$, while XGBoost has $O(n \log n)$

That's it for boosted trees
Now what is so special about boosted stumps (one-level trees)?

Results for boosted stumps

- **The certification problem** can be solved **exactly**!

$$\min_{\|\delta\|_\infty \le \epsilon} yF(x+\delta)$$

Results for boosted stumps

- **The certification problem** can be solved **exactly**!

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

- **Proof idea**: the objective is separable over each dimension $\implies$ just solve d simple one-dimensional optimization problems

Results for boosted stumps

- **The certification problem** can be solved **exactly**!

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

- **Proof idea**: the objective is separable over each dimension $\Longrightarrow$ just solve d simple one-dimensional optimization problems
- As a result, **the robust loss** can be also calculated **exactly**

$$\max_{\delta \in \Delta_\infty(\epsilon)} L(y\,F(x+\delta)) = L\Big(\min_{\delta \in \Delta_\infty(\epsilon)} yF(x+\delta)\Big),$$

Results for boosted stumps

- **The certification problem** can be solved **exactly**!

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

- **Proof idea**: the objective is separable over each dimension $\implies$ just solve d simple one-dimensional optimization problems
- As a result, **the robust loss** can be also calculated **exactly**

$$\max_{\delta \in \Delta_\infty(\epsilon)} L(y\,F(x+\delta)) = L\Big(\min_{\delta \in \Delta_\infty(\epsilon)} yF(x+\delta)\Big),$$

- Moreover, we also derive an efficient update of the ensemble.

Results for boosted stumps

- **The certification problem** can be solved **exactly**!

$$\min_{\|\delta\|_\infty \leq \epsilon} yF(x+\delta)$$

- **Proof idea**: the objective is separable over each dimension $\implies$ just solve d simple one-dimensional optimization problems
- As a result, **the robust loss** can be also calculated **exactly**

$$\max_{\delta \in \Delta_\infty(\epsilon)} L(y\,F(x+\delta)) = L\Big(\min_{\delta \in \Delta_\infty(\epsilon)} yF(x+\delta)\Big),$$

- Moreover, we also derive an efficient update of the ensemble.
- $\implies$ **interesting result** since previously exact certification and robust optimization was known only for linear classifiers

Experiments

Experiments

Dataset	# classes	# features	# train	# test	Reference
breast-cancer	2	10	546	137	Dua and Graff (2017)
diabetes	2	8	614	154	Smith et al. (1988)
cod-rna	2	8	59535	271617	Uzilov et al. (2006)
MNIST 1-5	2	784	12163	2027	LeCun (1998)
MNIST 2-6	2	784	11876	1990	LeCun (1998)
FMNIST shoes	2	784	12000	2000	Xiao et al. (2017)
GTS 100-rw	2	3072	4200	1380	Stallkamp et al. (2012)
GTS 30-70	2	3072	2940	930	Stallkamp et al. (2012)
MNIST	10	784	60000	10000	LeCun (1998)
FMNIST	10	784	60000	10000	Xiao et al. (2017)
CIFAR-10	10	3072	50000	10000	Krizhevsky (2009)

- We test our methods on various datasets, including some image classification datasets (to compare to the literature).

Experiments

Dataset	# classes	# features	# train	# test	Reference
breast-cancer	2	10	546	137	Dua and Graff (2017)
diabetes	2	8	614	154	Smith et al. (1988)
cod-rna	2	8	59535	271617	Uzilov et al. (2006)
MNIST 1-5	2	784	12163	2027	LeCun (1998)
MNIST 2-6	2	784	11876	1990	LeCun (1998)
FMNIST shoes	2	784	12000	2000	Xiao et al. (2017)
GTS 100-rw	2	3072	4200	1380	Stallkamp et al. (2012)
GTS 30-70	2	3072	2940	930	Stallkamp et al. (2012)
MNIST	10	784	60000	10000	LeCun (1998)
FMNIST	10	784	60000	10000	Xiao et al. (2017)
CIFAR-10	10	3072	50000	10000	Krizhevsky (2009)

