We thank all reviewers for their valuable comments and suggestions. Here we focus on clarifying major concerns, and will address all minor points (fix notations, typos, and improve legibility for tables and figures) in our next revision.

[R1] 1) Larger sample size: In Table A, we repeat our experiments on 5000 test examples for each dataset (or the entire test set when its size is less than 5000), 10X larger than originally reported. We highlighted the best and the second best methods. The average $\bar{r}$ are similar to Table 3 across all datasets, showing the effectiveness of our algorithm. We had to use a different machine for this larger experiment so time is not comparable, but the speedups are also similar to those in Table 3. We have 2 large datasets, HIGGS and Bosch (see reply to **[R3]-1)**). **2) Difference with prior works**: Our major novelty is to discretize the input space into a set of valid leaf tuples, on which we perform the greedy search. Table B highlights our differences. **3) Motivation:** We provide a strong attack as a tool for evaluating the robustness of tree based models. (see reply to **[R4]-1)**). **4) Figure 2 explanation:** We run each method with different number of random initial examples (x-axis). More initial examples lead to better attacks (smaller perturbation size on y-axis), but runtime cost is higher. Methods on bottom-left corner are better. We will enlarge figures and explain more.

Table A: Average ℓ_∞ and ℓ_2 perturbation of **5000** test examples on robustly trained GBDT models. **Bold** and blue highlight the best and the second best entries respectively (not including MILP). ("*" / "⋆"): Average of 1000 / 500 examples due to long running time.

Robust GBDT	SignOPT		HSJA		RBA-Appr		Cube		LT-Attack (Ours)		MILP		Ours vs. MILP	
ℓ_∞ Perturbation	$\bar{r}$	time	$\bar{r}$	time	$\bar{r}$	time	$\bar{r}$	time	$\bar{r}_{our}$	time	r^*	time	$\bar{r}_{our}/r^*$	Speedup
MNIST2-6	.588	3.06s	.470	1.30s	.671	**.137s**	.337	2.15s	**.333**	.275s	.313	177s*	1.06	641.6X
breast-cancer	**.403**	.371s	.405	.073s	.405	**.002s**	.888	.238s	.404	.002s	.401	.010s*	1.01	5.6X
covtype	.064	.540s	.080	.186s	.093	3.61s	.055	.720s	**.047**	**.047s**	.045	14min*	1.04	17164.9X
diabetes	.119	.364s	.123	.068s	.138	**.001s**	.230	.239s	**.113**	.003s	.112	.039s*	1.01	14.4X
FMNIST	.254	4.31s	.154	1.79s	.596	7.83s	.101	4.45s	**.095**	**.412s**	.076	74min*	1.25	10778.5X
HIGGS	.015	.466s	.016	.134s	.048	72.4s*	.012	.644s	**.01**	**.050s**	.009	73min⋆	1.11	87149.2X
ijcnn	.032	.353s	.030	.105s	.032	.018s	.027	.313s	**.025**	**.006s**	.022	4.24s*	1.14	759.6X
MNIST	.513	3.93s	.389	1.68s	.690	6.42s	.296	3.95s	**.290**	**.234s**	.270	20min*	1.07	5067.5X
webspam	.047	1.00s	.043	.414s	.061	.641s	.020	.756s	**.017**	**.031s**	.015	129s*	1.13	4129.4X
bosch	.343	3.28s	.337	1.42s	.533	1.22s	.158	2.49s	**.143**	**.213s**	.100	237s*	1.43	1112.0X

Robust GBDT	SignOPT		HSJA		RBA-Appr		Cube		LT-Attack (Ours)		MILP		Ours vs. MILP	
ℓ_2 Perturbation	$\bar{r}$	time	$\bar{r}$	time	$\bar{r}$	time	$\bar{r}$	time	$\bar{r}_{our}$	time	r^*	time	$\bar{r}_{our}/r^*$	Speedup
MNIST2-6	2.97	7.37s	3.32	1.28s	2.95	**.156s**	1.31	3.19s	**.971**	.438s	.762	25.0s*	1.27	57.1X
breast-cancer	.437	.711s	.449	.069s	.436	.002s	.940	.239s	**.434**	**.002s**	.431	.011s*	1.01	5.2X
covtype	.076	1.11s	.104	.196s	.137	3.26s	.096	.726s	**.062**	**.047s**	.058	9min*	1.07	11183.1X
diabetes	.142	.591s	.150	.061s	.161	**.003s**	.274	.240s	**.133**	.005s	.132	.025s*	1.01	4.8X
FMNIST	1.67	9.27s	1.34	1.64s	3.72	7.01s	.500	7.01s	**.310**	**.385s**	.233	231s*	1.33	600.8X
HIGGS	.020	.879s	.020	.128s	.085	66.5s*	.023	.580s	**.016**	**.045s**	.014	24min⋆	1.14	31715.5X
ijcnn	.033	.572s	.035	.096s	.040	.014s	.042	.307s	**.030**	**.006s**	.025	.853s*	1.20	140.3X
MNIST	3.08	9.14s	3.04	1.61s	4.07	5.11s	1.33	6.26s	**.932**	**.291s**	.670	7min*	1.39	1523.6X
webspam	.097	3.24s	.100	.431s	.148	.589s	.068	.869s	**.041**	**.034s**	.035	28.3s*	1.17	840.6X
bosch	.750	9.62s	2.33	1.54s	1.45	1.21s	.480	3.84s	**.258**	**.232s**	.214	28.0s*	1.21	120.7X

Table B: Comparisons to prior works.

