Robust Decision Trees Against Adversarial Examples

Honge Chen\textsuperscript{1}, Huan Zhang\textsuperscript{2}, Duane Boning\textsuperscript{1} and Cho-Jui Hsieh\textsuperscript{2}
\textsuperscript{1}MIT \textsuperscript{2}UCLA

36\textsuperscript{th} International Conference on Machine Learning (ICML)
June 11, 2019, Long Beach, CA, USA

Code (XGBoost compatible!) is available at: https://github.com/chenhongge/RobustTrees
DNNs are vulnerable to adversarial attacks

Prediction: Panda (57.7%) ✓ Imperceptible (very small) Adversarial Perturbation Prediction: Gibbon (99.3%) ✗

Goodfellow et al, *Explaining and harnessing adversarial examples*, ICLR 2015
Many **defenses** were proposed for **DNNs**:

<table>
<thead>
<tr>
<th>Literature</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Madry et al., ICLR 2018</td>
<td>Robust min-max optimization with alternative gradient descent/ascent on weights and inputs</td>
</tr>
<tr>
<td>Wong et al., ICML 2018</td>
<td><strong>Certified</strong> robust training with linear bounds by ReLU relaxation</td>
</tr>
<tr>
<td>Raghunathan et al., ICLR 2018</td>
<td><strong>Certified</strong> robust training with relaxation and Semidefinite Programming</td>
</tr>
<tr>
<td>Gowal et al., arXiv 2018</td>
<td>Fast <strong>certified</strong> robust training with interval bound propagation</td>
</tr>
<tr>
<td>Xiao et al., ICLR 2019</td>
<td><strong>Certified</strong> robust training by enforcing ReLU stability</td>
</tr>
<tr>
<td>Zhang et al., arXiv 2019</td>
<td>Stable and efficient <strong>certified</strong> robust training using tight CROWN bound and interval bound propagation</td>
</tr>
</tbody>
</table>
However, the robustness of **tree-based models** is largely unexplored...

```
<table>
<thead>
<tr>
<th>x_1</th>
<th>x_2</th>
<th>x_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;2</td>
<td>&lt;3</td>
<td>&gt;5</td>
</tr>
<tr>
<td></td>
<td>&gt;4</td>
<td></td>
</tr>
</tbody>
</table>
```

**Decission Trees**

**Tree Ensembles (GBDT/RandomForest)**

“Among the 29 challenge winning solutions published at Kaggle’s blog during 2015, 17 solutions used XGBoost.” Chen et al. KDD ‘16

![Graph showing primary ML software tools used by top-5 teams on Kaggle](https://twitter.com/fchollet/status/1113476428249464833) (April 2019)

Adversarial examples also exists in tree-based models.

Original and adversarial examples of natural GBDT models with 200 trees. Here we use a general search-based black-box attack from Cheng et al. ICLR 2019.
Why adversarial examples also exists in tree-based models?

Ordinary (natural) decision tree training finds the best split to minimize error, without considering robustness!
How to find the best split in an ordinary decision tree?

- Class 0 examples
- Class 1 examples

A decision tree node to be split

\[ x < a \]

How to find the best \( a \)?

\[ x > a \]
How to find the best split in an ordinary decision tree?

- Class 0 examples
- Class 1 examples

A decision tree node to be split

Sort by feature value

Repeat for each feature, finds the best feature and best split value

In the original (natural) decision tree training

$$j^*, \eta^* = \arg \max_{j, \eta} S(j, \eta, I)$$

Which feature to split

A score function

Split threshold

Points on the current node

Best accuracy ≠ Best robustness

10 data points with two labels, a split on feature 2 (horizontal) gives an accuracy of **80%**.
All points are close to the decision boundary and they can be perturbed to any sides of the boundary. **The worst case accuracy under perturbation is 0!**
All points are close to the decision boundary and they can be perturbed to any sides of the boundary. **The worst case accuracy under perturbation is 0!**
A better split would be on the feature 1 (vertical), which guarantees a 70% accuracy under perturbations.
In the original (natural) decision tree training

\[ j^*, \eta^* = \arg \max_{j, \eta} S(j, \eta, I) \]

Which feature to split
A score function
Split threshold
Points on the current node
Proposed robust decision tree training framework

\[ j^*, \eta^* = \arg \max_{j, \eta} RS(j, \eta, I) \]

Robust Score function 
(a maximin optimization function)

\[ RS(j, \eta, I) := \min_{I'=(x'_i, y_i)} S(j, \eta, I') \]

s.t. \( x'_i \in B_{\ell}^\infty (x_i) \), for all \( x'_i \in I' \).

Worst case score

example x perturbed in an \( \ell_\infty \) ball
It’s actually a 1D problem.
We need to optimize the worst case scenario.

However there are exponentially many possibilities...

\[ RS(j, \eta, I) := \min_{I' = \{(x'_i, y_i)\}} S(j, \eta, I') \]
\[ \text{s.t. } x'_i \in B^\infty_\epsilon(x_i), \text{ for all } x'_i \in I'. \]
• For **Information Gain or Gini Impurity** scores, there is a **closed form** solution to approximate the optimal perturbation to minimize the score.

