Structure of the appendix

We provide additional details and experiments which could not be included in the main part. In summary,

- Sec. A contains the proofs omitted above and experiments in support of the effect of the exact projection $P_S(u)$ compared to the approximation $A(u)$ and about the expected sparsity of the steepest descent direction,
- in Sec. B we analyse our observations about catastrophic overfitting in adversarial training wrt l_1.
- Sec. C describes the algorithm of our l_1-Square Attack,
- in Sec. D we provide the details of models and attacks used in Sec. 5,
- Sec. E contains experiments with a larger threshold, on other datasets, an ablation study on the importance of tuning the parameter of the sparsity k of the updates in SLIDE.

A. Omitted proofs

In the following we provide the missing proofs from the main paper. For the convenience of the reader we state the propositions and lemmas again.

Proposition 2.1 *The projection problem (3) onto $S = B_1(x, \epsilon) \cap H$ can be solved in $O(d \log d)$ with solution*

$$z_i^* = \begin{cases} 1 & \text{for } u_i \geq x_i \text{ and } 0 \leq \lambda_e^* \leq u_i - 1 \\ u_i - \lambda_e^* & \text{for } u_i \geq x_i \text{ and } u_i - 1 < \lambda_e^* \leq u_i - x_i \\ x_i & \text{for } \lambda_e^* > |u_i - x_i| \\ u_i + \lambda_e^* & \text{for } u_i \leq x_i \text{ and } -u_i < \lambda_e^* \leq x_i - u_i \\ 0 & \text{for } u_i \leq x_i \text{ and } 0 \leq \lambda_e^* \leq -u_i \end{cases},$$

where $\lambda_e^* \geq 0$. With $\gamma \in \mathbb{R}^d$ defined as

$$\gamma_i = \max\{-x_i \text{sign}(u_i - x_i), (1 - x_i)\text{sign}(u_i - x_i)\},$$

it holds $\lambda_e^* = 0$ if $\sum_{i=1}^d \max\{0, \min\{|u_i - x_i|, \gamma_i\} \leq \epsilon$ and otherwise λ_e^* is the solution of

$$\sum_{i=1}^d \max\left\{0, \min\{|u_i - x_i| - \lambda_e^*, \gamma_i\}\right\} = \epsilon.$$

Proof. By introducing the variable $w' = z - x$ we transform (3) into

$$\max_{w' \in \mathbb{R}^d} \frac{1}{2} \left\|u - x - w'\right\|_2^2 \quad \text{s.th.} \quad \left\|w'\right\|_1 \leq \epsilon, \quad w' + x \in [0, 1]^d.$$

Moreover, by introducing $w = \text{sign}(u - x)w'$ we get

$$\max_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^d (|u_i - x_i| - w_i)^2$$

$$\text{s.th.} \quad \sum_{i=1}^d w_i \leq \epsilon, \quad w_i \geq 0, \quad \text{sign}(u_i - x_i)w_i + x_i \in [0, 1].$$

The two componentwise constraints can be summarized with

$$\gamma_i := \max\{-x_i \,\text{sign}(u_i - x_i), (1 - x_i)\,\text{sign}(u_i - x_i)\}$$

as $w_i \in [0, \gamma_i]$. Note that if $\gamma_i = 0$ this fixes the variable $w_i = 0$ and we then remove this variable from the optimization problem. Thus wlog we assume in the following that $\gamma_i > 0$. The corresponding Lagrangian becomes

$$\begin{aligned} L(w, \alpha, \beta, \lambda_e) = &\frac{1}{2} \sum_{i=1}^d (|u_i - x_i| - w_i)^2 + \lambda_e(\langle \mathbf{1}, w \rangle - \epsilon) \\ &+ \langle \alpha, w - \gamma \rangle - \langle \beta, w \rangle, \end{aligned}$$

which yields the KKT optimality conditions for the optimal primal variable w^* and dual variables $\alpha^*, \beta^*, \lambda_e^*$

$$\begin{aligned} \nabla_w L_i = w_i^* - |u_i - x_i| + \lambda_e^* + \alpha_i^* - \beta_i^* = 0 \\ \lambda_e^*(\sum_{i=1}^d w_i^* - \epsilon) = 0 \\ \alpha_i^*(w_i - \gamma_i) = 0, \qquad \beta_i^* w_i = 0 \\ \lambda_e^* \geq 0, \ \alpha_i^* \geq 0, \ \beta_i^* \geq 0 \end{aligned}$$

Thus $\beta_i^* > 0$ implies $w_i^* = 0$ and with $\gamma_i > 0$ this yields $\alpha_i^* = 0$ and thus $\beta^* = \lambda_e - |u_i - x_i|$ and we get

$$\beta_i^* = \max\{0, \lambda_e^* - |u_i - x_i|\}.$$

On the other hand $\alpha_i^* > 0$ implies $w_i^* = \gamma_i$ and $\beta_i^* = 0$ and thus $\alpha_i^* = |u_i - x_i| - \gamma_i - \lambda_e^*$ and thus

$$\alpha_i^* = \max\{0, |u_i - x_i| - \gamma_i - \lambda_e^*\}.$$

Thus we have in total

$$w_i^* = \begin{cases} \gamma_i & \text{if } |u_i - x_i| - \gamma_i - \lambda_e^* > 0 \\ 0 & \text{if } \lambda_e^* - |u_i - x_i| > 0 \\ |u_i - x_i| - \lambda_e^* & \text{else.} \end{cases}$$

which can be summarized as $w_i^* = \max\{0, \min\{|u_i - x_i| - \lambda_e^*, \gamma_i\}\}$. Finally, if $\sum_{i=1}^d \max\{0, \min\{|u_i - x_i|, \gamma_i\}\} < \epsilon$ then $\lambda_e^* = 0$ is optimal, otherwise $\lambda_e^* > 0$ and we get the solution from the KKT condition

$$\sum_{i=1}^d \max\{0, \min\{|u_i - x_i| - \lambda_e^*, \gamma_i\}\} = \epsilon. \tag{11}$$

