Adversarially Robust Decision Tree Relabeling

Daniël Vos(✉) and Sicco Verwer

Delft University of Technology, Delft, The Netherlands
{d.a.vos,s.e.verwer}@tudelft.nl

Abstract. Decision trees are popular models for their interpretation properties and their success in ensemble models for structured data. However, common decision tree learning algorithms produce models that suffer from adversarial examples. Recent work on robust decision tree learning mitigates this issue by taking adversarial perturbations into account during training. While these methods generate robust shallow trees, their relative quality reduces when training deeper trees due the methods being greedy. In this work we propose robust relabeling, a post-learning procedure that optimally changes the prediction labels of decision tree leaves to maximize adversarial robustness. We show this can be achieved in polynomial time in terms of the number of samples and leaves. Our results on 10 datasets show a significant improvement in adversarial accuracy both for single decision trees and tree ensembles. Decision trees and random forests trained with a state-of-the-art robust learning algorithm also benefited from robust relabeling.

Keywords: Decision trees · Pruning · Adversarial examples

1 Introduction

With the increasing interest in trustworthy machine learning, decision trees have become important models [17]. Due to their simple structure humans can interpret the behavior of size-limited decision trees. Additionally, decision trees are popular for use within ensemble models where random forests [3] and particularly gradient boosting ensembles [6,7,9,15] achieve top performance on prediction tasks with tabular data. However, decision trees are optimized without considering robustness which results in models that misclassify many data points after adding tiny perturbations [14,26], i.e. adversarial examples. Therefore we are interested in training tree-based models that correctly predict data points not only at their original coordinates but also in a radius around these coordinates.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-26409-2_13.

M.-R. Amini et al. (Eds.): ECML PKDD 2022, LNAI 13715, pp. 203–218, 2023.
https://doi.org/10.1007/978-3-031-26409-2_13

Recent work has proposed decision tree learning algorithms that take adversarial perturbations into account during training to improve adversarial robustness [4,5,22]. These methods significantly improved robustness for shallow decision trees, but lacked performance for deeper trees due to their greedy nature. Optimal methods for robust decision tree learning [12,23] have also been proposed but they use combinatorial optimization solvers which makes them scale poorly in terms of both tree depth and data size. Adversarial pruning [24,25] is a method that pre-processes datasets by removing a minimal number of samples to make the dataset well-separated. While this method helps ignore samples that will only worsen robustness when predicted correctly, the learning algorithm is unchanged so the resulting models still suffer from adversarial examples. It is important to be able to train deeper robust trees as shallow trees can significantly underfit the data. Particularly in random forests where we aim to ensemble unbiased models [2] we need to be able to train very deep trees.

To improve the performance of robust decision trees we propose Robust Relabeling[1]. This post-learning procedure optimally changes the prediction labels of the decision tree leaves to maximize accuracy against adversarial examples. We assume that the user specifies an arbitrary region around each sample that represents the set of all possible perturbations of the sample. Then, we only consider a sample to be correctly predicted under adversarial attacks if there is no way for an attacker to perturb the sample such that the prediction is different from the label. We prove that in binary classification the optimal robust relabeling is induced by the minimum vertex cover of a bipartite graph. This property allows us to compute the relabeling in polynomial time in terms of the number of samples and leaves.

We compare the classification performance of decision trees and tree ensemble models on 10 datasets from the UCI Machine Learning Repository [8] and find that robust relabeling improves the average adversarial accuracy for all models. We also evaluate the performance when relabeling robust decision trees trained with a state-of-the-art method GROOT [22]. The resulting models improve adversarial accuracy compared to the default GROOT models by up to 20%. Additionally, we study the effects of standard Cost Complexity Pruning against robust relabeling. Both methods reduce the size of the learned decision trees and can improve both regular accuracy and adversarial accuracy compared to unpruned models. While, Cost Complexity Pruning performs better on regular accuracy, robust relabeling results in better adversarial accuracy.

2 Background Information

2.1 Decision Tree Learning

Decision trees are simple models that execute a series of logical tests to arrive at a prediction value. In our work, we focus on decision trees where a node k performs an operation of the type 'feature value a_k is less than or equal to some

[1] https://github.com/tudelft-cda-lab/robust-relabeling.

value b_k'. When we follow the path of such decision nodes to a leaf node t, we find the prediction value c_t.

The most popular methods to learn such decision trees are greedy algorithms that recursively create decision nodes to improve predictive accuracy. For instance, CART [1] starts by creating a root node and tests all possible combinations of feature a_k and value b_k to use for a split. These splits are all scored with the Gini impurity and the best one is selected. The samples are then sorted into a left side and right side of this split and the algorithm continues recursively on both sides until no improvements can be made or a user-defined stopping criterion is reached. While methods like CART have been hugely popular, their splitting criteria (e.g. Gini impurity or information gain) do not account for adversarial attacks. Therefore recent work has focused on modifying such algorithms in a way that the learned trees are robust to perturbations.