	SignOPT	HSJA	Cube	RBA-Appr	**Ours**
Access Level	B-box	B-box	B-box	W-box + data	W-box
Search Space	input	input	input	training data	**leaf tuple**
Step Size	small η	small ξ	ℓ_0 boundary	N/A	**one leaf node**
Queries / iter	200	100∼632	100	N/A	∼**1** (line 203)

Table C: RF statistics in addition to Table 7.

Dataset	training set size	test set size	subsample	acc.
MNIST2-6	11,876	1,990	.8	.963
diabetes	614	154	.8	.775
FMNIST	60,000	10,000	.8	.823
higgs	10,500,000	500,000	.8	.702
ijcnn	49,990	91,701	.8	.919
bosch	946,997	236,750	.8	.994

Table D: RF results in addition to Table 8.

ℓ_2	Cube		Ours		MILP		Ours vs. MILP	
Perturbation	$\bar{r}$	time	$\bar{r}_{our}$	time	r^*	time	$\bar{r}_{our}/r^*$	Speedup
MNIST2-6	.439	2.13s	.207	.045s	.194	.071s	1.07	1.6X
diabetes	.260	.285s	.151	.003s	.146	.042s	1.03	14.X
FMNIST	.141	3.51s	.066	.080s	.066	7.44s	1.00	93.X
higgs	.015	.423s	.009	.013s	.009	6.66s	1.00	512.3X
ijcnn	.046	.336s	.028	.003s	.028	.185s	1.00	61.7X

[R3] 1) Challenging datasets: In Table 2 and 3, HIGGS contains 10.5 million training examples and the ensemble has 300 trees. We additionally added Bosch (1.2 million examples, 968 features) in Table A. Both datasets are from challenging Kaggle competitions. Our method is effective on both datasets. **2) C++/Python:** Among the baselines, we implemented RBA-Appr in C++. MILP uses a thin wrapper around the Gurobi Solver. Other methods spend majority of time on XGBoost model inference rather than Python code. For instance, on Fashion-MNIST, SignOPT, HSJA, Cube spent 72.8%, 57.3%, 73.4% of runtime in XGBoost library (C++) calls, respectively. **3) Ablation experiments:** Our ablation experiments are spread across the paper: **(a)** *Size of the neighborhood:* we compare the effect of small (NaiveFeature) and large (NaiveLeaf) neighborhood space in Table 1, and study the minimum neighborhood distance in Appendix D.3. **(b)** *Random noise optimization* also improves the solution quality. We provide baseline results in Table 1 and optimized results in Table 2 and 3. **(c)** *number of initial examples* affects both the runtime and the solution quality, and we compare the effect in Figure 2. **4) Bounding boxes:** The exact definition is $B(\mathcal{C}) = \bigcap_{i\in\mathcal{C}} B^i = \bigcap_{i\in\mathcal{C}} (l_1^i, r_1^i] \times \cdots \times \bigcap_{i\in\mathcal{C}} (l_d^i, r_d^i]$. It is the Cartesian product of the intersection on each feature dimension. **5) Why x' and a in Figure 1 are local minimums:** Decision-based attacks update solution along the decision boundary. They will be trapped at x' and a since small perturbation on both sides will increase the distance to x_0. To update from a to b, the path will be $a \to (5, 10) \to b$, but since $a \to (5, 10)$ will increase the distortion they won't find this path. Other methods such as random sampling is inefficient in a large ℓ_p ball in the order of $\|a - b\|_p$.

[R4] 1) Motivation of minimizing ℓ_p perturbation: We minimize the perturbation to find a *smallest possible* attack, to uncover the true weakness of a model. ℓ_p distance is widely used in previous attacks (Carlini, Wagner, 2017; Kantchelian et al., 2015) and its prevalence is mostly due to mathematical convenience. Small ℓ_p perturbations are usually invisible, but we agree it cannot capture many real settings. Our method can be adapted to other distance metrics: in line 8 of Alg. 1, we enumerate the distances between x_0 and a set of candidates $\mathcal{C}$ to find the minimum. This distance can be redefined. **2) Distance notation:** We will clean up notation and use $\text{dist}_p(\mathcal{C}, x_0)$ to denote the ℓ_p distance.

[R5] 1) Size of neighborhood: Thanks for the correct understanding on this trade-off. Our ablation (Table 1) and experiments (Table 2, 3) empirically show that distance 1 is sufficient for outperforming other attacks. **2) Robust to structure changes:** For each tree, its non-leaf nodes and structures are irrelevant to our algorithm as long as the leaves produce the same bounding boxes. We conduct a small experiment on adversarial training and improve the ℓ_2 robustness from .082 to .115 on diabetes dataset. **3) Random forest:** We added the remaining experiments in Table D and C.