Noting that

$$\phi(\lambda_e) = \max\{0, \min\{|u_i - x_i| - \lambda_e, \gamma_i\}\}$$

is a piecewise linear and monotonically decreasing function in λ_e, the solution can be found by sorting the union of $|u_i - x_i| - \gamma_i$ and $|u_i - x_i|$ in non-decreasing order π and then starting with $\lambda_e = 0$ and then going through the sorted list until $\phi(\lambda_e) < \epsilon$. In this case one has identified the interval which contains the optimal solution λ_e^* and computes the solution with the "active" set of components

$$\{i \,|\, 0 < |u_i - x_i| - \lambda_e < \gamma_i\},$$

via Equation (11). The algorithm is provided in Algorithm 2, where the main complexity is the initial sorting step $O(2d\log(2d))$ and some steps in $O(d)$ so that the total complexity is $O(2d\log(2d))$. Once we transform back to the original variable of (3) we get with the form of γ the solution

$$\begin{aligned} z_i^* &= x_i + \text{sign}(u_i - x_i)w_i^* \\ &= \begin{cases} 1 & \text{for } u_i \geq x_i \text{ and } 0 \leq \lambda_e^* \leq u_i - 1 \\ u_i - \lambda_e^* & \text{for } u_i \geq x_i \text{ and } u_i - 1 < \lambda_e^* \leq u_i - x_i \\ x_i & \text{for } \lambda_e^* > |u_i - x_i| \\ u_i + \lambda_e^* & \text{for } u_i \leq x_i \text{ and } -u_i < \lambda_e^* \leq x_i - u_i \\ 0 & \text{for } u_i \leq x_i \text{ and } 0 \leq \lambda_e^* \leq -u_i \end{cases} \end{aligned}$$

□

In order to argue about the approximate projection $A(u)$ we need the same analysis for the l_1-projection which can be derived in an analogous way and the solution for the projection onto $B_1(x, \epsilon)$ in (4) has the form:

$$z_1 = \begin{cases} u_i - \lambda_1^* & \text{for } u_i \geq x_i \text{ and } \lambda_1^* \leq |u_i - x_i| \\ x_i & \text{for } \lambda_1^* > |u_i - x_i| \\ u_i + \lambda_1^* & \text{for } u_i \leq x_i \text{ and } \lambda_1^* \leq |u_i - x_i| \end{cases},$$

where the optimal $\lambda_1^* \geq 0$ is equal to 0 if $\sum_{i=1}^d \max\{0, |u_i - x_i|\} \leq \epsilon$ and otherwise it fulfills

$$\sum_{i=1}^d \max\{0, |u_i - x_i| - \lambda_1^*\} = \epsilon.$$

The two prior versions of PGD (Tramèr & Boneh, 2019; Maini et al., 2020) for the l_1-threat model use instead of the exact projection P_S the approximation $A : \mathbb{R}^d \to S$

$$A(u) = (P_H \circ P_{B_1(x,\epsilon)})(u).$$

The following proof first shows that $A(u) \in S$. However, it turns out that the approximation $A(u)$ "hides" parts of S due to the following property. Note that the condition is is slightly deviating from the one of Lemma 2.1 due to a corner case when $\|u - x\|_1 = \epsilon$. Thus we know require $\|u - x\|_1 > \epsilon$ which then implies that $\|P_S(u) - x\|_1 = \epsilon$.

Lemma 2.1 *It holds for any $u \in \mathbb{R}^d$,*

$$\|P_S(u) - x\|_1 \geq \|A(u) - x\|_1 .$$

In particular, if $P_{B_1(x,\epsilon)}(u) \notin H$ and $\|u - x\|_1 > \epsilon$ and one of the following conditions holds

- $\|P_S(u) - x\|_1 = \epsilon$
- $\|P_S(u) - x\|_1 < \epsilon$ *and* $\exists u_i \in [0, 1]$ *with* $u_i \neq x_i$

then

$$\|P_S(u) - x\|_1 > \|A(u) - x\|_1 .$$

Proof. It holds $A(u) \in H$ but also $A(u) \in B_1(x, \epsilon)$ as with $z_1 := P_{B_1(x,\epsilon)}(u)$

$$\|z_1 - x\| \leq \epsilon,$$

and it holds with $P_H(z) = \max\{0, \min\{z, 1\}\}$ and $z_A := P_H(z_1) = A(u)$ that

$$|z_{1,i} - x_i| = |z_{1,i} - z_{A,i} + z_{A,i} - x_i| = |z_{1,i} - z_{A,i}| + |z_{A,i} - x_i|,$$

which follows as if $z_{1,i} > 1$ then $z_{1,i} - z_{A,i} = z_{1,i} - 1 > 0$ and $1 - x_i \geq 0$ whereas if $z_{1,i} < 0$ then $z_{1,i} - z_{A,i} = z_{1,i} - 0 < 0$ and $0 - x_i \leq 0$. Thus we get

$$\|z_1 - x_1\|_1 = \|z_1 - z_A\|_1 + \|z_A - x\|_1 , \qquad (12)$$

Thus it holds $\|z_A - x\|_1 \leq \|z_1 - x\|_1 \leq \epsilon$ and we get $A(u) \in H \cap B_1(x, \epsilon)$. and thus it is a feasible point of the optimization problem in (3) and by the optimality of $z_e := P_{B_1(x,\epsilon)\cap H}(u)$ it follows

$$\|z_e - u\|_2 \leq \|A(u) - u\|_2 .$$

If $\|z_1 - x\|_1 < \epsilon$ then $z_1 = u$ and thus with (12) one gets $\|z_A - x\|_1 < \epsilon$ as well. In this case z_A is the optimal projection if we just had the box constraints. However, as $\|z_A - x\|_1 < \epsilon$ it is also feasible for (3) and thus optimal, $z_e = z_A$. Moreover, if $\|u - x\|_1 = \epsilon$ then $z_1 = u$ and thus with the same argument we get $z_e = z_A$. On the other hand if $\|z_1 - x\|_1 = \epsilon$ and $z_1 \in H$, then z_1 is feasible for (3) and the minimum over a larger set and thus $z_e = z_1 = z_A$.