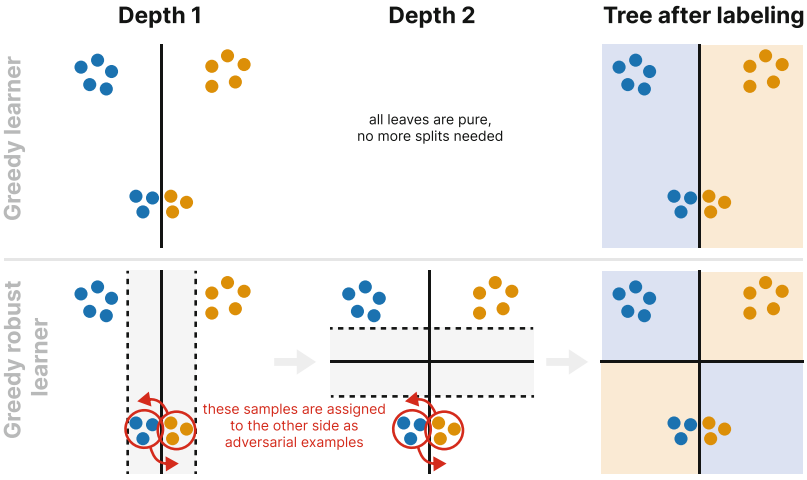


Fig. 1. Training decision trees using a (regular) greedy learner and robust greedy learner. The robust learner greedily perturbed samples close to the split which caused it to assign sub-optimal predictions to its leaves. This effect increases with the tree depth. By relabeling these leaves we can improve robustness.

2.2 Robust Decision Tree Learning

In the field of robust decision trees we usually assume that an adversary modifies our data points at test time in order to cause misclassifications. Then, we aim to train a decision tree that is maximally robust to such modifications. The type of modifications that we allow the adversary to make strongly influences the learned trees. In this work we consider an adversary that can make arbitrary changes to each test data point i within a radius ϵ of the original point. In line with previous works [5,22] we measure this distance with the l_∞ norm. Therefore the set of all possible perturbations applied to data point i is $S(i) = \{x + \delta \mid ||\delta||_\infty \leq \epsilon\}$. For

a decision tree $\mathcal{T}$ it is especially important to know what leaves $\mathcal{T}_L$ sample i can reach after applying perturbations, we refer to this set as $\mathcal{T}_L^{S(i)}$.

To improve the adversarial robustness of decision trees different methods have been proposed [4,5,22] that take adversarial perturbations into account during training. These methods are based on the same greedy algorithm that is used to train regular decision trees, but they use a different function to score the quality of splits. Then, when a locally optimal split is found, they apply perturbations to samples that are close to the split in the worst possible arrangement. This means that some number of samples that were originally on the left side of the split will be sent to the right and vice versa. Although this generally improves robustness, the fact that these samples are perturbed greedily can be detrimental to the quality of the learned tree after creating successive splits. For example in Fig. 1 the fact that samples were greedily perturbed caused the learned decision tree to create to leaves with bad predictions. Due to their greedy nature, these robust decision tree learning algorithms are successful in training shallow decision trees but they perform worse on deeper decision trees.

In recent work, optimal methods for robust decision tree learning [12,23] have also been proposed. These methods model the entire robust optimization problem and therefore do not suffer from greedy effects. However, these methods use combinatorial optimization solvers such as Mixed-Integer Linear Programming and Maximum Satisfiability. These solvers run in exponential time in terms of their input size, and the inputs grow with the size of the dataset and the depth of the tree. In practice, this means that training optimal robust decision trees on datasets of hundreds of samples is currently computationally infeasible for trees deeper than 2.

2.3 Minimum Vertex Covers and Robustness

To the best of our knowledge, Wang et al. [24] first published that for any given dataset D, there can be pairs of samples that can never be simultaneously correctly predicted against adversarial examples. For example, when considering perturbations within some radius ϵ, two samples with different labels that are within distance 2ϵ cannot both be correctly predicted when accounting for these perturbations. Given this fact, one can create a graph G with each vertex representing a sample and connect all such pairs. When we compute the minimum vertex cover of this graph, we find the minimum number of data points C to remove from D such that $D \setminus C$ can be correctly predicted. Although $D \setminus C$ *can* be correctly predicted, non-robust learning algorithms can and will still learn models that suffer from adversarial examples, e.g., because decision planes are placed too close to the remaining data points.

Wang et al. [24] used this minimum vertex cover idea by removing C from the training data in order to learn a robust nearest neighbor classifier. Adversarial pruning [25] uses a similar method to train nearest neighbor models, decision trees and tree ensembles from $D \setminus C$. The authors of ROCT [23] also used the minimum vertex cover but to compute an upper bound on adversarial accuracy which improved the time needed to train optimal robust decision trees.

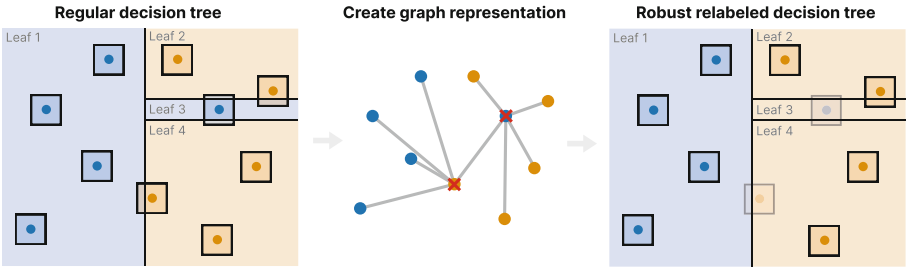


Fig. 2. Example of the robust relabeling procedure applied to a decision tree that suffers from adversarial examples. We first create a bipartite graph that connects samples with different labels that can reach the same leaf with perturbations. After removing the samples corresponding to the graph's minimum vertex cover we can relabel the decision tree to correctly predict the remaining samples. The resulting labeling is maximally robust to adversarial perturbations.