- We test our methods on various datasets, including some image classification datasets (to compare to the literature).
- However, our methods are primarily suitable for **tabular data**

Boosted trees: results

Dataset	l_∞ ϵ	Normal trees (standard training) TE	RTE	URTE	Adv. trained trees (with cube attack) TE	RTE	URTE	Robust trees Chen et al. [9] TE	RTE	Our robust trees (robust loss bound) TE	RTE	URTE
breast-cancer	0.3	0.7	81.0	81.8	**0.0**	27.0	27.0	0.7	13.1	0.7	**6.6**	6.6
diabetes	0.05	22.7	55.2	61.7	26.6	46.8	46.8	**22.1**	40.3	27.3	**35.7**	35.7
cod-rna	0.025	**3.4**	37.6	47.1	10.9	24.8	24.8	10.2	24.2	6.9	**21.3**	21.4
MNIST 1-5	0.3	**0.1**	90.7	96.0	1.3	9.0	9.5	0.3	2.9	0.2	**1.3**	1.4
MNIST 2-6	0.3	**0.4**	89.6	100	2.3	15.1	15.9	0.5	6.9	0.7	**3.8**	4.1
FMNIST shoes	0.1	**1.7**	99.8	99.9	5.5	14.1	14.2	3.1	13.2	3.6	**8.0**	8.1
GTS 100-rw	8/255	**0.9**	6.0	6.1	1.0	8.4	8.4	1.5	9.7	2.6	**4.7**	4.7
GTS 30-70	8/255	14.2	31.4	32.6	16.2	26.7	26.8	**11.5**	28.8	13.8	**20.9**	21.4

- Main metric: **RTE** (obtained via a mixed-integer solver)

Boosted trees: results

Dataset	l_∞ ϵ	Normal trees (standard training) TE	RTE	URTE	Adv. trained trees (with cube attack) TE	RTE	URTE	Robust trees Chen et al. [9] TE	RTE	Our robust trees (robust loss bound) TE	RTE	URTE
breast-cancer	0.3	0.7	81.0	81.8	**0.0**	27.0	27.0	0.7	13.1	0.7	**6.6**	6.6
diabetes	0.05	22.7	55.2	61.7	26.6	46.8	46.8	**22.1**	40.3	27.3	**35.7**	35.7
cod-rna	0.025	**3.4**	37.6	47.1	10.9	24.8	24.8	10.2	24.2	6.9	**21.3**	21.4
MNIST 1-5	0.3	**0.1**	90.7	96.0	1.3	9.0	9.5	0.3	2.9	0.2	**1.3**	1.4
MNIST 2-6	0.3	**0.4**	89.6	100	2.3	15.1	15.9	0.5	6.9	0.7	**3.8**	4.1
FMNIST shoes	0.1	**1.7**	99.8	99.9	5.5	14.1	14.2	3.1	13.2	3.6	**8.0**	8.1
GTS 100-rw	8/255	**0.9**	6.0	6.1	1.0	8.4	8.4	1.5	9.7	2.6	**4.7**	4.7
GTS 30-70	8/255	14.2	31.4	32.6	16.2	26.7	26.8	**11.5**	28.8	13.8	**20.9**	21.4

- Main metric: **RTE** (obtained via a mixed-integer solver)
- Better RTE on 8/8 datasets compared to adversarial training (baseline) and **Chen et al.** (ICML'19)