Suppose now that $\|u - x\|_1 > \epsilon$ and $z_1 \notin H$. Then $\|z_1 - u\|_1 = \epsilon$ and there exists $\lambda_1^* > 0$ (optimal solution of the l_1-projection problem) such that

$$z_A = \begin{cases} \min\{u_i - \lambda_1^*, 1\} & \text{for } u_i \geq x_i \text{ and } \lambda_1^* \leq |u_i - x_i| \\ x_i & \text{for } \lambda_1^* > |u_i - x_i| \\ \max\{u_i + \lambda_1^*, 0\} & \text{for } u_i \leq x_i \text{ and } \lambda_1^* \leq |u_i - x_i| \end{cases},$$

where we have used that $u_i - \lambda_1^* \geq x_i$ for $u_i \geq x_i$ resp. $u_i + \lambda_1^* \leq x_i$ for $u_i \leq x_i$. Moreover, as $z_1 \notin H$ this implies

that $\|z_A - x\|_1 < \|z_1 - x\|_1 = \epsilon$. Then if $\|z_e - x\|_1 = \epsilon$ (which implies $\lambda_e^* \leq \lambda_1^*$) there is nothing to prove (we get $\|z_e - x\|_1 > \|z_A - x\|_1$ and this represents the first condition in the Lemma for strict inequality) so suppose that $\lambda_e^* = 0$ (that is $\|z_e - x\|_1 < \epsilon$) and thus

$$z_e = \begin{cases} \min\{u_i, 1\} & \text{for } u_i \geq x_i \text{ and } |u_i - x_i| \geq 0 \\ x_i & \text{for } |u_i - x_i| < 0 \\ \max\{u_i, 0\} & \text{for } u_i \leq x_i \text{ and } |u_i - x_i| \geq 0 \end{cases}.$$

Then we get

$$|(z_e)_i - x_i| \geq |(z_A)_i - x_i| \quad \forall i = 1, \dots, d.$$

and thus $\|z_e - x\|_1 \geq \|z_A - x\|_1$. In fact, a stronger property can be derived under an extra condition on u. As $\|z_1 - x\|_1 = \epsilon$ it must hold $\|u - x\|_1 \geq \epsilon$ and thus $u \neq x$. Now suppose that wlog $u_i > x_i$ and $u_i \in [0, 1]$ then $(z_A)_i = u_i - \lambda_1^*$ if $|u_i - x_i| \geq \lambda_1^*$ or $(z_A)_i = x_i$ if $|u_i - x_i| < \lambda_1^*$. However, in both cases we get

$$(z_e)_i - x_i = u_i - x_i > \max\{u_i - x_i - \lambda_1^*, 0\} \geq (z_A)_i - x_i,$$

and thus $\|z_e - x\|_1 > \|z_A - x\|_1$. If $u_i < x_i$ and $u_i \in [0, 1]$ then

$$(z_e)_i - x_i = u_i - x_i < \min\{u_i - x_i - \lambda_1^*, 0\} \leq (z_A)_i - x_i$$

and thus $\|z_e - x\|_1 > \|z_A - x\|_1$.

□

In Figure 4 we simulate the projections after the steepest descent step (8) (for varying level of sparsity) to get a realistic picture of the influence of the approximation $A(u)$ versus the exact projection $P_S(u)$. For sparse updates (less than 50) the approximation behaves quite poorly in comparison to the exact projection in the sense that the true projection is located at the boundary of the l_1-ball around the target point x whereas $A(u)$ is located far into the interior of the l_1-ball. Note that such sparse updates are used in SLIDE (Tramèr & Boneh, 2019) and using the exact projection improves SLIDE (see Figure 2). Next we derive the form of the steepest descent step.

Proposition 2.2 *Let* $z_i = \max\{(1 - x_i)\,\text{sign}(w_i), -x_i\,\text{sign}(w_i)\}$, π *the ordering such that* $|w_{\pi_i}| \geq |w_{\pi_j}|$ *for* $i > j$ *and* k *the smallest integer for which* $\sum_{i=1}^k z_{\pi_i} \geq \epsilon$, *then the solution of* (6) *is given by*

$$\delta_{\pi_i}^* = \begin{cases} z_{\pi_i} \cdot \text{sign}(w_{\pi_i}) & \text{for } i < k, \\ (\epsilon - \sum_{i=1}^{k-1} z_{\pi_i}) \cdot \text{sign}(w_{\pi_k}) & \text{for } i = k, \\ 0 & \text{for } i > k \end{cases}. \quad (13)$$

Proof. We introduce the new variable $\alpha_i := \text{sign}(w_i)\delta_i$, with the convention $\text{sign}(t) = 0$ if $t = 0$. Then we get the equivalent optimization problem:

$$\max_{\alpha \in \mathbb{R}_+^d} \sum_i |w_i| \alpha_i \quad (14)$$
$$\text{s.th.} \sum_{i=1}^d \alpha_i \leq \epsilon, \quad \alpha_i \geq 0,$$
$$-x_i \leq \text{sign}(w_i)\alpha_i \leq 1 - x_i, \; i = 1, \dots, d.$$

and thus

$$0 \leq \alpha_i \leq \max\{-x_i \text{sign}(w_i), (1 - x_i)\text{sign}(w_i)\}.$$

Given the positivity of α and the upper bounds the maximum is attained when ordering $|w_i|$ in decreasing order π and setting always α_{π_i} to the upper bound until the budget $\sum_{i=1}^d \alpha_i = \epsilon$ is attained. Thus with k being the smallest integer in $[1, d]$ such that $\sum_{i=1}^k u_i \leq \epsilon$ we get the solution

$$\alpha_{\pi_i}^* = \begin{cases} z_{\pi_i} & \text{for } i < k, \\ (\epsilon - \sum_{i=1}^{k-1} z_{\pi_i}) & \text{for } i = k, \\ 0 & \text{for } i > k \end{cases}.$$

Noting that $\delta_i = \text{sign}(w_i)\alpha_i$ we get

$$\delta_{\pi_i}^* = \begin{cases} z_{\pi_i}\text{sign}(w_{\pi_i}) & \text{for } i < k, \\ (\epsilon - \sum_{i=1}^{k-1} z_{\pi_i})\text{sign}(w_{\pi_i}) & \text{for } i = k, \\ 0 & \text{for } i > k \end{cases}.$$