2.4 Relabeling and Pruning Decision Trees

Improving the quality of decision trees with respect to some metric by changing their leaf predictions is not a new idea. Many pruning algorithms have been proposed that remove parts of the decision tree to improve generalization. For example, Cost Complexity Pruning [1] is a widely used method that merges leaves when this improves the trade-off between the size of the tree and its predictive performance. Similarly, ideas to relabel decision trees have been used to improve performance for objectives such as fairness [13] and monotonicity [20]. Such metrics are not aligned with the objective that is optimized during training. To the best of our knowledge, we are the first to propose using leaf relabeling to improve adversarial robustness. Since relabeling methods never add new leaves, they can be seen as pruning methods since they reduce the size of the trees (after merging leaves that have the same label) (Fig. 2).

3 Robust Relabeling

Since regular decision trees and ensembles suffer from adversarial examples we are interested in post-processing the learned models to improve their robustness. In this work, we propose 'robust relabeling' (Algorithm 1) a method that keeps the decision tree structure intact but changes the predictions in the leaves to maximize adversarial accuracy. Robust relabeling is closely related to earlier works that determine minimum vertex covers to improve robustness [23–25]. In these works, the authors leverage the fact that samples with overlapping perturbation ranges and different labels can never be simultaneously classified correctly under optimal adversarial perturbations. We notice that in decision trees, two samples cannot be simultaneously classified correctly under optimal adversarial perturbations when they both reach the same leaf. Using this property we can

Algorithm 1. Robust relabeling decision tree

Input: dataset X (n samples, m features), labels y, tree leaves $\mathcal{T}_L$

$L \leftarrow \{i \mid y_i = 0\}$ ▷ $\mathcal{O}(n)$
$R \leftarrow \{i \mid y_i = 1\}$ ▷ $\mathcal{O}(n)$
$E \leftarrow \{(u, v) \mid u \in L, v \in R, \mathcal{T}_L^{S(u)} \cap \mathcal{T}_L^{S(v)} \neq \emptyset\}$ ▷ $\mathcal{O}(nm|\mathcal{T}_L| + n^2|\mathcal{T}_L|)$
$M \leftarrow$ MAXIMUM_MATCHING(L, R, E) ▷ $\mathcal{O}(n^{2.5})$
$C \leftarrow$ KŐNIG'S_THEOREM(M, L, R, E) ▷ $\mathcal{O}(n)$
for $t \in \mathcal{T}_L$ **do** ▷ $\mathcal{O}(n|\mathcal{T}_L|)$
    if $\{i \in L \mid t \in \mathcal{T}_L^{S(i)}\} \neq \emptyset$ **then**
        $c_t \leftarrow 0$
    else
        $c_t \leftarrow 1$
    end if
end for

find the smallest set of samples to remove from the dataset such that all remaining samples *can* be classified correctly under perturbations. These samples then induce a labeling of the decision tree that correctly classifies the largest possible set of samples against adversarial perturbations.

To robustly relabel a decision tree $\mathcal{T}$ we create a bipartite graph $G = (L, R, E)$ where L represents the set of samples with label $y_i = 0$ and R the set of samples with label $y_j = 1$. The set of edges E is defined by connecting all pairs of samples (i, j) that have different labels $y_i \neq y_j$ and overlapping perturbation ranges $\mathcal{T}_L^{S(i)} \cap \mathcal{T}_L^{S(j)} \neq \emptyset$. Here $S(i)$ is the set of all possible perturbations applied to data point i and $\mathcal{T}_L^{S(i)}$ is the set of leaves that are reached by $S(i)$. We find the minimum vertex cover C of G and remove the samples represented by C from the dataset. We can then relabel the decision tree to classify all remaining samples correctly even under optimal adversarial attacks.

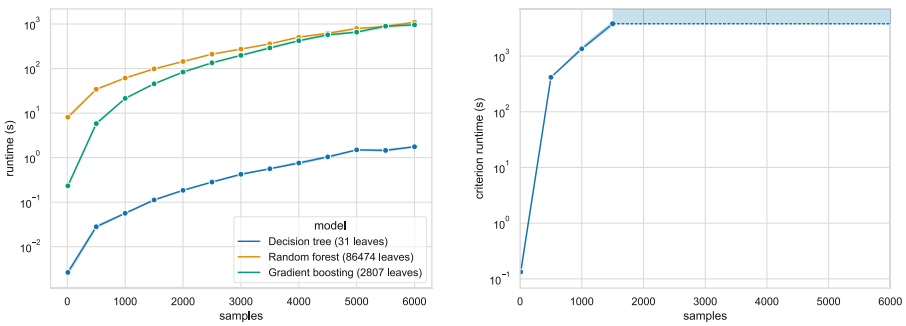
In this paper, we consider only the case where $S(i)$ describes an l_∞ radius around each data point as this is common in research on robust decision trees. However, our proof does not make use of this fact and robust relabeling can easily be extended to other attack models such as different l-norms or arbitrary sets of perturbations.

Theorem 1. *The optimal adversarially robust relabeling for decision tree leaves $\mathcal{T}_L$ is determined by the minimum vertex cover of the bipartite graph where samples i, j with different labels $y_i \neq y_j$ are represented by vertices that share an edge when their perturbations can reach any same leaf ($\mathcal{T}_L^{S(i)} \cap \mathcal{T}_L^{S(j)} \neq \emptyset$).*

