A. Proof of Proposition 1

Proposition 1. Given a box $B = (l_1, r_1] \times \cdots \times (l_d, r_d]$ and a point $\boldsymbol{x} \in \mathbb{R}^d$. The closest ℓ_p distance ($p \in [0, \infty]$) from x to B is $\|z - x\|_p$ where:

$$z_i = \begin{cases} x_i, & l_i \leq x_i \leq u_i \\ l_i, & x_i < l_i \\ u_i, & x_i > u_i. \end{cases}$$

Proof. For $p > 0$, The goal is to minimize the following objective:

$$\min_z \|z - x\|_p^p = \min_z \sum_{i=1}^{d} |z_i - x_i|^p$$
$$\text{s.t. } l_i < z_i \leq r_i, \;\; \forall i \in [d].$$

And for $p = 0$, the objective is

$$\min_z \|z - x\|_0 = \min_z \sum_{i=1}^{d} \mathbf{I}(z_i \neq x_i)$$
$$\text{s.t. } l_i < z_i \leq r_i, \;\; \forall i \in [d].$$

where $\mathbf{I}(\cdot)$ is an indicator function. For $p = \infty$, the objective is

$$\min_z \|z - x\|_\infty = \min_z \sum_{i=1}^{d} |z_i - x_i|$$
$$\text{s.t. } l_i < z_i \leq r_i, \;\; \forall i \in [d].$$

Since each term in the summation is separable, we can consider minimizing each term in the summation signs separately. Given the constraints on z_i, the minimum is achieved at the condition specified in Eq. (3) regardless of the choice of p:

$$z_i = \begin{cases} x_i, & l_i \leq x_i \leq u_i \\ l_i, & x_i < l_i \\ u_i, & x_i > u_i. \end{cases}$$

□

B. Closed form update rule for ℓ_p Stump Ensemble Training

For exponential loss we can rewrite eq (15) as

$$\begin{aligned} \sum_{i=1}^{N-1} L(\tilde{D}(\lceil \epsilon \rceil, d)) &= \sum_{i=1}^{N-1} \gamma_i \exp(-y_i w_l) \\ &= \sum_{y_i=1} \gamma_i \exp(-w_l) + \sum_{y_i=-1} \gamma_i \exp(w_l) \end{aligned}$$

where

$$\gamma_i = L(\tilde{D}(\lceil \epsilon \rceil, d) - y_i w_l)$$

which is fixed with a fixed w_r.

And we can further derive the optimal w_l at each update step

$$\sum_{y_i=1} \gamma_i(-\exp(-w_l^*)) + \sum_{y_i=-1} \gamma_i \exp(w_l^*) = 0$$
$$\sum_{y_i=1} \gamma_i \exp(-w_l^*) = \sum_{y_i=-1} \gamma_i \exp(w_l^*)$$
$$w_l^* = \ln \frac{\sum_{y_i=1} \gamma_i}{\sum_{y_i=-1} \gamma_i}/2.$$

C. Robustness verification for ensemble trees

In this section, we provide the detail algorithm of robustness verification for ensemble trees. This algorithm is based on the robustness verification framework in (Chen et al., 2019b). In Algorithm 1, we describe the modified function `CliqueEnumerate`, which is the key procedure of this framework. The main difference is that after we form the initial set of cliques, we will recheck whether the formed cliques have intersection with the ℓ_p perturbation ball (line 18 to 22).

D. Proof of Theorem 2

Proof. By definition, we have

$$L(\tilde{D}(\lceil\epsilon\rceil, d)) = L(\min(\tilde{D}_L(\lceil\epsilon\rceil,d), \tilde{D}_R(\lceil\epsilon\rceil,d)))$$
$$= \max(L(\tilde{D}_L(\lceil\epsilon\rceil,d)), L(\tilde{D}_R(\lceil\epsilon\rceil,d))).$$

Exponential loss L is convex and monotonically increasing; $L(\tilde{D}_L(\lceil\epsilon\rceil,d))$ and $L(\tilde{D}_R(\lceil\epsilon\rceil,d))$ are both jointly convex in w_l, w_r. Note that the dynamic programming related terms become constants after they are computed, so they are irrelevant to w_l, w_r. Therefore, $L(\tilde{D}(\lceil\epsilon\rceil, d))$ and further $\sum_{i=0}^{N-1} L(\tilde{D}(\lceil\epsilon\rceil, d))$ are jointly convex in w_l, w_r. □

Dataset	ensemble stumps lr.	ensemble trees lr.	ℓ_1 training ensemble trees sample size
breast-cancer	0.4	-	-
diabetes	0.4	-	-
Fashion-MNIST shoes	0.4	1.0	5000
MNIST 1 vs. 5	0.4	1.0	5000
MNIST 2 vs. 6	0.4	1.0	5000

Table 6. **Detail settings of the experiments**. Here we report the learning rate of different training methods for ensemble stumps and trees. We also report the sample size in experiments for ensemble tree training and the scheduling length in ℓ_p robust training for ensemble stumps.