□

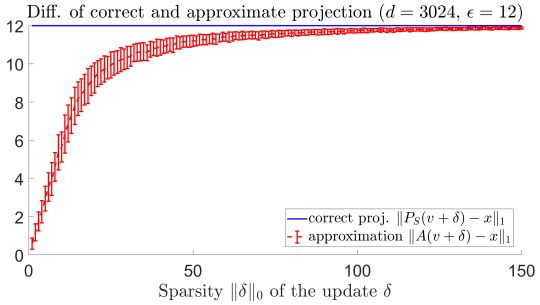

Figure 4. In order to simulate update steps of PGD, we sample target points $x \sim \mathcal{U}([0, 1]^d)$ and $w \in \mathcal{N}(0, 1)$ and define $v = P_S(w)$ and project the point $u = v + \epsilon\delta$, where δ is generated as the descent step in (8) for different sparsity levels k. Then we plot $\|P_S(u) - x\|_1$ and $\|A(u) - x\|_1$ for varying sparsity k of the update (for each level k we show average and standard deviation over 100 samples). It can clearly be seen that the approximate projection is highly biased towards interior regions of the set $S = B_1(x, \epsilon) \cap [0, 1]^d$ in particular for small levels of k whereas the correct projection stays on the surface of S. Note that SLIDE uses $k = 31$ (corresponds to 99% quantile) and thus is negatively affected by this strong bias as effectively large portions of the threat model S cannot be easily explored by SLIDE.

Next we show the expected sparsity of the steepest descent step.

Proposition 2.3 *Let $w \in \mathbb{R}^d$ with $w_i \neq 0$ for all $i = 1, \dots, d$ and $x \in \mathcal{U}([0,1]^d)$ and let δ^* be the solution from 2.2. Then it holds for any $\frac{d-1}{2} \geq \epsilon > 0$,*

$$\mathbb{E}\big[\, \|\delta^*\|_0 \,\big] = \lfloor \epsilon + 1 \rfloor + \sum_{m=\lfloor \epsilon \rfloor + 2}^{d} \sum_{k=0}^{\lfloor \epsilon \rfloor} (-1)^k \frac{(\epsilon - k)^{m-1}}{k!\,(m-1-k)!} \geq \frac{\lfloor 3\epsilon \rfloor + 1}{2}.$$

Proof. We first note that the components z_i defined in Proposition 2.2 are independent and have distribution $z_i \sim \mathcal{U}([0,1])$, $i = 1, \dots, d$ as both $1 - x_i$ and x_i are uniformly distributed and the x_i are independent. As the z_i are i.i.d. the distribution of k is independent of the ordering of w and thus we can just consider the probability $\sum_{i=1}^{k} z_i \geq \epsilon$. Note that for any $\epsilon > 0$, we have $1 \leq k \leq d$. Moreover, we note that

$$k > m \iff \sum_{i=1}^{m} z_i < \epsilon \text{ and } k = d \iff \sum_{i=1}^{d} z_i \leq \epsilon,$$

and as k is an integer valued random variable we have

$$k \geq m \iff k > m - 1.$$

Thus as k is a non-negative integer valued random variable, we have

$$\mathbb{E}[k] = \sum_{m=1}^{d} \mathrm{P}(k \geq m) = \sum_{m=1}^{d} \mathrm{P}\big(k > m-1\big) = \sum_{m=1}^{d} \mathrm{P}\Big(\sum_{i=1}^{m-1} z_i < \epsilon\Big) = \sum_{m=1}^{d} \mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big)$$

where in the last step we use that $\sum_{i=1}^{m-1} z_i$ is a continuous random variable and thus we add a set of measure zero. The sum of uniformly distributed random variables on $[0,1]$ has the Irwin-Hall distribution with a cumulative distribution function (Irwin, 1927; Hall, 1927) given by

$$\mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big) = \frac{1}{(m-1)!} \sum_{k=0}^{\lfloor \epsilon \rfloor} (-1)^k \binom{m-1}{k} (\epsilon - k)^{m-1}.$$

Note that the distribution of $\sum_{i=1}^{m-1} z_i$ is symmetric around the mean value $\frac{m-1}{2}$ and thus the median is also $\frac{m-1}{2}$. We note that for $m = 1$ the first sum is empty and thus $\mathrm{P}\Big(\sum_{i=1}^{0} z_i \leq \epsilon\Big) = 1$ and in general as $0 \leq z_1 \leq 1$ it holds for $d - 1 \geq \epsilon \geq m - 1$:

$$\mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big) = 1$$

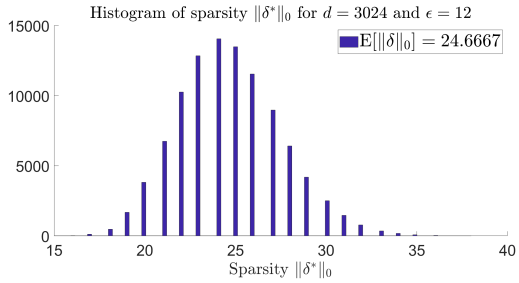

Figure 5. Histogram of the sparsity level $\|\delta^*\|_0$ of the steepest descent step from Proposition 2.3 for $d = 3024$ and $\epsilon = 12$ (histogram computed using 100.000 samples from $x \in \mathcal{U}([0,1]^d)$ and $w \in \mathcal{N}(0, \mathbb{I})$). The exact expected sparsity can be computed as $\mathbb{E}[\|\delta^*\|_0] \approx 24.6667$.

Thus

$$\sum_{m=1}^{d} \mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big) = \lfloor \epsilon + 1 \rfloor + \sum_{m=\lfloor \epsilon+1 \rfloor + 1}^{d} \mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big).$$

As the median is given by $\frac{m-1}{2}$ we get for $\epsilon \geq \frac{m-1}{2}$

$$\mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big) \geq \frac{1}{2}$$

and thus

$$\begin{aligned}\sum_{m=1}^{d} \mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big) &\geq \lfloor \epsilon + 1 \rfloor + \frac{\lfloor 2\epsilon + 1 \rfloor - \lfloor \epsilon + 1 \rfloor}{2} + \sum_{m=\lfloor 2\epsilon+1 \rfloor + 1}^{d} \mathrm{P}\Big(\sum_{i=1}^{m-1} z_i \leq \epsilon\Big) \\ &\geq \frac{\lfloor \epsilon + 1 \rfloor + \lfloor 2\epsilon + 1 \rfloor}{2} \geq \frac{\lfloor 3\epsilon \rfloor + 1}{2}\end{aligned}$$

□

Note that tighter lower bounds could be derived with more sophisticated technical tools but we decided for the simple argument as we just want to show that the sparsity is non-trivially lower bounded. The exact expression is difficult to evaluate in high dimensions. In Figure 5 we provide an empirical evaluation of the sparsity of δ^* for $d = 3024$ and $\epsilon = 12$ for 100.000 samples from the uniform distribution on $[0,1]^d$ (w is drawn from a standard multivariate Gaussian). The exact expected sparsity is about 24.6667 which is slightly higher than 2ϵ. Thus 2ϵ is a reasonable guideline in practice.