Proof. For a sample i to be correctly classified by a decision tree $\mathcal{T}$, all leaves $\mathcal{T}_L^{S(i)}$ reachable by adversarial perturbations applied to X_i need to predict the correct label, i.e. $\forall t \in \mathcal{T}_L^{S(i)}, c_t = y_i$ (otherwise an adversarial example exists). Given two samples i, j with different labels $y_i \neq y_j$ and overlapping sets of

reachable leaves by perturbations $\mathcal{T}_L^{S(i)} \cap \mathcal{T}_L^{S(j)} \neq \emptyset$ these samples cannot be correctly robustly predicted as there exists a leaf t that is in both sets $\mathcal{T}_L^{S(i)}$ and $\mathcal{T}_L^{S(j)}$ and that misclassifies one of the samples. Create the bipartite graph $G = (L, R, E)$ where $L = \{i \mid y_i = 0\}$, $R = \{i \mid y_i = 1\}$ and $E = \{(u, v) \mid u \in L, v \in R, \mathcal{T}_L^{S(u)} \cap \mathcal{T}_L^{S(v)} \neq \emptyset\}$. By removing the minimum vertex cover C from G, we are left with the largest graph $G' = (L \setminus C, R \setminus C, \emptyset)$ for which no edges remain. Since none of the remaining vertices (representing samples) share an edge we are able to set $\forall t \in \mathcal{T}_L^{S(i)}, \forall i \in (L' \cup R') : c_t = y_i$, so all remaining samples get correctly robustly classified. Since C is of minimum cardinality the induced relabeling maximizes the adversarial accuracy. □

Where a naive relabeling algorithm would take exponential time to enumerate all $2^{|\mathcal{T}_L|}$ labelings, the above relabeling procedure runs in polynomial time in terms of the dataset size ($n \times m$ matrix) and the number of leaves $|\mathcal{T}_L|$. When building the graph G the runtime is dominated by computing the edges which takes $\mathcal{O}(nm|\mathcal{T}_L| + n^2|\mathcal{T}_L|)$ time. This is because we first build a mapping for each sample i to their reachable leaves $\mathcal{T}_L^S(i)$ in $\mathcal{O}(nm|\mathcal{T}_L|)$ time, then compute samples that reach any same leaf in $\mathcal{O}(n^2|\mathcal{T}_L|)$ time. Given the bipartite graph G we use the Hopcroft-Karp algorithm [11] to compute a maximum matching in $\mathcal{O}(n^{2.5})$ time and convert this in linear time into a minimum vertex cover using Kőnig's theorem. Combining all steps, robust relabeling runs in worst-case $\mathcal{O}(nm|\mathcal{T}_L| + n^2|\mathcal{T}_L| + n^{2.5})$ time.



(a) Robust relabeling

(b) Relabeling criterion tree

Fig. 3. Runtime of robust relabeling and relabeling criterion trees on samples of the Wine dataset. While decision tree relabeling runs in within seconds, relabeling ensembles takes more time due to the number of trees and increase in tree size. The runtime for relabeling criterion trees quickly increases with the number of samples.

3.1 Robust Relabeling as Splitting Criterion

While the robust relabeling procedure described before provides an intuitive use case as a post-processing step for decision tree learners, we can also use the procedure to select splits during learning. In greedy decision tree learning the learner finds a locally optimal split, partitions the samples into a left and right set (including perturbed samples) and continues this process recursively. While this approach finds an optimal split for the top decision node, the detrimental effect of choosing splits greedily increases with the depth of the tree. We will try to reduce the impact of greedily perturbing samples by using the robust relabeling procedure. To do this we can consider all samples each time we score a split and use the cardinality of the maximum matching M as a splitting criterion. By choosing splits that minimize this criterion we are then directly optimizing the adversarial accuracy of the decision tree. The pseudo-code for this algorithm is given in the appendix. We will refer to this method as relabeling criterion trees.

3.2 Runtime Comparison

We compare the runtimes of robust relabeling and relabeling criterion trees on different sample sizes of the Wine dataset in Fig. 3. Robust relabeling decision trees runs in a matter of seconds since the number of trees is small. In tree ensembles where there are 100 trees to relabel and many more leaves the runtime quickly increases. We find that relabeling criterion trees take more than an hour to train on 2000 samples, training on larger sample sizes quickly becomes infeasible.

In this work all experiments ran without parallelism on a laptop with 16 GB of RAM and a 2 GHz Quad-Core Intel Core i5 CPU. All results in this paper took approximately a day to compute, this is including robustness verification with combinatorial optimization solvers. Particularly robust relabeling criterion trees and robustness verification of tree ensembles for the wine dataset require much runtime. Without robust relabeling criterion trees and wine robustness verification the runtime is approximately 2 h.

4 Improving Robustness

To investigate the effect of robust relabeling on adversarial robustness, we compare performance on 10 datasets with a fixed perturbation radius for each dataset. We used datasets from the UCI Machine Learning Repository [8] retrieved through OpenML [19]. All datasets, their properties and perturbation radii ϵ are listed in Table 1. We pre-process each dataset by scaling the features to the range $[0, 1]$. This way, we can interpret ϵ as representing a fraction of each feature's range. We compare robust relabeling to regular decision trees and ensembles trained with Scikit-learn [16], robust decision trees trained with GROOT [22] and adversarial pruning [25]. All adversarial accuracy scores were

Table 1. Summary of datasets used. Features are scaled to $[0, 1]$ so the l_∞ perturbation radius ϵ represents a fraction of each feature's range.

Dataset	ϵ	Samples	Features	Majority class
Banknote-authentication	.05	1,372	4	.56
Breast-cancer-diagnostic	.05	569	30	.63
Breast-cancer	.1	683	9	.65
Connectionist-bench-sonar	.05	208	60	.53
Ionosphere	.05	351	34	.64
Parkinsons	.05	195	22	.75
Pima-Indians-diabetes	.01	768	8	.65
Qsar-biodegradation	.05	1,055	41	.66
Spectf-heart	.005	349	44	.73
Wine	.025	6,497	11	.63

computed with optimal adversarial attacks using the GROOT toolbox[2]. For single trees, computing optimal adversarial attacks is done by enumerating all the leaves and for tree ensembles by solving the Mixed-Integer Linear Programming formulation by Kantchelian et al. [14] using GUROBI 9.1 [10].