Boosted trees: results

Dataset	l_∞ ϵ	Normal trees (standard training) TE	RTE	URTE	Adv. trained trees (with cube attack) TE	RTE	URTE	Robust trees Chen et al. [9] TE	RTE	Our robust trees (robust loss bound) TE	RTE	URTE
breast-cancer	0.3	0.7	81.0	81.8	**0.0**	27.0	27.0	0.7	13.1	0.7	**6.6**	6.6
diabetes	0.05	22.7	55.2	61.7	26.6	46.8	46.8	**22.1**	40.3	27.3	**35.7**	35.7
cod-rna	0.025	**3.4**	37.6	47.1	10.9	24.8	24.8	10.2	24.2	6.9	**21.3**	21.4
MNIST 1-5	0.3	**0.1**	90.7	96.0	1.3	9.0	9.5	0.3	2.9	0.2	**1.3**	1.4
MNIST 2-6	0.3	**0.4**	89.6	100	2.3	15.1	15.9	0.5	6.9	0.7	**3.8**	4.1
FMNIST shoes	0.1	**1.7**	99.8	99.9	5.5	14.1	14.2	3.1	13.2	3.6	**8.0**	8.1
GTS 100-rw	8/255	**0.9**	6.0	6.1	1.0	8.4	8.4	1.5	9.7	2.6	**4.7**	4.7
GTS 30-70	8/255	14.2	31.4	32.6	16.2	26.7	26.8	**11.5**	28.8	13.8	**20.9**	21.4

- Main metric: **RTE** (obtained via a mixed-integer solver)
- Better RTE on 8/8 datasets compared to adversarial training (baseline) and **Chen et al.** (ICML'19)
- Adversarial training doesn't work well for boosted trees (the conclusion is different from the neural networks literature)

Boosted trees: results

Dataset	l_∞ ϵ	Normal trees (standard training) TE	RTE	URTE	Adv. trained trees (with cube attack) TE	RTE	URTE	Robust trees Chen et al. [9] TE	RTE	Our robust trees (robust loss bound) TE	RTE	URTE
breast-cancer	0.3	0.7	81.0	81.8	**0.0**	27.0	27.0	0.7	13.1	0.7	**6.6**	6.6
diabetes	0.05	22.7	55.2	61.7	26.6	46.8	46.8	**22.1**	40.3	27.3	**35.7**	35.7
cod-rna	0.025	**3.4**	37.6	47.1	10.9	24.8	24.8	10.2	24.2	6.9	**21.3**	21.4
MNIST 1-5	0.3	**0.1**	90.7	96.0	1.3	9.0	9.5	0.3	2.9	0.2	**1.3**	1.4
MNIST 2-6	0.3	**0.4**	89.6	100	2.3	15.1	15.9	0.5	6.9	0.7	**3.8**	4.1
FMNIST shoes	0.1	**1.7**	99.8	99.9	5.5	14.1	14.2	3.1	13.2	3.6	**8.0**	8.1
GTS 100-rw	8/255	**0.9**	6.0	6.1	1.0	8.4	8.4	1.5	9.7	2.6	**4.7**	4.7
GTS 30-70	8/255	14.2	31.4	32.6	16.2	26.7	26.8	**11.5**	28.8	13.8	**20.9**	21.4

- Main metric: **RTE** (obtained via a mixed-integer solver)
- Better RTE on 8/8 datasets compared to adversarial training (baseline) and **Chen et al.** (ICML'19)
- Adversarial training doesn't work well for boosted trees (the conclusion is different from the neural networks literature)
- The heuristic robust training of **Chen et al.** works better, but not as good as our approach