B. Adversarial training wrt l_1

As mentioned in Sec. 3.4, adversarial training (Madry et al., 2018) in the $l_1 \cap [0,1]^d$-threat model is more delicate than for

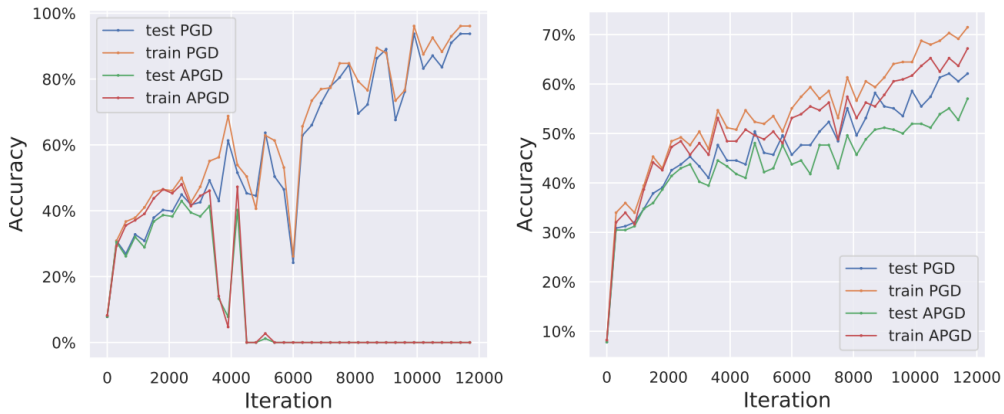

Figure 6. **Left:** Illustration of catastrophic overfitting in l_1 adversarial training when using the 10-step PGD-attack SLIDE (Tramèr & Boneh, 2019) with $k = 0.01$ for training. While the model is robust against the 10-step SLIDE attack, it is completely non-robust when running our stronger l_1-APGD attack with 100 iterations. **Right:** Catastrophic overfitting does not happen when using 10-step l_1-APGD for adversarial training to train AT-APGD, even when one attacks the model with much stronger attacks at test time, see the evaluation in Table 3. In both plots we report the robust accuracy on training and test set over the 30 epochs of training.

l_∞ or l_2. In particular, we observe that using the *multi-step* PGD-based attack SLIDE with the standard sparsity of the updates $k = 0.01$ (here and in the following k indicates the percentage of non zero elements) leads to catastrophic overfitting when training classifiers on CIFAR-10 with $\epsilon = 12$ and 10 steps. Here we mean by catastrophic overfitting that model is robust against the attack used during training even at test time but it fails completely against a stronger attack (APGD in the left part of Figure 6). This is similar to what has been reported by (Wong et al., 2020) in the l_∞-threat model using FGSM (Goodfellow et al., 2014), i.e. a single step method, in adversarial training might produce classifiers resistant to FGSM but not to a stronger PGD attack. Moreover, the robustness against PGD drops abruptly during training, on both training and test sets. In our scenario, as illustrated in Figure 6 by the left plot, a similar phenomenon happens: when we use SLIDE in adversarial training the classifier is initially robust against both SLIDE with 10 steps (blue and yellow curves) and the stronger l_1-APGD (green and red) with 100 iterations, but around iteration 4000 the robustness computed with l_1-APGD goes close to 0%, while the model is still very robust against the attack used for training (which is a 10-step attack and not a single step attack). Conversely, when l_1-APGD with 10 steps is used for training (right plot) the model stays robust against both l_1-APGD with 100 iterations and SLIDE (and other attacks as shown in Sec. 5).

We could prevent catastrophic overfitting by decreasing the sparsity in SLIDE to $k = 0.05$, but this yields poor final robustness (around 50%) compared to what we achieved using l_1-APGD (59.7% against multiple attacks, see Table 2), since a weaker attack is used at training time. In fact, we show in Sec. E.1 that values $k > 0.01$ in SLIDE lead to worse performance in most of the cases. We hypothesize that l_1-APGD, adapting the sparsity of the updates per point, first prevents the model from seeing only perturbations with similar sparsity and second is able to effectively maximize the loss, allowing adversarial training to perform. We note that even (Tramèr & Boneh, 2019) observed that using the standard fixed value of k leads to overfitting, and proposed as remedies random sparsity levels and 20 steps. We instead keep the usual 10 steps, and have a scheme which adaptively chooses the sparsity rather than randomly.

C. l_1-Square Attack

(Andriushchenko et al., 2020) introduce the Square Attack, a black-box score-based adversarial attacks, based on random search with square-shaped updates. The key component of such scheme is using an effective distribution to sample at each iteration a new candidate update of the current best point, i.e. achieving the best loss. We adapt the algorithm proposed for l_2-bounded attacks and give in Alg. 3 the detailed procedure constituting the sampling distribution of our version of Square Attack for the $l_1 \cap [0, 1]^d$-threat model. If the new point $x^{(i)} + \delta$ attains a lower margin loss (since in this case we aim at minimizing the difference between the logit of the correct class and the largest of the others, until a different classification is achieved), then $x^{(i+1)} = x^{(i)} + \delta$, otherwise $x^{(i+1)} = x^{(i)}$. The input w of Alg. 3 controls the size of the update and is progressively reduced according to a piecewise constant schedule, in turn regulated by the only free parameter of the method p.