4.1 Decision Trees

Decision trees have the desirable property that they are interpretable when constrained to be small enough. What exactly is the number of leaves that allow a decision tree to be interpretable is not well defined. In this work, we decide to train single trees up to a depth of 5 which enforces a maximum number of leaves of $2^5 = 32$. In Table 2 we compare the performance of regular, robust GROOT [22] trees and relabeling criterion trees defined in Sect. 3.1. We score the regular and GROOT trees before and after relabeling but we skip this step for relabeling criterion trees as this does not affect the learned tree.

Robust relabeling improves the performance of regular and GROOT trees significantly on most datasets and never reduces the mean adversarial accuracy. Relabeling criterion trees and relabeled GROOT trees performed similarly on average but relabeled GROOT trees run orders of magnitude faster.

4.2 Decision Tree Ensembles

For tasks that do not require model interpretability, it is a popular choice to ensemble multiple decision trees to create stronger models. We experiment with the robust relabeling of random forests, GROOT random forests and gradient boosting ensembles, all limited to 100 decision trees. For the gradient boosting ensembles we limit the trees to a depth of 5 to prevent overfitting. This

[2] https://github.com/tudelft-cda-lab/GROOT.

Table 2. Mean **adversarial accuracy** scores of decision trees of depth 5 on 5-fold cross validation. GROOT trees with robust relabeling and relabeling criterion trees score best against adversarial attacks. However, GROOT with relabeling runs orders of magnitude faster.

Dataset	Tree	Tree relabeled	GROOT	GROOT relabeled	Relabeling criterion
Banknote	.734 ± .077	.823 ± .035	.794 ± .049	**.824** ± .038	.811 ± .049
Breast-cancer	.874 ± .013	.903 ± .025	.912 ± .035	.922 ± .012	**.925** ± .013
Breast-cancer-d	.617 ± .158	.810 ± .026	.835 ± .013	.847 ± .014	**.851** ± .038
Sonar	.482 ± .140	.573 ± .073	.601 ± .048	**.606** ± .048	.582 ± .070
Ionosphere	.689 ± .071	.792 ± .045	.892 ± .030	.889 ± .028	**.895** ± .028
Parkinsons	.513 ± .139	.759 ± .126	.749 ± .071	.790 ± .058	**.795** ± .075
Diabetes	.708 ± .009	**.712** ± .025	.677 ± .053	**.712** ± .025	.710 ± .032
Qsar-bio	.292 ± .060	.661 ± .004	.704 ± .029	**.736** ± .050	.686 ± .014
Spectf-heart	**.840** ± .041	**.840** ± .041	.831 ± .044	.831 ± .044	.768 ± .016
Wine	.526 ± .027	.610 ± .047	**.618** ± .043	**.618** ± .052	Timeout

is not required for random forests where one purposefully ensembles low bias, high variance models [2], i.e., unconstrained decision trees. We did not compare to random forests trained with the robust relabeling criterion as this was computationally infeasible. The adversarial accuracy scores before and after robust relabeling are presented in Table 3. On the Wine dataset we only used 100 test samples to limit the runtime.

Robust relabeling increases the mean adversarial accuracy over 5-fold cross-validation on all datasets and models. On average, the GROOT random forests with robust relabeling performed best. Clearly, the combination of robust splits and robust labeling is better than regular splits and robust labeling. Additionally we find that relabeled GROOT forests (Table 3) outperform relabeled GROOT trees (Table 2) on many datasets. This is in contrast with the original GROOT paper [22]. In that paper, large values were used for ϵ that did not allow for the models to achieve significant improvements over predicting the majority class.

4.3 Adversarial Pruning

Adversarial pruning [25] is a technique that implicitly prunes models by removing samples from the training dataset that are not well separated ($D \setminus C$). This intuitively makes models more robust as the models then explicitly ignore samples that will make the models more susceptible to adversarial attacks. Using decision tree learning algorithms as-is on this dataset ($D \setminus C$), without taking robustness into account, still provides models that suffer from adversarial examples. In Table 4 we compare the adversarial robustness of models trained with adversarial pruning and robust relabeling. We notice that on many datasets, adversarial pruning only removes a small number of samples which results in models that are similar to the fragile models produced by regular decision tree learning algorithms.

Table 3. Mean **adversarial accuracy** scores of decision tree ensembles on 5-fold cross validation. GROOT trees with robust relabeling score best against adversarial attacks, relabeled regular trees perform on average similarly to robust GROOT trees that did not use relabeling.

Dataset	Boosting	Forest	GROOT forest	Boosting relabeled	Forest relabeled	GROOT forest rel
Banknote	.786 ± .072	.846 ± .032	.851 ± .037	.822 ± .052	.849 ± .039	**.862** ± .039
Breast-cancer	.873 ± .027	.908 ± .020	.946 ± .017	.937 ± .020	.930 ± .011	**.952** ± .017
Breast-cancer-d	.606 ± .070	.745 ± .035	.805 ± .025	.821 ± .019	.842 ± .022	**.847** ± .021
Sonar	.438 ± .089	.389 ± .051	.510 ± .069	**.616** ± .076	.582 ± .034	.577 ± .048
Ionosphere	.635 ± .163	.812 ± .037	.903 ± .018	.815 ± .010	.872 ± .017	**.912** ± .021
Parkinsons	.492 ± .177	.508 ± .170	.728 ± .092	.759 ± .126	.749 ± .021	**.826** ± .066
Diabetes	.596 ± .043	.668 ± .049	.703 ± .052	.697 ± .032	.729 ± .036	**.730** ± .047
Qsar-bio	.078 ± .025	.173 ± .015	.648 ± .046	.663 ± .002	.663 ± .002	**.781** ± .026
Spectf-heart	.863 ± .034	.891 ± .028	.888 ± .023	.877 ± .037	**.897** ± .025	.894 ± .029
Wine	.202 ± .044	.184 ± .050	.384 ± .051	.494 ± .081	.482 ± .090	**.606** ± .038