Boosted trees: results

Dataset	l_∞ ϵ	Normal trees (standard training) TE	RTE	URTE	Adv. trained trees (with cube attack) TE	RTE	URTE	Robust trees Chen et al. [9] TE	RTE	Our robust trees (robust loss bound) TE	RTE	URTE
breast-cancer	0.3	0.7	81.0	81.8	**0.0**	27.0	27.0	0.7	13.1	0.7	**6.6**	6.6
diabetes	0.05	22.7	55.2	61.7	26.6	46.8	46.8	**22.1**	40.3	27.3	**35.7**	35.7
cod-rna	0.025	**3.4**	37.6	47.1	10.9	24.8	24.8	10.2	24.2	6.9	**21.3**	21.4
MNIST 1-5	0.3	**0.1**	90.7	96.0	1.3	9.0	9.5	0.3	2.9	0.2	**1.3**	1.4
MNIST 2-6	0.3	**0.4**	89.6	100	2.3	15.1	15.9	0.5	6.9	0.7	**3.8**	4.1
FMNIST shoes	0.1	**1.7**	99.8	99.9	5.5	14.1	14.2	3.1	13.2	3.6	**8.0**	8.1
GTS 100-rw	8/255	**0.9**	6.0	6.1	1.0	8.4	8.4	1.5	9.7	2.6	**4.7**	4.7
GTS 30-70	8/255	14.2	31.4	32.6	16.2	26.7	26.8	**11.5**	28.8	13.8	**20.9**	21.4

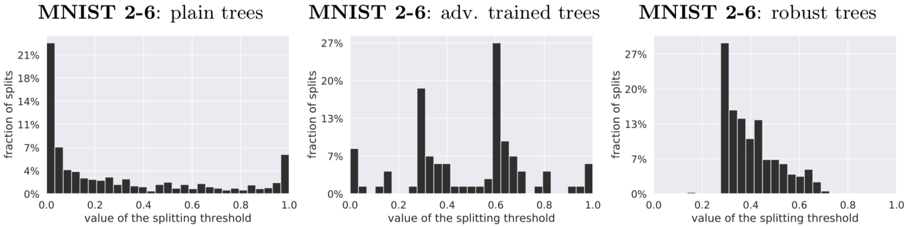
- Main metric: **RTE** (obtained via a mixed-integer solver)
- Better RTE on 8/8 datasets compared to adversarial training (baseline) and **Chen et al.** (ICML'19)
- Adversarial training doesn't work well for boosted trees (the conclusion is different from the neural networks literature)
- The heuristic robust training of **Chen et al.** works better, but not as good as our approach
- **Note**: upper bounds (URTE) are remarkably close to RTE!

Multi-class comparison to provable defenses for CNNs

Dataset	l_∞ ϵ	Approach	TE	LRTE	URTE
MNIST	0.3	Wong et al. [73]*	13.52%	26.16%	26.92%
		Xiao et al. [75]	2.67%	7.95%	19.32%
		Our robust trees, depth 30	2.68%	12.46%	12.46%
		Gowal et al. [25]	1.66%	6.12%	**8.05%**
FMNIST	0.1	Wong and Kolter [72]	21.73%	31.63%	34.53%
		Croce et al. [13]	14.50%	26.60%	30.70%
		Our robust trees, depth 30	14.15%	23.17%	**23.17%**
CIFAR-10	8/255	Xiao et al. [75]	59.55%	73.22%	79.73%
		Wong et al. [73]	71.33%	–	78.22%
		Our robust trees, depth 4	58.46%	74.69%	74.69%
		Dvijotham et al. [16]	59.38%	67.68%	70.79%
		Gowal et al. [25]	50.51%	65.23%	**67.96%**

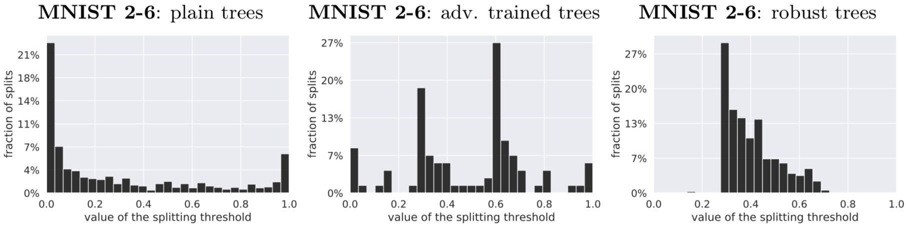
We outperform almost all provable defenses for CNNs, except one recent method (Gowal et al, 2018)!