Algorithm 2 Projection onto $B_1(x, \epsilon) \cap [0, 1]^d$

Input: point to be projected u, x and radius ϵ
Output: projection z onto $B_1(x, \epsilon) \cap [0, 1]^d$
$\gamma_i = \max\{-x_i \operatorname{sign}(u_i - x_i), (1 - x_i) \operatorname{sign}(u_i - x_i)\}$, $\lambda^* = 0$
$z_i = \min\{|u_i - x_i|, \gamma_i\}$
$S = \sum_{i=1}^d z_i$
if $S > \epsilon$ **then**
  sort $t_i = \{|u_i - x_i|, |u_i - x_i| - \gamma_i\}$ in decreasing order π and memorize if it is $|u_i - x_i|$ (category 0) or $|u_i - x_i| - \gamma$ (category 1)
  $M = |\{i \,|\, z_i = |u_i - x_i| \text{ and } |u_i - x_i| > 0\}|$
  $\lambda = 0$
  for $j = 1$ **to** 2d **do**
    $\lambda_{\text{old}} = \max\{0, \lambda\}$
    $\lambda = t_{\pi_j}$
    $S = S - M(\lambda - \lambda_{\text{old}})$
    if category$(\pi_j) = 0$ **then**
      $M = M + 1$
    else
      $M = M - 1$
    end if
    if $S < \epsilon$ **then**
      if category$(\pi_j) = 0$ **then**
        $M = M - 1$
      else
        $M = M + 1$
      end if
      $S = S + M(\lambda - \lambda_{\text{old}})$
      $\lambda^* = \lambda_{\text{old}} + (S - \epsilon)/M$
      BREAK
    end if
  end for
end if
$z_i = \max\{0, \min\{|u_i - x_i| - \lambda^*, \gamma_i\}\}, \; i = 1, \dots, d$
$P_S(u)_i = x_i + \operatorname{sign}(u_i - x_i) \cdot z_i$;

Algorithm 3 Sampling distribution in Square Attack for $l_1 \cap [0, 1]^d$

Input: target point x of shape $h \times h \times c$, radius ϵ, current iterate $x^{(i)}$, size of the windows w
Output: new update δ
$\nu \leftarrow x^{(i)} - x$
sample uniformly $r_1, s_1, r_2, s_2 \in \{0, \dots, w - h\}$
$W_1 := r_1 + 1 : r_1 + h, s_1 + 1 : s_1 + h$, $W_2 := r_2 + 1 : r_2 + h, s_2 + 1 : s_2 + h$
$\epsilon_{\text{unused}} \leftarrow \epsilon - \|\nu\|_1$,
$\eta^* \leftarrow \eta / \|\eta\|_1$ with η as in Eq. (2) of (Andriushchenko et al., 2020)
for i = 1 **to** c **do**
  $\rho \leftarrow Uniform(\{-1, 1\})$
  $\nu_{\text{temp}} \leftarrow \rho\eta^* + \nu_{W_1,i} / \|\nu_{W_1,i}\|_1$
  $\epsilon^i_{\text{avail}} \leftarrow \|\nu_{W_1 \cup W_2,i}\|_1 + \epsilon_{\text{unused}}/c$
  $\nu_{W_2,i} \leftarrow 0, \quad \nu_{W_1,i} \leftarrow (\nu_{\text{temp}} / \|\nu_{\text{temp}}\|_1)\epsilon^i_{\text{avail}}$
end for
$z \leftarrow P_S(x + 3\nu)$
$\delta \leftarrow z - x^{(i)}$

D. Experimental details

We here report details about the experimental setup used in Sec. 5.

D.1. Models

Almost all the models we used are publicly available: the classifiers from (Engstrom et al., 2019; Carmon et al., 2019; Rice et al., 2020; Augustin et al., 2020) are provided in the library RobustBench (Croce et al., 2020). Those of (Maini et al., 2020; Xu & Yang, 2020) can be found in the official pages[2,3]. Moreover, (Madaan et al., 2021; Xiao et al., 2020) made models available via OpenReview[4,5]. Upon request, (Kim et al., 2020) could not give access to the original models out of privacy reasons. Therefore we trained new classifiers using the official code[6] following the suggested parameters. For the models denoted in (Kim et al., 2020) by "RoCL" and "RoCL+rLE" we could reproduce both clean and robust accuracy wrt l_∞ with the original evaluation code, while for "RoCL+AT+SS" we could match the robust accuracy but not the clean one (here robust accuracy is the one computed using their code). However, we used this last one in our experiments since it is the most robust one wrt l_1.

Table 5. Architecture of the models used in the experimental evaluation on CIFAR-10.

model	*architecture*
APGD-AT **(ours)**	PreAct ResNet-18
(Madaan et al., 2021)	WideResNet-28-10
(Maini et al., 2020) - AVG	PreAct ResNet-18
(Maini et al., 2020) - MSD	PreAct ResNet-18
(Augustin et al., 2020)	ResNet-50
(Engstrom et al., 2019) - l_2	ResNet-50
(Rice et al., 2020)	PreAct ResNet-18
(Xiao et al., 2020)	DenseNet-121
(Kim et al., 2020)*	ResNet-18
(Carmon et al., 2019)	WideResNet-28-10
(Xu & Yang, 2020)	ResNet-18
(Engstrom et al., 2019) - l_∞	ResNet-50

For APGD-AT we trained a PreAct ResNet-18 (He et al.,

[2] https://github.com/locuslab/robust_union
[3] https://github.com/MTandHJ/amoc
[4] https://openreview.net/forum?id=tv8n52XbO4p
[5] https://github.com/iclrsubmission/kwta
[6] https://github.com/Kim-Minseon/RoCL

2016) with softplus activation function, using cyclic learning rate with maximal value 0.1 for 100 epochs, random cropping and horizontal flipping as training set augmentations. We set 10 steps of APGD for maximizing the robust loss in the standard adversarial training setup (Madry et al., 2018).

Table 5 reports the architecture of every model. As mentioned in Sec. 5, we chose such models to have different architectures, training schemes and even training data, as (Carmon et al., 2019; Augustin et al., 2020) use unlabeled data in their methods.