4.4 Accuracy Robustness Trade-Off

Since we optimize robustness by enforcing samples to be correctly classified in a region around each sample, there can be a cost in regular accuracy. In Table 5 we compare the accuracy of regular models with and without robust relabeling. We find that indeed robust relabeling reduces regular accuracy in approximately two out of three cases that we tested. However, there are also instances where accuracy actually improves, such as in the case of gradient boosting on the breast cancer datasets. We expect that robustification has a helpful regularization effect in these situations.

5 Regularizing Decision Trees

Robust relabeling regularizes decision trees and tree ensembles by changing the leaf labels to maximize adversarial robustness. To understand the regularization effect we first visualize models before and after robust relabeling on toy datasets. Additionally, we contrast the regularization effect of robust relabeling with Cost Complexity Pruning. We show that while both methods can improve test accuracy, robust relabeling favors robustness while Cost Complexity Pruning favors regular accuracy.

5.1 Toy Datasets

To understand the effects of robust relabeling we generate three two-dimensional datasets and visualize the decision regions before and after relabeling. In Fig. 4 we train decision trees, random forests and gradient boosting with 5% label noise and adversarial attacks within an l_∞ radius of $\epsilon = 0.05$. All features are scaled to the range $[0, 1]$ therefore ϵ represents 5% of each feature's range. The boosted and single decision trees were limited to a depth of 5.

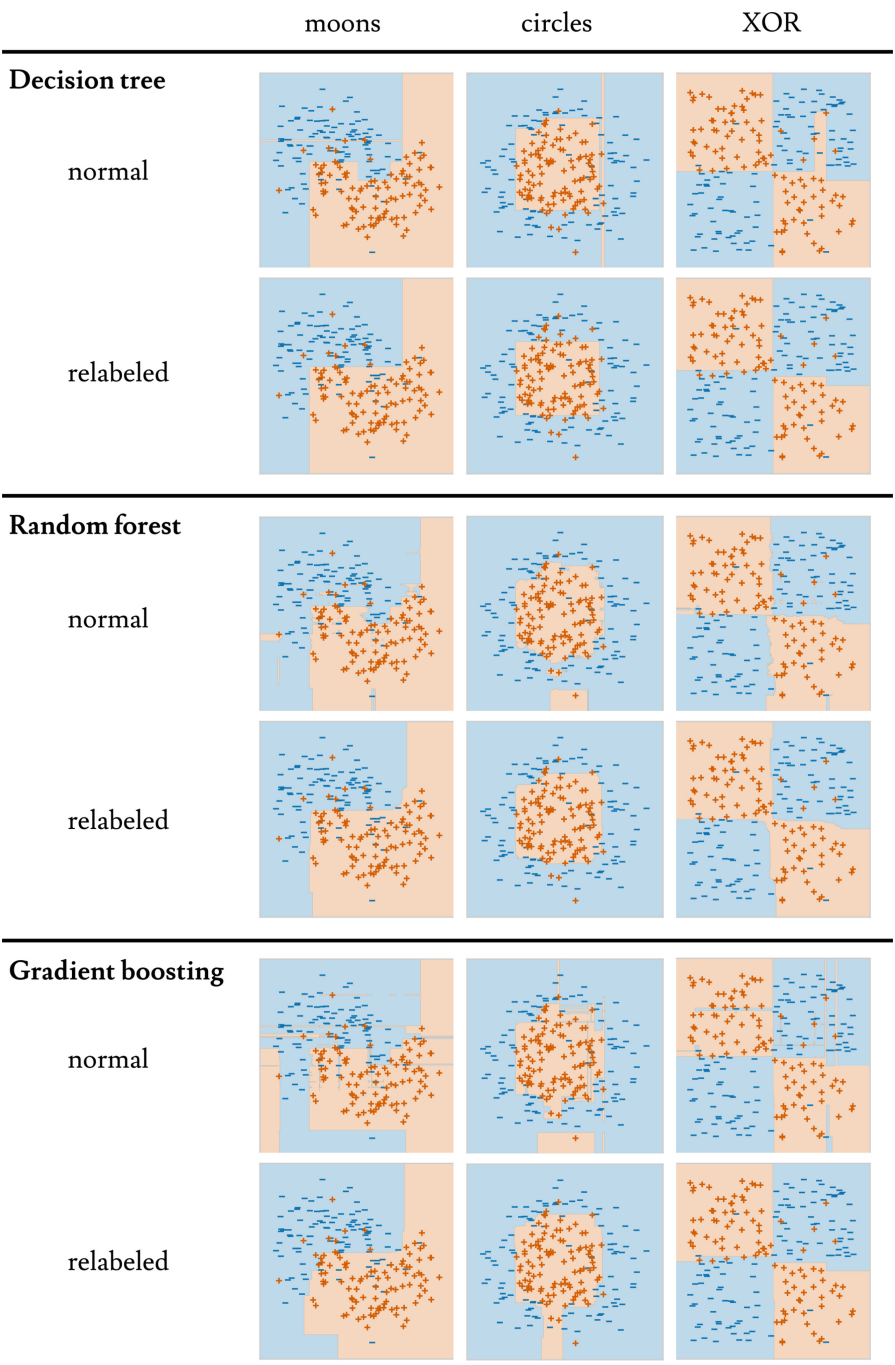


Fig. 4. Decision regions of tree models before and after robust relabeling. Robust relabeling effectively removes fragile regions resulting in visually simpler models.