Distribution of splitting thresholds



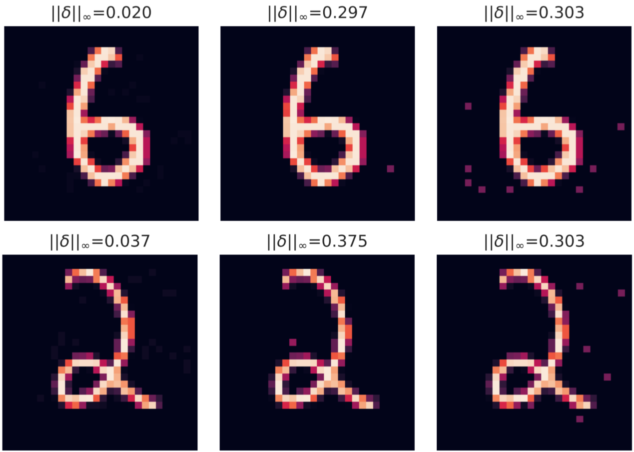
- Robust training changes the threshold distribution **dramatically**!

Distribution of splitting thresholds



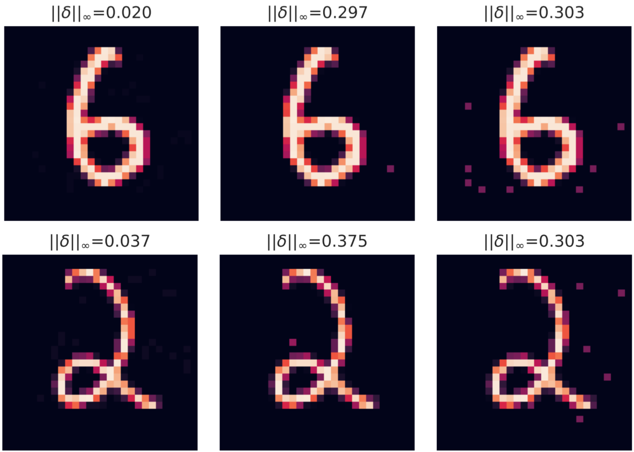
- Robust training changes the threshold distribution **dramatically**!
- Adversarial training also changes it, **but still has non-robust splits**

Adversarial examples for boosted trees



- **Models**: normal, adversarially trained, our robust boosted trees.

Adversarial examples for boosted trees



- **Models**: normal, adversarially trained, our robust boosted trees.
- Adversarial training leads to examples with $\|\delta\|_\infty < 0.3$
- **Our method** consistently leads to $\|\delta\|_\infty \geq 0.3$

Conclusions and outlook

Outlook

- Our results put the provable defenses for CNNs into a perspective $\implies$ so far **they have achieved only limited success**

Outlook

- Our results put the provable defenses for CNNs into a perspective $\Longrightarrow$ so far **they have achieved only limited success**
- **Shallow models** (i.e. no layer-wise structure) are easy to certify!

Outlook

- Our results put the provable defenses for CNNs into a perspective $\Longrightarrow$ so far **they have achieved only limited success**
- **Shallow models** (i.e. no layer-wise structure) are easy to certify!
- L_p-robustness for image data – **no applications so far**

Outlook

- Our results put the provable defenses for CNNs into a perspective $\Longrightarrow$ so far **they have achieved only limited success**
- **Shallow models** (i.e. no layer-wise structure) are easy to certify!
- L_p-robustness for image data – **no applications so far**
- **Tabular data** matters and it is ubiquitous. Real applications of L_p-robustness are rather there.

Outlook

- Our results put the provable defenses for CNNs into a perspective $\Longrightarrow$ so far **they have achieved only limited success**
- **Shallow models** (i.e. no layer-wise structure) are easy to certify!
- L_p-robustness for image data – **no applications so far**
- **Tabular data** matters and it is ubiquitous. Real applications of L_p-robustness are rather there.
- Robust **and** interpretable models are needed!

Thanks for your attention! Questions?