D.2. Attacks

In the following we report the details of the presented attacks. We use ALMA from Adversarial Library (Rony & Ben Ayed, 2020), with the default parameters for the l_1-threat models, in particular $\alpha = 0.5$ with 100 iterations, $\alpha = 0.9$ with 1000. EAD, B&B and Pointwise Attack (PA) are available in FoolBox (Rauber et al., 2017): for EAD we use the l_1 decision rule and regularization $\beta = 0.01$, for B&B we keep the default setup, while PA does not have tunable parameters. We reimplemented SLIDE following the original code, according to which we set sparsity of the updates $k = 0.01$ for CIFAR-10 and step size $\eta = 3.06$, which is obtained rescaling the one used in (Tramèr & Boneh, 2019) $\eta = 2$ for $\epsilon = 2000/255$ (see below for a study of the effect of different values of k). Also, since no code is available for ZO-ADMM for l_1, we adapted the l_2 version following (Zhao et al., 2019) and then optimized its parameters, using $\rho = 2$, $\gamma = 0.1$. Finally, we use FAB$^{\text{T}}$ as available in AutoAttack.

For l_1-APGD we fix the values of all parameters to those mentioned in Sec. 3.1 and Sec. 3.2. Moreover, we set $p = 0.8$ in l_1-Square Attack as done in AutoAttack for l_2 and l_∞. Thus even l_1-AutoAttack can be used without any parameter tuning.

Attacks runtime: Direct comparison of runtime is not necessarily representative of the computational cost of each method since it depends on many factors including implementation and tested classifier. We gave similar budget to (almost all) attacks: for the low budget comparison (see Table 1) we use 100 iterations, which are equivalent to 100 forward and 100 backward passes for ALMA, SLIDE and l_1-APGD, 110 forward and 100 backward passes for B&B (because of the initial binary search), 150 forward and 100 backward passes for FAB$^{\text{T}}$. EAD has instead a 9 times larger budget since we keep the default 9 binary search steps. As an example, when run using a classifier on CIFAR-10 with PreAct ResNet-18 as architecture, 1000 test points, ALMA and SLIDE take around 25 s, l_1-APGD 27 s, FAB$^{\text{T}}$ 32 s, EAD 105 s, B&B 149 s.

E. Additional experiments

E.1. Effect of sparsity in SLIDE

Since the sparsity k of the updates is a key parameter in SLIDE, the PGD-based attack for l_1 proposed in (Tramèr & Boneh, 2019), we study the effect of varying k on its performance. In Table 6 we report the robust accuracy achieved by SLIDE with 5 values of $k \in \{0.001, 0.003, 0.01, 0.03, 0.1\}$ on the CIFAR-10 models used for the experiments in Sec. 5, with a single run of 100 iterations. As a reference, we also show the results of our l_1-APGD with the same budget (grey column). We observe that while the default value $k = 0.01$ performs best in most of the cases, for 3/12 models the lowest robust accuracy is obtained by $k = 0.03$, for 1/12 by $k = 0.001$. This means that SLIDE would require to tune the value of k for each classifier to optimize its performance. Moreover, l_1-APGD, which conversely automatically adapts the sparsity of the updates, outperforms the best out of the 5 versions of SLIDE in 11 out of 12 cases, with the only exception of the model from (Xiao et al., 2020) which is know to present heavy gradient obfuscation and on which the black-box Square Attack achieves the best result (see Sec. 5).

E.2. Larger threshold ϵ

We here test that the performance of our attacks at the larger threshold $\epsilon = 16$. In Table 7 we run all the methods, with the lower budget, on the four most robust models: one can observe that even with a larger threshold our l_1-APGD outperforms the competitors on all models.

E.3. Other datasets

We test the effectiveness of our proposed attacks on CIFAR-100 and ImageNet-1k, with $\epsilon = 12$ and $\epsilon = 60$ respectively, in the same setup of Sec. 5. For CIFAR-100 we use the models (PreAct ResNet-18) from (Rice et al., 2020), for ImageNet those (ResNet-50) of (Engstrom et al., 2019): in both cases one classifier is trained for l_∞-robustness, the other one for l_2, all are publicly available[7,8]. On ImageNet, because of the different input dimension, we use $k = 0.001$ for SLIDE (after tuning it), and we do not run Pointwise Attack since it does not scale. For B&B we use random images not classified in the target class from the respective test or validation sets as starting points. We observe that on ImageNet, the gap in runtime between B&B and the faster attacks increases significantly: for example, to run 100 steps for 1000 test points B&B takes 3612 s, that is around 14 times more than l_1-APGD (254 s). Thus B&B

[7] https://github.com/locuslab/robust_overfitting

[8] https://github.com/MadryLab/robustness/tree/master/robustness

Table 6. Effect of the sparsity k of the updates in SLIDE (Tramèr & Boneh, 2019), whose default value is $k = 0.01$.

model	$k = 0.001$	$k = 0.003$	$k = 0.01$	$k = 0.03$	$k = 0.1$	APGD_{CE}
APGD-AT **(ours)**	74.5	70.1	66.6	64.4	65.7	**61.3**
(Madaan et al., 2021)	61.9	58.2	56.1	57.0	66.2	**54.7**
(Maini et al., 2020) - AVG	71.7	64.4	53.8	51.8	64.7	**50.4**
(Maini et al., 2020) - MSD	63.6	58.5	53.2	51.7	62.2	**49.7**
(Augustin et al., 2020)	60.9	53.0	48.8	59.3	74.1	**37.1**
(Engstrom et al., 2019) - l_2	49.6	40.0	35.1	47.4	67.4	**30.2**
(Xiao et al., 2020)	**28.2**	30.1	33.3	36.1	45.4	41.4
(Rice et al., 2020)	47.0	37.6	32.3	45.2	65.2	**27.1**
(Kim et al., 2020)*	38.4	30.6	25.1	34.9	58.3	**18.9**
(Carmon et al., 2019)	41.4	29.9	19.7	24.2	64.4	**13.1**
(Xu & Yang, 2020)	35.3	24.9	18.2	21.1	58.2	**10.9**
(Engstrom et al., 2019) - l_∞	34.0	23.6	14.2	17.0	59.4	**8.0**

Table 7. **Low Budget ($\epsilon = 16$):** Robust accuracy achieved by the SOTA l_1-adversarial attacks on models for CIFAR-10 at $\epsilon = 16$.

model	clean	EAD	ALMA	SLIDE	B&B	FAB^{T}	APGD_{CE}	PA	Square
APGD-AT **(ours)**	87.1	55.2	55.3	59.1	52.7	59.7	**50.6**	78.5	66.4
(Madaan et al., 2021)	82.0	46.2	50.0	47.7	46.0	48.2	**45.6**	70.4	58.5
(Maini et al., 2020) - MSD	82.1	43.9	46.2	44.9	43.2	48.5	**40.9**	69.8	57.9
(Maini et al., 2020) - AVG	84.6	43.7	44.9	45.9	43.0	55.3	**38.9**	75.1	62.9