D.1. Detail settings of the experiments

Here we report the detail settings of our experiments in Table 6. For most of the experiments, we follow the learning rate settings in (Andriushchenko & Hein, 2019). For ϵ scheduling length, we empirically set to the best value near $\epsilon_p/\epsilon_{std}$ for each dataset and ϵ settings (e.g., for ℓ_1 norm training, the best schedule length is among 2, 3 and 4 epochs for $\epsilon_1 = 1.0$ and $\epsilon_{std} = 0.3$). Here the ϵ_{std} is ϵ_∞ used in (Andriushchenko & Hein, 2019). For each dataset, different methods are trained with the same group of parameters.

For ℓ_1 robust training for ensemble trees, we use a subsample of training datasets to reduce training time. On Fashion-MNIST shoes, MNIST 1 vs. 5 and MNIST 2 vs. 6 datasets, we subsample 5000 images of the selected classes from the original dataset. For ℓ_2 robust training, we subsample 1000 images of the selected classes from the original dataset.

D.2. ℓ_∞ vs. ℓ_p robust training

For a binary classifier $y = \text{sgn}(F(x))$, and a fixed ϵ, we have $\min_{\|\delta\|_p \le \epsilon} yF(x+\delta) \ge \min_{\|\delta\|_\infty \le \epsilon} yF(x+\delta)$. Therefore, the exact ℓ_∞ robust loss can be a natural upper bound of ℓ_p robust loss. This explains the close result from ℓ_∞ and ℓ_p robust training, when using the same ϵ. However, this ℓ_∞ upper bound tends to hurt the clean accuracy , which we can see from Table 4. Additionally, unlike ℓ_1 or ℓ_2 norms, it is impossible to set this ℓ_∞ perturbation to a large value (e.g., $\epsilon_\infty = 1.0$).

E. Additional experiment results

E.1. Comparison of ℓ_∞ robustness

In this section, we report the ℓ_∞ verified errors of models in Table 4. For each model in the table, we verify the models using ℓ_∞ robustness verification of decision stumps (Andriushchenko & Hein, 2019) with perturbation norm ϵ_∞. In general, Andriushchenko & Hein (2019) produces better ℓ_∞ norm verification error because it is designed for that case, but when training using our ℓ_1 robust training procedure with a larger ℓ_1, models also get relatively good ℓ_∞ robustness. Note that here we train different number of stumps for different ϵ_1(e.g. For MNIST dataset, we train 20 stumps for $\epsilon_1 = 0.3$ and 40 stumps for $\epsilon_1 = 1.0$). And for a fixed ϵ, we train the ℓ_∞ robust model with the same number of stumps with other methods when making comparisons.

Dataset			standard training	ℓ_∞ training	ℓ_1 training
name	ϵ_∞	ϵ_1	ℓ_∞ verified err.	ℓ_∞ verified err.	ℓ_∞ verified err.
breast-cancer	0.3	1.0	88.32%	10.94%	17.51%
diabetes	0.05	0.05	42.85%	35.06%	31.81%
Fashion-MNIST shoes	0.1	0.1	69.85%	11.35%	11.75%
	0.2	0.5	98.85%	19.30%	27.60%
MNIST 1 vs. 5	0.3	0.3	67.09%	4.09%	4.05%
	0.3	1.0	66.20%	3.60%	11.59%
MNIST 2 vs. 6	0.3	0.3	97.74%	8.63%	9.10%
	0.3	1.0	100.0%	8.69%	15.28%

Table 7. ℓ_∞ **robustness of ensemble decision stumps.** This table reports the ℓ_∞ robustness for the same set of models in Table 4. For each dataset, we evaluate standard models, the ℓ_∞ robust models trained using (Andriushchenko & Hein, 2019) with perturbation norm ϵ_∞, and our ℓ_p robust model with $p = 1$ and perturbation norm ϵ_1. We test the models with ℓ_∞ norm perturbation ϵ_∞. Standard test errors are omitted as they as the same as in Table 4.

E.2. ℓ_2 robust training

In Section 5 we mainly presented results for the $p = 1$ setting, however our robust training procedure works for general ℓ_p norm. In this section, we show some ℓ_2 robust training results. For each dataset, we train three models using standard training, ℓ_∞ robust training (Andriushchenko & Hein, 2019) with ℓ_∞ perturbation norm ϵ_∞, and ℓ_p robust training with $p = 2$ and ℓ_2 perturbation norm ϵ_2. And in Table 8 and 9, we report the verification results of these models from ℓ_2 verification.