• For general scores, we need to solve a **0-1 integer minimization** to put each point in ambiguity set to left/right leaf, which can be very slow.

\[
L_{split} = \frac{1}{2} \left[ \frac{\left( \sum_{i \in I_L} g_i \right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left( \sum_{i \in I_R} g_i \right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left( \sum_{i \in I} g_i \right)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma
\]

XGBoost’s score function

Instead, we consider **4 representative cases** to approximate the **robust score**

- Does not increase the asymptotic complexity of the original decision tree training algorithm (only a constant factor slower)

How well this approximation works?

$$RS(j, \eta, \mathcal{I}) \approx \min_{i \in \{1, 2, 3, 4\}} S(j, \eta, \mathcal{I}_i')$$
Experiments

Test accuracy

- Empirical results of robust and natural GBDT tree ensemble models on 10 datasets
- Using a general attack for non-smooth non-differentiable function (Cheng et al. ICLR 2019)
- Remarkable robustness improvement on all datasets, without harming accuracy

• MNIST models with different number of trees in GBDT
• Regardless the number of trees in the model, the robustness improvement is consistently observed.

For MNIST:
- Original image
- Natural model’s adversarial example (0.074 $\ell_\infty$ distortion)
- Robust model’s adversarial example (0.394 $\ell_\infty$ distortion)

For Fashion-MNIST:
- Original image
- Natural model’s adversarial example (0.069 $\ell_\infty$ distortion)
- Robust model’s adversarial example (0.344 $\ell_\infty$ distortion)
Does there exist a stronger attack?

Can robustness be **formally verified**?

**The robustness verification problem:**

For a point $x$, a constant $\varepsilon$, and a classifier $f(\cdot)$, does there exists an $x'$ such that $|x - x'|_\infty \leq \varepsilon$ and $f(x) \neq f(x')$?
For a point $x$, a constant $\varepsilon$, and a classifier $f(\cdot)$, does there exists an $x'$ such that $|x - x'|_\infty \leq \varepsilon$ and $f(x) \neq f(x')$?

- **minimum adversarial distortion: $\varepsilon^*$** is the smallest $\varepsilon$ such that an adversarial example exists (reflects true robustness)

- Attack algorithms find an upper bound $\varepsilon_U$ of $\varepsilon^*$

- Verification algorithms find a lower bound $\varepsilon_L$ of $\varepsilon^*$ (can guarantee that no adversarial example exists if $\varepsilon < \varepsilon_L$)
• Finding the minimum adversarial distortion $\varepsilon^*$ is \textbf{NP-complete} for general tree ensembles

• A Mixed Integer Linear Programming (MILP) based method was proposed by Kantchelian et al. (ICML 2016) and is not hopelessly slow

• MILP is the \textbf{strongest possible} attack (since it finds minimum $\varepsilon^*$)

• MILP gives robustness guarantee that with perturbation less than $\varepsilon^*$

• Finding $\varepsilon^*$ is impractical for typical large neural networks (NNs are harder to verify)

"Evasion and hardening of tree ensemble classifiers". Alex Kantchelian. J. D. Tygar, and Anthony Joseph. ICML 2016
Attack vs. MILP based verification

The same trend can be observed!

- **Remarkable verifiable** robustness improvement on all datasets
MILP can still be slow (takes days/weeks to run) if the model or dataset is large!

We recently proposed an efficient and tight robustness verification bound for tree-based models.

Robustness Verification of Tree-based Models,
Hongge Chen*, Huan Zhang*, Si Si, Yang Li, Duane Boning, and Cho-Jui Hsieh
(*equal contribution)

It’s at the SPML workshop on Friday!
Average $\ell_\infty$ distortion and running time on a **1000-tree robust GBDT model** trained with MNIST 2 vs. 6 (a binary classification)

Average $\ell_\infty$ distortion

- Attack (upper bound)
- MILP(exact)
- Chen et al. (lower bound)

Verification time per example

- Attack
- MILP

$\varepsilon_U$ vs. $\varepsilon^*$ vs. $\varepsilon = 0.98 \varepsilon^*$

25X faster than MILP
Comparing to DNNs: **verified error** on MNIST dataset with $\varepsilon=0.3$

- Unlike the minimax based adversarial training on deep training, our method uses a similar maximin robust optimization formulation but can be verified.
- Decision tree based models are **more verifiable** (fast and tight bounds exist)
- Future work: how to further improve verified error of tree based ensembles?
Conclusions

• Tree-based models are also vulnerable to adversarial examples

• Maximin robust optimization based training is effective on tree-based models

• Tree robustness can be more easily verified than DNNs
Thank You!

Code available at https://github.com/chenhongge/RobustTrees

Code is compatible with XGBoost and we plan to merge it into XGBoost upstream.


Checkout our new paper on fast robustness verification of tree-based models: https://arxiv.org/abs/1906.03849. It’s also at the SPML workshop on Friday!