Table 8. **Low Budget:** Robust accuracy achieved by the SOTA l_1 -adversarial attacks on various models for CIFAR-100 and ImageNet in the l_1-threat model with radius indicated. The statistics are computed on 1000 points of the test set. PA and Square are black-box attacks. The budget is 100 iterations for white-box attacks ($\times 9$ for EAD and +10 for B&B) and 5000 queries for our l_1-Square-Attack.

model	clean	EAD	ALMA	SLIDE	B&B	FAB^{T}	APGD_{CE}	PA	Square
CIFAR-100 ($\epsilon = 12$)									
(Rice et al., 2020) - l_2	58.7	19.5	24.4	17.7	19.3	19.6	**14.6**	37.0	23.8
(Rice et al., 2020) - l_∞	54.5	8.6	9.9	6.0	6.5	13.0	**4.5**	23.0	10.6
ImageNet ($\epsilon = 60$)									
(Engstrom et al., 2019) - l_2	56.6	45.6	50.8	44.7	44.4	44.8	**43.6**	-	50.2
(Engstrom et al., 2019) - l_∞	61.9	11.7	26.6	11.5	9.5	34.6	**6.3**	-	23.9

Table 9. **High Budget:** Robust accuracy achieved by the SOTA l_1 -adversarial attacks on various models for CIFAR-100 and ImageNet in the l_1-threat model with l_1-radius indicated. The statistics are computed on 1000 points of the test set. "WC" denotes the pointwise worst-case over all restarts/runs of EAD, ALMA, SLIDE, B&B and, if available, Pointwise Attack. Note that $\text{APGD}_{\text{CE+T}}$, the combination of APGD_{CE} and $\text{APGD}_{\text{T-DLR}}$ (5 restarts each), yields a similar performance as AA (ensemble of $\text{APGD}_{\text{CE+T}}$, l_1-FAB^T and l_1-Square Attack) with the same or smaller budget than the other individual attacks.

model	clean	EAD	ALMA	SLIDE	B&B	$\text{APGD}_{\text{CE+T}}$	WC	AA
CIFAR-100 ($\epsilon = 12$)								
(Rice et al., 2020) - l_2	58.7	18.4	17.1	15.5	13.1	**12.1**	12.7	**12.1**
(Rice et al., 2020) - l_∞	54.5	8.1	5.5	4.5	3.0	3.4	**2.9**	3.1
ImageNet ($\epsilon = 60$)								
(Engstrom et al., 2019) - l_2	56.6	44.1	45.6	44.2	**40.3**	40.5	**40.3**	40.5
(Engstrom et al., 2019) - l_∞	61.9	9.6	17.1	8.5	6.2	4.6	5.8	**4.4**

scales much worse to high-resolution datasets. Also, while B&B and l_1-APGD have in principle a similar budget in terms of forward/backward passes, one could do much more restarts for l_1-APGD in the same time as for B&B.

We report in Table 8 and Table 9 the robust accuracy given by every attack on 1000 points of test set of CIFAR-100 or validation set of ImageNet. Similarly to CIFAR-10, our l_1-APGD achieves the best results for all models in the low budget regime (see Table 8) with a significant gap to the second best, either B&B or SLIDE. Moreover, when using higher computational budget, l_1-AutoAttack gives the lowest robust accuracy in 2/4 cases, improving up 1.4% over WC, the pointwise worst case over all attacks not included in AA, while in the other cases it is only 0.2% worse than WC, showing that it gives a good estimation of the robustness of the models.

E.4. Composition of l_1-AutoAttack

We report in Table 10 the individual performance of the 4 methods constituting l_1-AutoAttack, recalling that each version of l_1-APGD, with either cross-entropy or targeted DLR loss, is used with 5 runs of 100 iterations, FAB$^{\text{T}}$ exploits 9 restarts of 100 iterations and l_1-Square Attack has a budget of 5000 queries. Note that the robust accuracy given by l_1-AutoAttack is in all cases lower than that of the best individual attack, which varies across models.

Table 10. Individual performance of the components of l_1-AutoAttack.

model	clean	APGD_{CE}	$\text{APGD}_{\text{T-DLR}}$	FAB^{T}	Square	AA
CIFAR-10 ($\epsilon = 12$)						
APGD-AT **(ours)**	87.1	60.8	60.8	65.9	71.8	**60.3**
(Madaan et al., 2021)	82.0	54.2	52.0	54.7	62.8	**51.9**
(Maini et al., 2020) - AVG	84.6	48.9	47.5	59.5	68.4	**46.8**
(Maini et al., 2020) - MSD	82.1	48.6	47.4	53.5	63.5	**46.5**
(Augustin et al., 2020)	91.1	34.7	34.5	42.4	56.8	**31.0**
(Engstrom et al., 2019) - l_2	91.5	27.9	29.3	32.9	52.7	**26.9**
(Rice et al., 2020)	89.1	25.5	26.3	30.3	50.3	**24.0**
(Xiao et al., 2020)	79.4	32.2	33.4	78.6	20.2	**16.9**
(Kim et al., 2020)*	81.9	17.0	16.9	22.2	36.0	**15.1**
(Carmon et al., 2019)	90.3	9.9	9.9	21.5	34.5	**8.3**
(Xu & Yang, 2020)	83.8	9.6	9.3	17.7	32.0	**7.6**
(Engstrom et al., 2019) - l_∞	88.7	6.1	6.7	13.0	28.0	**4.9**
CIFAR-100 ($\epsilon = 12$)						
(Rice et al., 2020) - l_2	58.7	13.4	13.8	16.4	23.8	**12.1**
(Rice et al., 2020) - l_∞	54.5	3.8	4.2	7.7	10.6	**3.1**
ImageNet ($\epsilon = 60$)						
(Engstrom et al., 2019) - l_2	56.6	42.8	40.8	43.1	50.2	**40.5**
(Engstrom et al., 2019) - l_∞	61.9	5.7	5.5	18.9	23.9	**4.4**