Dataset			standard training		ℓ_∞ training		ℓ_2 training	
name	ϵ_∞	ϵ_2	standard err.	ℓ_2 verified err.	standard err.	ℓ_2 verified err.	standard err.	ℓ_2 verified err.
breast-cancer	0.3	0.7	0.73%	97.08%	4.37%	99.27%	8.76%	**39.42%**
Fashion-MNIST shoes	0.2	0.4	5.05%	69.85%	9.25%	81.05%	14.55%	**49.55%**
MNIST 1 vs. 5	0.3	0.8	0.59%	67.09%	1.33%	66.45%	4.44%	**36.56%**
MNIST 2 vs. 6	0.3	0.8	2.81%	97.74%	3.91%	85.52%	13.67%	**76.98%**

Table 8. ℓ_2 **robust training for ensemble stumps** In this table, we train the model with $p = 2$ and compare the results with ℓ_∞ trained models. For each dataset, we train three models using standard training, ℓ_∞ norm robust training with ϵ_∞ and ℓ_2 norm robust training with ϵ_2. And we test and compare the ℓ_2 robustness of these models using ℓ_2 robust verification.

Dataset					standard training		ℓ_∞ training (Andriushchenko & Hein, 2019)		ℓ_2 training (ours)	
name	ϵ_∞	ϵ_2	n. trees	depth	standard err.	verified err.	standard err.	verified err.	standard err.	verified err.
Fashion-MNIST shoes	0.2	0.4	3	5	8.05%	99.40%	7.65%	93.49%	17.36%	**68.23%**
breast-cancer	0.3	0.8	5	5	1.47%	97.06%	1.47%	97.79%	12.50%	**55.88%**
MNIST 1 vs. 5	0.3	0.8	3	5	2.37%	100.0%	2.12%	97.72%	23.25%	**50.54%**
MNIST 2 vs. 6	0.3	0.8	3	5	3.82%	100.0%	3.12%	100.0%	19.80%	**93.56%**

Table 9. ℓ_2 **robust training for tree ensembles.** We report standard and ℓ_2 robust test error for all the three methods. We also report ϵ_∞ and ϵ_2 for each dataset, and the number of trees in each ensemble.

Algorithm 1 Enumerating all K-cliques on a K-partite graph with ϵ_p, dimension d and example x

input : $V_1,\ V_2,\ ,\ldots,\ V_K$ are the K independent sets ("parts") of a K-partite graph; the graph is defined similarly as in Chen et al. (2019b).

for $k \leftarrow 1,\ 2,\ 3,\ \ldots,\ K$ **do**
    $U_k \leftarrow \{(A_i,\ B^{i^{(k)}})|i^{(k)} \in V_k,\ A_i = \{i^{(k)}\}\}$ `/*` U `is a set of tuples` (A, B)`, which stores a set of cliques and their corresponding boxes.` A `is the set of nodes in one clique and` B `is the corresponding box of this clique. Initially, each node in` V_k `forms a 1-clique itself. */`
end
`CliqueEnumerate(`$U_1,\ U_2,\ ,\ldots,\ U_K$`)`
Function `CliqueEnumerate(`$U_1,\ U_2,\ ,\ldots,\ U_K$`)`
    $\hat{U}_{\text{old}} \leftarrow U_1$
    for $k \leftarrow 2,\ 3,\ \ldots,\ K$ **do**
        $\hat{U}_{\text{new}} \leftarrow \emptyset$
        for $(\hat{A},\ \hat{B}) \in \hat{U}_{old}$ **do**
            for $(A,\ B) \in U_k$ **do**
                if $B \cap \hat{B} \neq \emptyset$ **then**
                    `/* A` k`-clique is found; add it as a pseudo node with the intersection of two boxes. */`
                    $\hat{U}_{\text{new}} \leftarrow \hat{U}_{\text{new}} \cup \{(A \cup \hat{A},\ B \cap \hat{B})\}$
            end
        end
        $\hat{U}_{\text{old}} \leftarrow \hat{U}_{\text{new}}$
    end
    $\hat{U} \leftarrow \emptyset$
    for $(A, B) \in \hat{U}_{new}$ **do**
        if `CheckClique(`B, d, p, ϵ_p`)` **then**
            `/* After finding all the` k`-cliques, we need to recheck whether these cliques have intersection with the` ℓ_p `perturbation ball around the example` x`. */`
            $\hat{U} \leftarrow \hat{U} \cup \{(A, B)\}$
    end
    return $\hat{U}$
end
Function `CheckClique(`$B,\ d,\ p,\ \epsilon$`)`
    $dist \leftarrow \min_{z \in B} \|z - x\|_p^p$ using Proposition 1
    if $dist < \epsilon^p$ **then**
        return false
    return true
end