Table 4. Comparison of **adversarial accuracy** scores for adversarial pruning [25] and robust relabeling (ours). Adversarial pruning does not take into account that the learner can select non-robust splits where relabeling effectively removes such splits thus producing more robust models.

Dataset	Decision tree		Gradient boosting		Random forest	
	Pruning	Relabeling	Pruning	Relabeling	Pruning	Relabeling
Banknote	.718 ± .060	**.823** ± .035	.809 ± .053	**.822** ± .052	**.855** ± .032	.849 ± .039
Breast-cancer	.867 ± .016	**.903** ± .025	.868 ± .031	**.937** ± .020	.906 ± .016	**.930** ± .011
Breast-cancer-d	.617 ± .158	**.810** ± .026	.619 ± .081	**.821** ± .019	.749 ± .035	**.842** ± .022
Sonar	.482 ± .140	**.573** ± .073	.438 ± .089	**.616** ± .076	.389 ± .051	**.582** ± .034
Ionosphere	.689 ± .071	**.792** ± .045	.635 ± .163	**.815** ± .010	.812 ± .037	**.872** ± .017
Parkinsons	.513 ± .139	**.759** ± .126	.492 ± .177	**.759** ± .126	.508 ± .170	**.749** ± .021
Diabetes	.708 ± .009	**.712** ± .025	.596 ± .043	**.697** ± .032	.668 ± .049	**.729** ± .036
Qsar-bio.	.262 ± .052	**.661** ± .004	.149 ± .024	**.663** ± .002	.183 ± .010	**.663** ± .002
Spectf-heart	**.840** ± .041	**.840** ± .041	.863 ± .034	**.877** ± .037	.891 ± .028	**.897** ± .025
Wine	.562 ± .030	**.610** ± .047	.212 ± .094	**.494** ± .081	.240 ± .047	**.482** ± .090

In all types of models we see that there are small regions with a wrong prediction in areas where the model predicts the correct label. For instance, in the normal decision tree trained on 'moons', we see a slim orange leaf in the region that is otherwise predicted as blue. This severely reduces the robustness of the model against adversarial attacks since nearby samples can be perturbed into those regions. Robust relabeling effectively removes these leaves from the models to improve adversarial robustness. Although the effect of regularization of decision trees is hard to quantify, we intuitively see that the relabeled models are more consistent in their predictions. We expect this property to also improve the explainability of the models by methods such as counterfactual explanations [21].

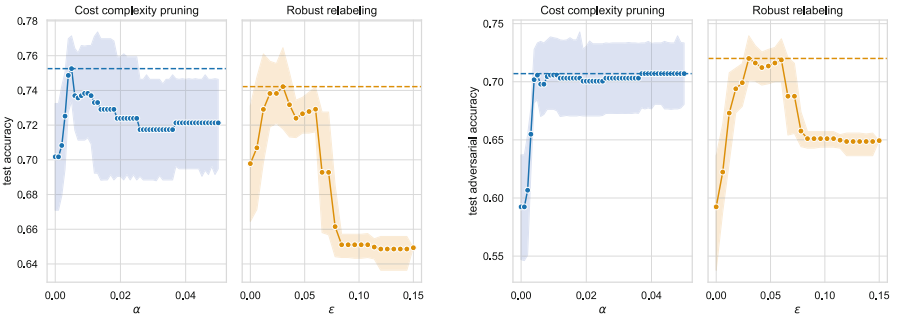
5.2 Comparison with Cost Complexity Pruning

Cost Complexity Pruning is a method that reduces the size of decision trees to improve their generalization capabilities. This method iteratively merges leaves that have a lower increase in predictive performance than some user-defined threshold α. In Fig. 5 we compare the effects of Cost Complexity Pruning and robust relabeling on the Diabetes dataset. Here, we trained decision trees without size constraints and varied the hyperparameters α and ϵ then measured accuracy before and after adversarial attacks.

While the effectiveness of cost complexity pruning and robust relabeling varies between datasets we find that generally both methods can increase test accuracy compared to the baseline model. However, cost complexity pruning achieved better accuracy scores on average while robust relabeling achieved better adversarial accuracy scores. Clearly, there is a difference between regularization for generalization and adversarial robustness. Such a trade-off between accuracy and robustness has been widely described in the literature [18,27].

Table 5. Comparison of **regular accuracy** scores before and after applying robust relabeling. Since robustness is generally at odds with accuracy robust relabeling loses out on accuracy in about 2 out of 3 cases. However, in some cases robustness actually improves accuracy as a type of regularization.

Dataset	Decision tree		Gradient boosting		Random forest	
	Before	After	Before	After	Before	After
Banknote	**.967** ± .022	.948 ± .036	**.991** ± .007	.941 ± .038	**.994** ± .005	.956 ± .020
Breast-cancer	**.969** ± .014	.958 ± .019	.962 ± .008	**.965** ± .009	.968 ± .008	**.969** ± .003
Breast-cancer-d	**.930** ± .034	.912 ± .032	.917 ± .036	**.931** ± .024	**.954** ± .017	.947 ± .028
Sonar	**.740** ± .058	.716 ± .054	.731 ± .044	**.740** ± .046	**.803** ± .031	.755 ± .054
Ionosphere	**.883** ± .034	.857 ± .047	**.906** ± .041	.863 ± .022	.926 ± .026	**.932** ± .033
Parkinsons	**.872** ± .065	.851 ± .071	**.867** ± .071	.851 ± .071	**.913** ± .082	.759 ± .014
Diabetes	.737 ± .026	**.738** ± .034	**.742** ± .026	.719 ± .028	**.769** ± .039	.768 ± .040
Qsar-bio	**.819** ± .042	.661 ± .004	**.872** ± .023	.663 ± .002	**.868** ± .023	.663 ± .002
Spectf-heart	**.840** ± .041	**.840** ± .041	.871 ± .028	**.880** ± .033	**.897** ± .025	**.897** ± .025
Wine	**.696** ± .048	.686 ± .047	**.774** ± .024	.494 ± .081	**.786** ± .019	.498 ± .080



(a) Test accuracy (b) Test accuracy against $\epsilon = 0.01$ attacks

Fig. 5. Test scores on the Diabetes dataset when varying the hyperparameters of cost complexity pruning and robust relabeling. Both improve upon unpruned trees ($\alpha = \epsilon = 0$) but cost complexity pruning performs better at regular accuracy while robust relabeling enhances adversarial accuracy.

6 Conclusions

In this work, we studied relabeling as a method to improve the adversarial robustness of decision trees and their ensembles. As training optimal robust decision trees is expensive and training heuristic robust trees inexact, we propose a polynomial-time post-learning algorithm to overcome these problems: robust relabeling. Our results show that robust relabeling significantly improves the robustness of regular and robust tree models. Robustly relabeling models trained with the state-of-the-art robust tree heuristic GROOT further improved the performance. While we can also use the robust relabeling method during the

tree learning procedure this took up to hours of runtime and produced decision trees that were approximately as robust as relabeled GROOT trees.

We expect robust relabeling in combination with methods such as GROOT to become important for training models that get deployed in adversarial contexts such as fraud or malware detection. The result that finding an optimal robust labeling can be done in polynomial time can help to further improve methods for optimal robust decision tree learning. In future work, we will explore the regularity effects of robust models for instance for improved counterfactual explanations.

References

1. Breiman, L., Friedman, J., Stone, C., Olshen, R.: Classification and Regression Trees. Taylor & Francis, Milton Park (1984)
2. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
4. Calzavara, S., Lucchese, C., Tolomei, G., Abebe, S.A., Orlando, S.: Treant: training evasion-aware decision trees. Data Min. Knowl. Disc. **34**(5), 1390–1420 (2020)
5. Chen, H., Zhang, H., Boning, D., Hsieh, C.J.: Robust decision trees against adversarial examples. In: International Conference on Machine Learning, pp. 1122–1131. PMLR (2019)
6. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016)
7. Dorogush, A.V., Ershov, V., Gulin, A.: CatBoost: gradient boosting with categorical features support. arXiv preprint arXiv:1810.11363 (2018)
8. Dua, D., Graff, C.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
9. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. 1189–1232 (2001)
10. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2022). https://www.gurobi.com
11. Hopcroft, J.E., Karp, R.M.: An n˜5/2 algorithm for maximum matchings in bipartite graphs. SIAM J. Comput. **2**(4), 225–231 (1973)
12. Justin, N., Aghaei, S., Gomez, A., Vayanos, P.: Optimal robust classification trees. In: The AAAI-22 Workshop on Adversarial Machine Learning and Beyond (2021)
13. Kamiran, F., Calders, T., Pechenizkiy, M.: Discrimination aware decision tree learning. In: 2010 IEEE International Conference on Data Mining, pp. 869–874. IEEE (2010)
14. Kantchelian, A., Tygar, J.D., Joseph, A.: Evasion and hardening of tree ensemble classifiers. In: International Conference on Machine Learning, pp. 2387–2396. PMLR (2016)
15. Ke, G., et al.: LightGBM: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
16. Pedregosa, F., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
17. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)

18. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., Madry, A.: Robustness may be at odds with accuracy. arXiv preprint arXiv:1805.12152 (2018)
19. Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: networked science in machine learning. SIGKDD Explor. **15**(2), 49–60 (2013). https://doi.org/10.1145/2641190.2641198. http://doi.acm.org/10.1145/2641190.2641198
20. Velikova, M., Daniels, H.: Decision trees for monotone price models. CMS **1**(3), 231–244 (2004)
21. Verma, S., Dickerson, J., Hines, K.: Counterfactual explanations for machine learning: a review. arXiv preprint arXiv:2010.10596 (2020)
22. Vos, D., Verwer, S.: Efficient training of robust decision trees against adversarial examples. In: International Conference on Machine Learning, pp. 10586–10595. PMLR (2021)
23. Vos, D., Verwer, S.: Robust optimal classification trees against adversarial examples. arXiv preprint arXiv:2109.03857 (2021)
24. Wang, Y., Jha, S., Chaudhuri, K.: Analyzing the robustness of nearest neighbors to adversarial examples. In: International Conference on Machine Learning, pp. 5133–5142. PMLR (2018)
25. Yang, Y.Y., Rashtchian, C., Wang, Y., Chaudhuri, K.: Robustness for non-parametric classification: a generic attack and defense. In: International Conference on Artificial Intelligence and Statistics, pp. 941–951. PMLR (2020)
26. Zhang, C., Zhang, H., Hsieh, C.J.: An efficient adversarial attack for tree ensembles. Adv. Neural. Inf. Process. Syst. **33**, 16165–16176 (2020)
27. Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., Jordan, M.: Theoretically principled trade-off between robustness and accuracy. In: International Conference on Machine Learning, pp. 7472–7482. PMLR (2019)